

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

Факультет
вычислительной техники

Кафедра
МОиПЭВМ

Направление подготовки 09.03.02 «Информационные системы и технологии»

Направленность Информационные системы и технологии

БАКАЛАВРСКАЯ РАБОТА

на тему

Разработка системы рекомендации товаров в интернет-магазине на основе 1С и машинного обучения

Студент

(подпись, дата)

Федулова Диана Алексеевна

(ФИО полностью)

Руководитель

(подпись, дата)

Попова Н.А.

(фамилия, инициалы)

Нормоконтролёр

(подпись, дата)

Такташкин Д.В.

(фамилия, инициалы)

Работа допущена к защите (протокол заседания кафедры от 03.06.2026г. № 12)

Заведующий кафедрой

(подпись)

Козлов А.Ю.

(фамилия, инициалы)

Работа защищена с отметкой _____ (протокол заседания ГЭК от _____ № _____)

Секретарь ГЭК

(подпись)

Попова Н.А.

(фамилия, инициалы)

Пенза, 2026

Федеральное государственное бюджетное образовательное учреждение
высшего образования
ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

Факультет
Вычислительной техники

Кафедра
МО и ПЭВМ

«Утверждаю»
Заведующий кафедрой
_____ А.Ю. Козлов
«26» декабря 2025 г

**ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
БАКАЛАВРА**

1. Студент _____ Федулова Диана Алексеевна _____ гр. 22ВИ1 _____ факультета ВТ
<Фамилия, имя, отчество полностью>

2. Тема работы _____ Разработка системы рекомендации товаров в интернет-магазине на
основе 1С и машинного обучения _____

Тема утверждена приказом ПГУ №1583/о от "30" декабря 2025 г.

3. Руководитель работы к.т.н. доцент, доцент кафедры «МО и ПЭФВМ» Попова Н.А.

4. Задание на работу (назначение разработки, исходные данные и т.п.)

1) Назначение: разработка системы рекомендации товаров в интернет-магазине на основе
1С и машинного обучения _____

2) Среда выполнения: Windows 11 _____

Языковые средства: 1С, Python, Java, UML _____

Инструментальные средства: Платформа «1С: Предприятие», интерпретатор Python,
веб-сервер Apache 2.4, Visual Paradigm, Apache Tomcat _____

Технологии разработки: Машинное обучение (Collaborative Filtering), косинусное
сходство, разреженные матрицы (CSR), REST API, файловый обмен (CSV) _____

3) Основные функции приложения: _____

а) автоматизированная выгрузка истории продаж из учетной системы 1С; _____

б) построение рекомендательной модели методами машинного обучения; _____

в) расчет сходства товаров методом косинусного сходства; _____

г) передача данных через программный интерфейс (API) в режиме реального времени; _____

д) визуализация блоков рекомендаций в карточке товара на сайте магазина; _____

е) автоматическое создание заказов с сайта в базе 1С _____

5. Перечень подлежащих разработке вопросов

1. Анализ предметной области и постановка задачи

2. Проектирование;

3. Реализация;

4. Контроль качества системы;

5. Оформление пояснительной записки.

6. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1	Анализ предметной области и требований к системе	10%	07.04.2026 – 13.04.2026	
2	Проектирование программных средств	20%	14.04.2026 – 25.04.2026	
3	Реализация программных средств	30%	27.04.2026 – 11.05.2026	
4	Тестирование программных средств	20%	12.05.2026 – 21.05.2026	
5	Оформление пояснительной записки	20%	22.05.2026 – 02.06.2026	

Дата выдачи задания «26» декабря 2025г.

Руководитель бакалаврской работы _____ 26.12.2025г.
(подпись, дата)

Задание к исполнению принял студент _____ 26.12.2025г.
(подпись, дата)

Работу к защите допустить
Декан ФВТ _____ Л.Р. Фионова
(подпись, дата)

Реферат

Пояснительная записка содержит 116 листов, 39 рисунков, 27 таблиц, 23 использованных источников и 4 приложений.

ИНТЕРНЕТ-МАГАЗИН, СИСТЕМА РЕКОМЕНДАЦИЙ, 1С:ПРЕДПРИЯТИЕ 8.3, МАШИННОЕ ОБУЧЕНИЕ, PYTHON, JAVA, КОЛЛАБОРАТИВНАЯ ФИЛЬТРАЦИЯ, КОСИНУСНОЕ СХОДСТВО, REST API, JSON, CSV.

Объект исследования – алгоритмы и программные средства формирования товарных рекомендаций на основе данных о продажах покупателей.

Цель выпускной квалификационной работы – разработать систему товарных рекомендаций на основе интеграции учетных данных 1С и алгоритмов машинного обучения.

Технологии разработки – разработка механизмов подготовки данных и API-интерфейса выполнена на платформе «1С:Предприятие 8.3». Программная реализация модуля проведена на языке программирования Python с использованием библиотек Pandas, Scikit-learn и SciPy. Алгоритм Item-based Коллаборативная фильтрация. Клиентская часть системы разработана на языке Java с использованием фреймворка Spring Boot. Интеграция модулей реализована через протокол HTTP (REST API) в формате JSON и файловый обмен данными в формате CSV.

Среда исполнения – Windows 11, веб-браузер.

Результаты работы – в ходе выполнения выпускной работы был создан программный комплекс, обеспечивающий автоматизированное формирование рекомендаций. От сбора истории продаж в 1С до визуализации рекомендаций на сайте. Проведенное тестирование подтвердило корректность работы алгоритмов и устойчивость системы к нагрузкам.

					ПГУ 09.03.02–8BP131.17 ПЗ			
Изм.	Лист	№ докум.	Подпись	Дата	«Разработка системы рекомендации товаров в интернет-магазине на основе 1С и машинного обучения» Пояснительная записка	Лит.	Лист	Листов
Разраб.		Федулова Д.А.						
Провер.		Попова Н.А.					4	116
Н. Контр.		Такташкин Д.В.				Группа 22ВИ1		
Утверд.		Козлов А.Ю.						

Содержание

Введение.....	7
1 Анализ предметной области и постановка задачи.....	9
1.1 Постановка задачи	9
1.2 Анализ информационных систем аналогичного назначения	10
1.3 Выбор технологий проектирования системы	13
1.4 Выбор инструментальных средств	15
1.5 Анализ функциональных требований к системе.....	16
1.6 Анализ нефункциональных требований к системе	20
1.7 Планирование разработки и оценка бюджета.....	22
2 Проектирование информационной системы	25
2.1 Функциональное моделирование процессов системы.....	25
2.2 Проектирование структуры хранения данных	28
2.3 Проектирование логики взаимодействия программных модулей	36
2.4 Проектирование сценариев обмена данными	38
2.5 Проектирование архитектуры развертывания и логической структуры системы	41
2.6 Проектирование ML-алгоритма товарных рекомендаций.....	43
2.7 Проектирование пользовательского интерфейса	48
2.7.1 Проектирование интерфейса управления в 1С	48
2.7.2 Проектирование интерфейса веб-приложения.....	50
3 Реализация системы.....	51
3.1 Разработка структуры объектов информационной базы 1С	51
3.2 Реализация программных средств для системы рекомендаций.....	59
3.2.1 Реализация алгоритмов учета и обмена данными на языке 1С.....	59
3.2.2 Реализация аналитического модуля на языке Python	63
3.2.3 Реализация серверной и клиентской логики веб-приложения	66
3.3 Реализация пользовательского интерфейса	70
3.3.1 Реализация интерфейса в среде «1С: Предприятие 8.3»	70
3.3.2 Реализация интерфейса веб-приложения интернет-магазина	75
4 Контроль качества системы рекомендаций товаров.....	79

4.1	Обоснование выбора режима тестирования и метрик кода	79
4.2	Функциональное тестирование	80
4.2.1	Классы эквивалентности	80
4.2.2	Граничные условия.....	82
4.3	Нефункциональное тестирование.....	84
4.4	Метрики кода.....	87
4.4.1	Метрика SLOC	87
4.4.2	Метрики Холстеда.....	90
4.4.3	Метрика Чепина	95
	Заключение	98
	Список использованных источников.....	100
	Приложение А (справочное) Глоссарий	103
	Приложение Б (обязательное) Листинг подсистемы «1С: Предприятие».....	106
	Приложение В (обязательное) Листинг ML-модуля.....	113
	Приложение Г (обязательное) Листинг серверной части приложения.....	115

Введение

В современном мире интернет-торговля стала неотъемлемой частью жизни. Сейчас почти каждый человек делает покупки онлайн, начиная от продуктов питания и заканчивая электроникой. Из-за этого конкуренция между интернет-магазинами постоянно растет, и владельцам бизнеса нужно искать новые способы, как удержать клиента и увеличить продажи.

Одной из самых эффективных технологий для этого стали системы рекомендаций. Мы часто видим их на крупных маркетплейсах в блоках «С этим товаром покупают». Такие системы помогают покупателю быстрее найти нужную вещь, а магазину-значительно увеличить средний чек за счет сопутствующих товаров.

В России большинство торговых предприятий используют для учета платформу «1С: Предприятие». В базе 1С накапливаются огромные объемы данных о продажах, но стандартных инструментов для их интеллектуального анализа часто не хватает. Поэтому актуальной задачей является создание системы, которая могла бы забирать данные из 1С и обрабатывать с помощью современных алгоритмов машинного обучения на языке Python.

Объектом работы является: Процесс предложения рекомендуемых товаров в интернет-магазине и информационные технологии для его автоматизации.

Предмет работы: Алгоритмы машинного обучения и программные инструменты интеграции системы «1С:Предприятие» с веб-витриной для создания товарных рекомендаций.

Цель данной работы: разработка программного решения для автоматического подбора сопутствующих товаров на сайте, основанного на истории продаж из базы 1С и методах машинного обучения.

Для достижения цели были определены следующие задачи:

- изучить современные методы создания рекомендательных систем и провести обзор программных средств для автоматизации торговой деятельности;

- выявить и проанализировать функциональные и нефункциональные требования к проектируемому программному комплексу;
- разработать общую архитектуру системы, включая логику обмена данными между учетной базой и аналитическим модулем;
- реализовать программное решение для автоматического формирования рекомендаций и их отображения в интерфейсе пользователя;
- провести тестирование разработанной системы;
- оформить пояснительную записку в соответствии с установленными стандартами.

Разработанная система для автоматизации процесса товарных рекомендаций позволит упростить поиск нужных товаров для покупателей, обеспечить автоматический подбор сопутствующих позиций и создать удобную среду для совершения покупок в интернет-магазине. Внедрение системы упростит покупателям выбор товаров и увеличит средний чек за счёт сопутствующих рекомендаций.

1 Анализ предметной области и постановка задачи

1.1 Постановка задачи

Интернет-торговля сегодня быстро меняется. Покупатели активно оформляют заказы онлайн, и магазины вынуждены постоянно улучшать сервис. При большом ассортименте ключевая задача – не просто держать товар в наличии, а быстро и удобно помогать покупателю находить нужные позиции.

Многие малые и средние предприятия используют для учета товаров и продаж платформу «1С:Предприятие». Однако зачастую данные в 1С и витрина интернет-магазина существуют отдельно друг от друга. Старые методы подбора рекомендованных товаров, когда менеджер вручную связывает позиции в карточках на сайте, имеют существенные недостатки. Это требует огромных временных затрат и не позволяет оперативно реагировать на изменение спроса. Кроме того, ручной подход не учитывает реальную историю покупок, которая уже накоплена в учетной системе.

На рынке существуют и готовые автоматические платформы для рекомендаций, но большинство из них работают по облачному принципу. Это создает для бизнеса дополнительные сложности: необходимость ежемесячной оплаты дорогой подписки и риски, связанные с передачей данных о своих продажах и клиентах на сторонние серверы. Также такие системы часто не могут обеспечить мгновенное обновление информации об остатках и ценах напрямую из базы 1С.

Разработка системы для автоматического формирования рекомендаций на основе реальных данных из 1С - это важная задача, решение которой поможет вывести продажи на новый уровень. Такая система должна стать связующим звеном, основными элементами которой являются:

- товары, которые хранятся в справочнике 1С с актуальными ценами и остатками;
- история покупок, зафиксированная в документах реализации. Она служит базой для анализа;

- алгоритмы анализа, которые находят скрытые связи между товарами («с этим часто покупают»);
- веб-интерфейс, где покупатель видит готовый результат в удобном и наглядном виде.

Программное средство должно иметь интуитивно понятный интерфейс, корректно отображающий как основной каталог, так и блок рекомендаций в карточке товара.

Важно, чтобы система работала автоматически. Информация о ценах и остатках должна быть актуальной, а блок рекомендаций должен подстраиваться под историю продаж без участия программиста.

Главная цель разработки – автоматизировать процесс подбора сопутствующих товаров. Это позволит магазину увеличить средний чек и лояльность клиентов, так как они будут получать действительно полезные предложения, а не просто случайный список товаров.

1.2 Анализ информационных систем аналогичного назначения

Для решения задачи автоматизации товарных рекомендаций сегодня существует несколько подходов. Рассмотрим наиболее популярные решения, их достоинства и ограничения.

Одним из примеров являются стандартные модули системы «1С-Битрикс». «1С-Битрикс: Управление сайтом» – это самая распространенная в России коммерческая CMS-система для создания интернет-магазинов. В её дистрибутив входит стандартный компонент `catalog.recommended.products`, предназначенный для вывода блока рекомендуемых товаров в карточке [1].

Плюсы встроенного модуля Битрикса – он полностью совместим с платформой, работает надёжно, не требует сложной настройки и лишних платежей. Также администратор имеет полный контроль над списком товаров, которые увидит покупатель.

Однако существенным минусом является принцип формирования этих рекомендаций. В стандартной версии Битрикса подбор сопутствующих товаров

осуществляется через ручную привязку в свойстве `PROPERTY_LINK`. Это означает, что менеджер магазина должен самостоятельно зайти в каждую карточку товара и вручную выбрать из каталога подходящие дополнения. При ассортименте в несколько тысяч позиций такая работа становится практически невыполнимой. Таким образом, отсутствие автоматизации и анализа реальных продаж является ключевым ограничением стандартных средств CMS. К тому же, настроенные вручную рекомендации не меняются сами по себе. Если интересы покупателей изменятся или наступит другой сезон, сотрудникам придется снова заходить в каталог и вручную обновлять все связи между товарами, чтобы они оставались актуальными

Также нужно учитывать сложность интеграции этого модуля с учетной системой. Хотя «С-Битрикс» и «С:Предприятие» выпускаются под общим брендом, их стандартная синхронизация настроена только на передачу названий товаров и цен. Сам блок рекомендаций работает только внутри сайта. Чтобы он начал показывать результаты сложного анализа из базы 1С, программисту придется полностью переписывать код обмена данными. Это является трудоемкой и дорогой задачей

Другим популярным подходом является использование облачной платформы Retail Rocket.

Retail Rocket – это один из крупнейших облачных сервисов (SaaS), специализирующийся на анализе поведения пользователей и генерации умных рекомендаций для крупного ритейла. Сервис использует сложные математические модели для построения персональных предложений на сайте и в email-рассылках [2]. Платформа анализирует поведение каждого гостя на сайте и выстраивает индивидуальную модель его предпочтений. Такой персональный подход позволяет предлагать покупателю именно те позиции, которые его интересуют в данный момент.

Основным преимуществом системы является высокое качество алгоритмов машинного обучения, которые работают полностью автоматически. Система сама «запоминает» предпочтения каждого клиента, основываясь на его кликах,

просмотрах и покупках, что позволяет выдавать очень точные советы покупателю без участия менеджера.

К недостаткам Retail Rocket стоит отнести высокую стоимость подписки, которая зачастую недоступна для малого и среднего бизнеса. Кроме того, работа платформы организована как облачный сервис (SaaS). Это означает, что для формирования рекомендаций магазин обязан передавать информацию о своих продажах и клиентах на сторонние серверы. Для многих организаций это является проблемой с точки зрения безопасности данных. Таким образом, отсутствие возможности локального развертывания и необходимость передачи данных третьим лицам являются основными ограничениями для широкого использования данного решения.

Интеграция с базой 1С также является слабым местом этой платформы. Так как Retail Rocket – это внешнее облако, для его работы нужно постоянно выгружать из 1С огромные файлы с историей продаж. Настроить этот обмен так, чтобы сайт видел актуальные остатки со склада в реальном времени, очень сложно. Часто данные приходят с задержкой, и система может рекомендовать товары, которых уже нет в наличии.

Таким образом, каждое из рассмотренных решений имеет свои плюсы, но не всегда подходит для среднего бизнеса.

Сравнительный анализ рассмотренных систем и разрабатываемой подсистемы представлен в таблице 1.

Таблица 1 – Сравнительный анализ программных продуктов

Критерий	«1С-Битрикс»	«Retail Rocket»	Разрабатываемая система
Метод подбора товаров	Ручная привязка менеджером	Машинное обучение	Машинное обучение
Уровень автоматизации	Низкий	Высокий	Высокий
Доступность (цена)	Низкая	Высокая	Низкая

Продолжение таблицы 1

Критерий	«1С-Битрикс»	«Retail Rocket»	Разрабатываемая система
Сложность интеграции с базой 1С	Высокая	Высокая	Низкая
Безопасность данных	Высокая	Низкая	Высокая

Проведенный сравнительный анализ показал, что ни одно из существующих решений не является идеальным для малого и среднего бизнеса. Либо это требует огромных ручных трудозатрат (как в Битриксе), либо связано с большими финансовыми расходами и риском утечки данных (как в Retail Rocket).

Разрабатываемая система призвана избежать недостатков рассмотренных решений, предлагая автоматический, локальный и недорогой инструмент для получения товарных рекомендаций. Использование 1С как надежного источника данных, Python как инструмента машинного обучения и Java для веб-интерфейса позволяет получить работающее локально решение без дорогостоящих подписок. Это подтверждает актуальность разработки.

1.3 Выбор технологий проектирования системы

Проектирование архитектуры любого программного продукта начинается с создания визуальных моделей. Они позволяют структурировать логику работы системы. Чтобы система из нескольких модулей работала слаженно, нужна наглядная схема, описывающая взаимодействие всех компонентов. Для этих целей применяется графический язык UML. Данный язык является универсальным средством моделирования, которое используется для визуального описания внутренней структуры и алгоритмов поведения сложных систем [3].

В процессе выбора инструмента для создания схем было рассмотрено несколько популярных программ, описанные ниже.

Draw.io (diagrams.net). Удобный и простой онлайн-сервис для рисования различных графиков. Он удобен для создания простых схем и не требует установки на компьютер. Но так как это графический редактор общего назначения, в нем нет встроенной проверки правил UML. Из-за этого в схемах легко допустить технические ошибки, которые сложно заметить в процессе проектирования.

Enterprise Architect. Это очень мощная профессиональная среда, которую используют в огромных корпорациях. В ней можно спроектировать систему любой сложности. Программа обладает избыточным функционалом для рамок дипломного проекта, ее довольно сложно освоить. Также она требует приобретения коммерческой лицензии.

StarUML. Современная и быстрая программа. Она поддерживает все актуальные типы диаграмм, но является коммерческим продуктом. Возможности бесплатной версии сильно ограничены

В итоге в качестве основного средства проектирования я выбрала Visual Paradigm (Community Edition). Этот инструмент стал «золотой серединой» по следующим причинам:

- удобство работы за счет функции «умных» подсказок и быстрого создания аккуратных схем;
- поддержка всех необходимых нотаций, позволяющая совмещать BPMN-схемы и UML-диаграммы в одном проекте;
- доступность бесплатной версии для образовательных целей при сохранении широкого функционала.

Выбранный инструмент позволяет детально проработать диаграммы вариантов использования, диаграммы последовательностей и других, что необходимо для качественного описания взаимодействия модулей системы. Таким образом, использование Visual Paradigm позволяет создать хорошую техническую документацию, которая соответствует всем стандартам проектирования информационных систем.

1.4 Выбор инструментальных средств

Разработка современной системы рекомендаций требует объединить несколько различных технологий. Чтобы определиться с выбором инструментов для реализации проекта, необходимо было учесть следующие факторы:

- возможность работы с большими объемами данных о продажах;
- наличие мощных и проверенных библиотек для реализации алгоритмов машинного обучения;
- высокая скорость работы и отклика веб-интерфейса;
- надежность хранения учетной информации о товарах и складе;
- удобство интеграции различных программных модулей между собой.

Учитывая эти критерии, было принято решение создать программный комплекс на основе трех ключевых технологий: платформа «1С:Предприятие 8.3», язык программирования Python (библиотеки Pandas и Scikit-learn, SciPy) и веб-интерфейс на языке Java (фреймворк Spring Boot).

Платформа «1С:Предприятие 8.3» выбрана в качестве основного хранилища учетной информации. В этой системе будет сосредоточена вся первичная база данных предприятия: сведения о товарах, ценах, складских остатках и история совершенных продаж. 1С выбрана потому, что это стандарт автоматизации бизнес-процессов в России. Она обладает развитой средой разработки («Конфигуратор»). Это позволяет проектировать сложные структуры данных и настраивать механизмы обмена информацией с внешними системами.

Для реализации интеллектуального модуля системы выбран язык Python. Сейчас Python – один из главных инструментов для анализа данных, преимуществом которого является наличие огромного количества специализированных библиотек [4].

Например, Pandas позволяет очень эффективно работать со структурами данных: очищать выгрузку из 1С от ошибок и готовить её к математической обработке. С её помощью данные превращаются в удобные таблицы, ускоряя расчеты.

Библиотека Scikit-learn считается одним из лучших инструментов в области машинного обучения. Она содержит в себе множество проверенных алгоритмов для построения моделей и анализа [5].

SciPy – библиотека для вычислений. Она широко применяется в задачах обработки и анализа больших объёмов данных.

В качестве технологии для создания клиентской части был выбран язык Java и фреймворк Spring Boot [6]. Java хорошо подходит для разработки серверной части: она стабильна и справляется с большим потоком запросов. Фреймворк Spring Boot даёт возможность быстро собрать современный веб-интерфейс с учётом требований безопасности. Spring Boot уже включает встроенный сервер Tomcat, поэтому готовое приложение запускается само по себе. Не требуется устанавливать и настраивать отдельный веб-сервер. Такая связка позволяет витрине магазина работать быстро и без задержек при обмене данными с внешними системами.

Для того чтобы учётная система 1С общалась с сайтом, предполагается использовать веб-сервер Apache. Он позволит опубликовать программные интерфейсы 1С в сети. Это сделает базу данных доступной для взаимодействия с другими частями системы.

В итоге выбрана гибкая и производительная конфигурация. Учёт продаж и склада ведёт надёжная 1С, интеллектуальный анализ берёт на себя Python, а пользователь работает с быстрым и стабильным интерфейсом на Java.

1.5 Анализ функциональных требований к системе

Разрабатываемый комплекс обязан не только вести учёт товаров и продаж, но и на основе этих данных автоматически находить рекомендации, чтобы помогать магазину увеличивать прибыль. Система должна быть простой в использовании как для менеджера, который работает в 1С, так и для обычного покупателя, который заходит на сайт. Важно, чтобы предлагаемые рекомендации были точными и действительно помогали покупателям быстрее найти нужную вещь.

Для того чтобы система работала эффективно, были определены основные функции для каждой роли. Для наглядного представления функциональных требований к системе используется диаграмма вариантов использования.

Диаграмма вариантов использования (Use Case Diagram) – это один из основных типов диаграмм языка UML, который используется для описания функциональных требований к системе [7]. Она показывает, какие действия могут выполнять пользователи при взаимодействии с программой. Диаграмма вариантов использования представлена на рисунке 1.

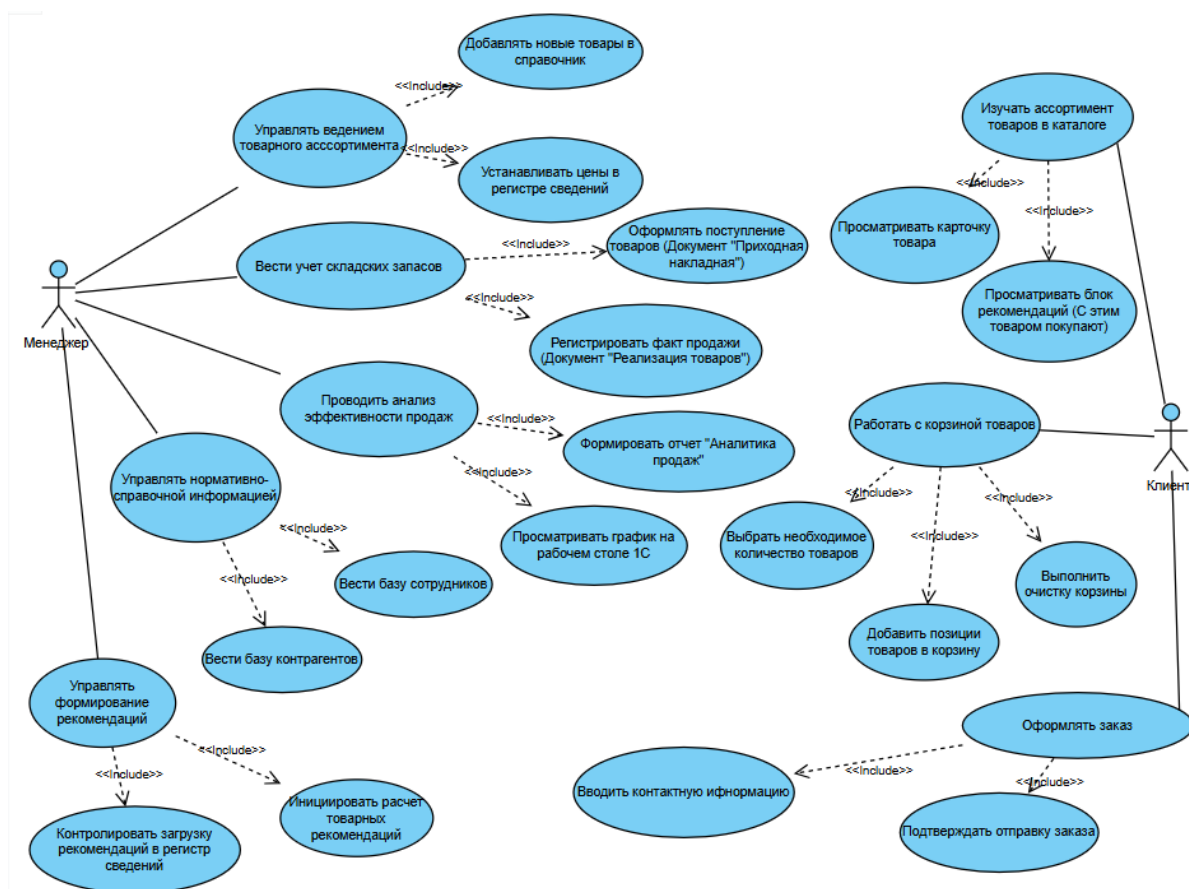


Рисунок 1 – Диаграмма вариантов использования

Менеджер является основным пользователем системы со стороны предприятия, обеспечивающим ввод данных и контроль работы аналитических модулей в среде «1С:Предприятие». В соответствии с диаграммой вариантов использования, менеджер выполняет следующие функции:

- а) управлять ведением товарного ассортимента:
 - 1) добавить новые товары в справочник;

- 2) устанавливать цены в регистре сведений;
- b) вести учет складских запасов:
 - 1) оформить поступление товара (документ «Приходная накладная»);
 - 2) зарегистрировать факт продажи (документ «Реализация товаров»);
- c) провести анализ эффективности продаж:
 - 1) просмотреть график на рабочем столе 1С;
 - 2) сформировать отчет «Аналитика продаж»;
- d) управлять нормативно-справочной информацией:
 - 1) вести базу контрагентов;
 - 2) вести базу сотрудников;
- e) управлять формированием рекомендаций:
 - 1) инициировать расчет товарных рекомендаций: запуск процесса интеллектуального анализа на основе истории продаж;
 - 2) контролировать загрузку рекомендаций в регистр сведений: проверка корректности импорта результатов анализа обратно в базу 1С.

Клиент является внешним пользователем системы, взаимодействующим с интерфейсом интернет-магазина. Согласно диаграмме, функции клиента разделены на три сценария:

- a) изучение ассортимента в каталоге:
 - 1) просматривать детальную карточку товара;
 - 2) изучать блок товарных рекомендаций;
- b) работать с корзиной товаров:
 - 3) выбирать необходимое количество товара;
 - 4) добавлять выбранные позиции в корзину;
 - 5) выполнять полную очистку корзины;
- c) оформлять заказ:
 - 6) вводить контактную информацию (имя, телефон);
 - 7) подтверждать отправку заказа.

Для детального описания ключевых функций системы ниже представлены спецификации основных прецедентов.

Спецификация прецедента «Просматривать товарные рекомендации» представлена в таблице 2.

Таблица 2 – Спецификация прецедента «Просматривать товарные рекомендации»

Прецедент: Просматривать товарные рекомендации
ID: 1
Краткое описание: Клиент получает рекомендации при просмотре товара
Главные актеры: Клиент
Предусловие: Покупатель перешел на страницу конкретного товара в каталоге
Основной поток: 1. Система определяет идентификатор просматриваемого товара.; 2. Система запрашивает связанные товары из базы данных.; 3. Система отображает блок «С этим товаром покупают» в интерфейсе; 4. Покупатель изучает предложенный ассортимент. Альтернативные потоки: 1.а. Для данного товара рекомендации отсутствуют. Система скрывает блок рекомендаций.
Постусловия: Покупателю предоставлена информация для совершения дополнительных покупок.

Спецификация прецедента «Инициировать расчет товарных рекомендаций» представлена в таблице 3.

Таблица 3 – Спецификация прецедента «Инициировать расчет товарных рекомендаций»

Прецедент: Инициировать расчет товарных рекомендаций
ID: 2
Краткое описание: Менеджер запускает процесс обновления базы рекомендаций на основе актуальной истории продаж
Главные актеры: Менеджер
Предусловие: Менеджер авторизован в системе 1С, открыта форма обработки «Интеллектуальный анализ»
Основной поток: 1. Менеджер нажимает кнопку запуска расчета.; 2. Система собирает данные о продажах за последние 6 месяцев.; 3. Система выполняет интеллектуальный расчет ассоциативных связей (машинное обучение).; 4. Система сохраняет полученные результаты в регистр сведений.; 5. Система выводит сообщение об успешном окончании расчета. Альтернативные потоки: 2.а. Обнаружена ошибка в данных или работа аналитического модуля прервана. Система выводит оповещение об ошибке.
Постусловия: В системе 1С обновлены данные о сопутствующих товарах.

Таким образом, я определила все основные функции, которые должна выполнять программа. Подробные описания (спецификации) в таблицах помогли пошагово разобрать, что делает пользователь и как на это должна реагировать система в каждом конкретном случае.

1.6 Анализ нефункциональных требований к системе

Для того чтобы программный комплекс был надежным и эффективным, я сформулировала ряд нефункциональных требований [8]. Это условия, которым должна соответствовать система для стабильной работы.

Производительность и время отклика. Основным требованием является высокая скорость работы пользовательского интерфейса. Страницы каталога и карточки товаров должны загружаться без видимых задержек. Так как математические расчеты в модуле Python являются ресурсозатратными, они должны выполняться в фоновом режиме. Это позволит сайту мгновенно выдавать уже готовые результаты из базы данных, не дожидаясь завершения работы алгоритма.

Удобство использования (Usability). Веб-интерфейс должен быть максимально простым и интуитивно понятным, чтобы покупатель мог совершить покупку без предварительного обучения. Важным требованием к дизайну является расположение блока «С этим товаром покупают» на видном месте в карточке товара, чтобы он сразу привлекал внимание пользователя. Для сотрудников магазина работа в среде «1С:Предприятие» должна строиться на базе стандартных форм и привычных механизмов управления.

Безопасность и конфиденциальность данных. Система должна обеспечивать сохранность информации о продажах. Ключевым требованием к безопасности является локальное хранение и обработка данных. Все вычисления модуля и база 1С должны находиться внутри инфраструктуры предприятия. Использование внешних облачных сервисов для анализа данных не допускается, чтобы исключить риск утечки коммерческой информации о клиентах.

Надежность. Система должна сохранять работоспособность основных функций (просмотр товаров, оформление заказа) даже в случае временной недоступности аналитического модуля Python. В такой ситуации интерфейс сайта должен просто скрывать блок рекомендаций, не блокируя работу всего магазина.

Масштабируемость. Система должна сохранять работоспособность и приемлемую производительность при увеличении объёмов данных (росте ассортимента товаров и накоплении истории продаж). Увеличение нагрузки не должно приводить к отказам в работе сайта или к невозможности расчёта рекомендаций. Аналитический модуль на Python должен продолжать корректно выполнять вычисления даже при значительном росте числа транзакций.

Техническая совместимость и стек технологий. Разрабатываемая система должна корректно работать в последних версиях браузеров (Google Chrome, Яндекс.Браузер, Safari).

К реализации системы предъявляются следующие технологические требования:

- использование платформы «1С:Предприятие 8.3» для хранения данных и учета;
- применение языка Python (библиотеки Pandas, Scikit-learn, SciPy) для аналитического модуля;
- реализация серверной части сайта на Java (Spring Boot) со встроенным сервером Tomcat;
- организация обмена данными между модулями через стандартные форматы CSV (между 1С и аналитическим модулем на Python) и JSON (между 1С и веб-сайтом на Java).

Выбранный технологический стек и соблюдение нефункциональных требований обеспечивают создание масштабируемой системы, способной эффективно решать задачи товарных рекомендаций в рамках единого информационного пространства предприятия.

1.7 Планирование разработки и оценка бюджета

Прежде чем приступить к созданию программы, нужно решить, в каком порядке будут выполняться работы и сколько это будет стоить. Обычно выбирают либо гибкую методику (Agile), либо жёсткую последовательную (Waterfall) [9]. В данной работе использовалась последовательная модель: сначала полностью заканчивается один этап, затем следующий.

Для ведения графика работ и учёта финансов была применена программа Project Libre. ProjectLibre – то бесплатная программа для планирования проектов, управления задачами и расчёта бюджета.

Сначала в системе был создан проект с датой старта 8 апреля 2026 года. Затем в него добавлены все запланированные задачи, указана их длительность, а также связи между ними (какие задачи должны закончиться раньше, чем начнутся другие).

Всего выделено 4 крупных этапа и 20 подзадач. Общая длительность разработки – 45 рабочих дней (с апреля по июнь 2026 года). Рабочая неделя – пятидневная. Этапы разработки представлены на рисунке 2.

Название	Продолжительность	Начало	Окончание	Предшествующие
Начало работы	0 дней	08.04.2026, 08:00	08.04.2026, 08:00	
☒ Анализ предметной области и постановка задачи	6,5 дней	08.04.2026, 08:00	16.04.2026, 13:00	
Постановка задачи	1 день	08.04.2026, 08:00	08.04.2026, 17:00	1
Анализ информационных систем аналогичных	2 дней	09.04.2026, 08:00	10.04.2026, 17:00	3
Выбор технологий проектирования системы	1 день	13.04.2026, 08:00	13.04.2026, 17:00	4
Выбор инструментальных средств	0,5 дней	14.04.2026, 08:00	14.04.2026, 13:00	5
Анализ функциональных требований к системе	1 день	14.04.2026, 13:00	15.04.2026, 13:00	6
Анализ нефункциональных требований к системе	1 день	15.04.2026, 13:00	16.04.2026, 13:00	7
☒ Проектирование информационной системы	13,5 дней	16.04.2026, 13:00	05.05.2026, 17:00	
Функциональное моделирование процессов	1,5 дней	16.04.2026, 13:00	17.04.2026, 17:00	8
Проектирование структуры хранения данных	2 дней	20.04.2026, 08:00	21.04.2026, 17:00	10
Проектирование логики взаимодействия подсистем	2 дней	22.04.2026, 08:00	23.04.2026, 17:00	11
Проектирование сценариев обмена данными	2 дней	24.04.2026, 08:00	27.04.2026, 17:00	12
Проектирование архитектуры развертывания	2 дней	28.04.2026, 08:00	29.04.2026, 17:00	13
Проектирование ML-алгоритма товарных рекомендаций	2 дней	30.04.2026, 08:00	01.05.2026, 17:00	14
Проектирование пользовательского интерфейса	2 дней	04.05.2026, 08:00	05.05.2026, 17:00	15
☒ Реализация системы	19 дней	06.05.2026, 08:00	01.06.2026, 17:00	
Разработка структуры объектов информационной системы	3 дней	06.05.2026, 08:00	08.05.2026, 17:00	16
Реализация программных средств для системы	10 дней	11.05.2026, 08:00	22.05.2026, 17:00	18
Реализация пользовательского интерфейса	6 дней	25.05.2026, 08:00	01.06.2026, 17:00	19
☒ Контроль качества системы рекомендаций	6 дней	02.06.2026, 08:00	09.06.2026, 17:00	
Определение режима тестирования и метрики	0,5 дней	02.06.2026, 08:00	02.06.2026, 13:00	20
Функциональное тестирование	2 дней	02.06.2026, 13:00	04.06.2026, 13:00	22
Нефункциональное тестирование	2 дней	04.06.2026, 13:00	08.06.2026, 13:00	23
Метрики кода	1,5 дней	08.06.2026, 13:00	09.06.2026, 17:00	24
Конец работы	0 дней	09.06.2026, 17:00	09.06.2026, 17:00	25

Рисунок 2 – Этапы разработки системы

Для каждой задачи назначены исполнители – аналитик, разработчик, дизайнер, тестировщик. Для каждого специалиста задана часовая ставка:

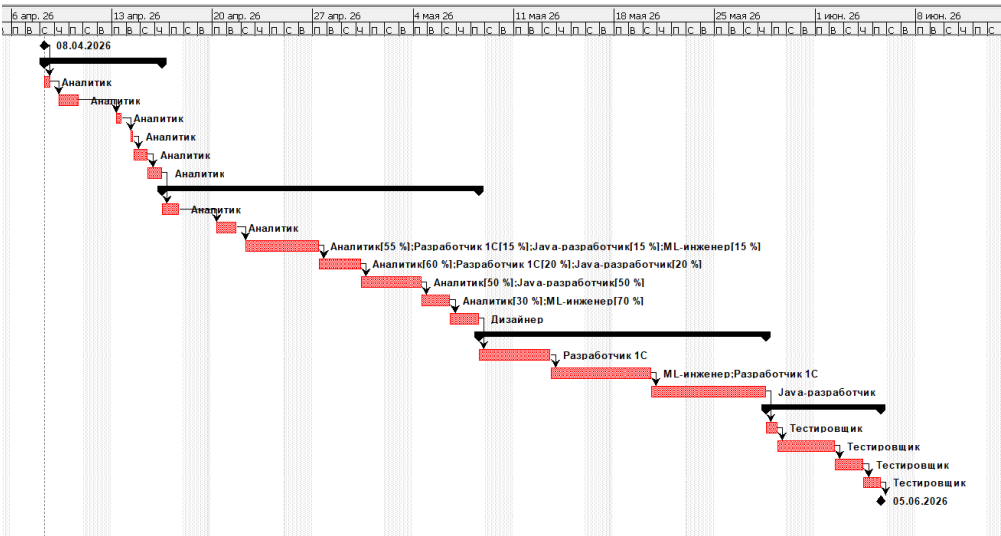
- аналитик: 400 руб./час;
- разработчик: 500 руб./час;
- тестировщик: 350 руб./час;
- дизайнер: 300 руб./час.

Перечень задействованных специалистов и их часовая ставка приведены на рисунке 3.

Название	Тип	Стандартная ставка	Начислять	Основной календарь
Аналитик	Работа	400/час	Пропорционально	Пятидневка
Разработчик 1С	Работа	500/час	Пропорционально	Пятидневка
Тестировщик	Работа	350/час	Пропорционально	Пятидневка
Дизайнер	Работа	300/час	Пропорционально	Пятидневка
Java-разработчик	Работа	500/час	Пропорционально	Пятидневка
ML-инженер	Работа	600/час	Пропорционально	Пятидневка

Рисунок 3 – Исполнители и стоимость часа работы

Визуальное представление сроков выполнения задач дано на диаграмме Ганта – рисунок 4. Она наглядно показывает последовательность этапов, длительность каждой задачи, а также временные интервалы, в которые задействованы разные исполнители (аналитик, разработчик, дизайнер, тестировщик).



Стоимость каждой задачи вычисляется как сумма затрат по всем участвующим в ней специалистам (произведение отработанных часов на часовую ставку). Общая стоимость разработки составила 154 320 рублей. Детализация затрат приведена на рисунке 5.

№	Название	Продолжительность	Начало	Окончание	Предшествующие	Стоимость	Название ресурса
	Начало работы	0 дней	08.04.2026, 08:00	08.04.2026, 08:00		0,00 Р	
	Анализ предметной области и постановка задачи	6,5 дней	08.04.2026, 08:00	16.04.2026, 13:00		20800,00 Р	
	Постановка задачи	1 день	08.04.2026, 08:00	08.04.2026, 17:00	1	3200,00 Р	Аналитик
	Анализ информационных систем аналогичных	2 дней	09.04.2026, 08:00	10.04.2026, 17:00	3	6400,00 Р	Аналитик
	Выбор технологий проектирования систем	1 день	13.04.2026, 08:00	13.04.2026, 17:00	4	3200,00 Р	Аналитик
	Выбор инструментальных средств	0,5 дней	14.04.2026, 08:00	14.04.2026, 13:00	5	1600,00 Р	Аналитик
	Анализ функциональных требований к системе	1 день	14.04.2026, 13:00	15.04.2026, 13:00	6	3200,00 Р	Аналитик
	Анализ нефункциональных требований к системе	1 день	15.04.2026, 13:00	16.04.2026, 13:00	7	3200,00 Р	Аналитик
	Проектирование информационной системы	16,269 дней	16.04.2026, 13:00	08.05.2026, 15:09		46752,54 Р	
	Функциональное моделирование процессов	1,5 дней	16.04.2026, 13:00	17.04.2026, 17:00	8	4800,00 Р	Аналитик
	Проектирование структуры хранения данных	2 дней	20.04.2026, 08:00	21.04.2026, 17:00	10	6400,00 Р	Аналитик
	Проектирование логики взаимодействия компонентов	3,333 дней	22.04.2026, 08:00	27.04.2026, 10:39	11	7999,82 Р	Аналитик(55 %);Разработчик(45 %)
	Проектирование сценариев обмена данными	2,769 дней	27.04.2026, 10:39	30.04.2026, 08:49	12	6646,06 Р	Аналитик(60 %);Разработчик(40 %)
	Проектирование архитектуры развертывания	2,667 дней	30.04.2026, 08:49	04.05.2026, 15:09	13	7466,67 Р	Аналитик(50 %);Java-разработчик(50 %)
	Проектирование ML-алгоритма товарных рекомендаций	2 дня	04.05.2026, 15:09	06.05.2026, 15:09	14	8640,00 Р	Аналитик(30 %);ML-инженер(70 %)
	Проектирование пользовательского интерфейса	2 дня	06.05.2026, 15:09	08.05.2026, 15:09	15	4800,00 Р	Дизайнер
	Реализация системы	14 дней	08.05.2026, 15:09	28.05.2026, 15:09		80000,00 Р	
	Разработка структуры объектов информационной системы	3 дня	08.05.2026, 15:09	13.05.2026, 15:09	16	12000,00 Р	Разработчик 1С
	Реализация программных средств для системы	5 дней	13.05.2026, 15:09	20.05.2026, 15:09	18	44000,00 Р	ML-инженер;Разработчик 1С
	Реализация пользовательского интерфейса	6 дней	20.05.2026, 15:09	28.05.2026, 15:09	19	24000,00 Р	Java-разработчик
	Контроль качества системы рекомендаций	6 дней	28.05.2026, 15:09	05.06.2026, 15:09		16800,00 Р	
	Определение режима тестирования и метрик	0,5 дней	28.05.2026, 15:09	29.05.2026, 10:09	20	1400,00 Р	Тестировщик
	Функциональное тестирование	2 дня	29.05.2026, 10:09	02.06.2026, 10:09	22	5600,00 Р	Тестировщик
	Нефункциональное тестирование	2 дня	02.06.2026, 10:09	04.06.2026, 10:09	23	5600,00 Р	Тестировщик
	Метрики кода	1,5 дня	04.06.2026, 10:09	05.06.2026, 15:09	24	4200,00 Р	Тестировщик
	Конец работы	0 дней	05.06.2026, 15:09	05.06.2026, 15:09	25	0,00 Р	

Рисунок 5 – Детализация затрат на разработку

Таким образом, полученный бюджет разработки является обоснованным. Полностью соответствует запланированному объёму работ.

2 Проектирование информационной системы

2.1 Функциональное моделирование процессов системы

Одним из важнейших этапов проектирования является создание функциональной модели процесса. Для проектирования системы товарных рекомендаций необходимо построить модель, которая наглядно покажет путь превращения исходных данных о продажах в товарные рекомендации для покупателя.

Наиболее подходящей для этой задачи является методология функционального моделирования IDEF0. Она позволяет описать систему как совокупность взаимодействующих функций [10]. Основой модели является контекстная диаграмма, которая представляет систему как единый процесс, взаимодействующий с внешней средой.

Контекстная диаграмма процесса формирования и отображения товарных рекомендаций в интернет-магазине представлена на рисунке 6.

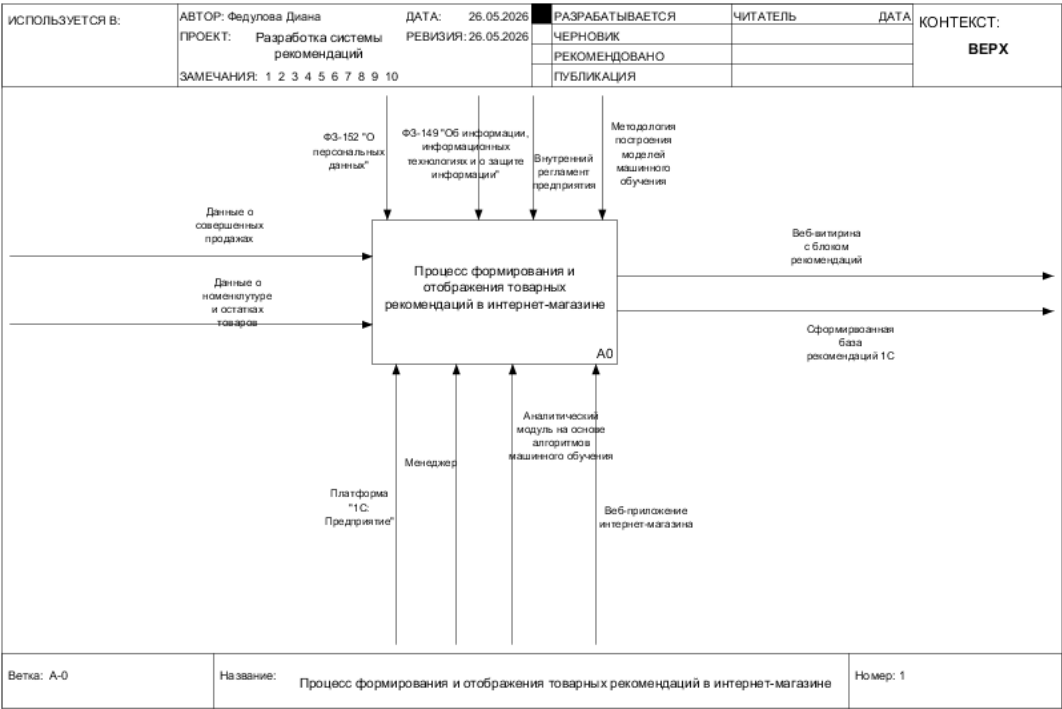


Рисунок 6 – Контекстная диаграмма IDEF0

В представленной контекстной диаграмме входными данными являются данные о номенклатуре и остатках товаров – актуальный список позиций из

справочника 1С, их цены и наличие на складе. Эти данные необходимы для того, чтобы система не рекомендовала отсутствующие товары.

А также данные о совершенных продажах – массив транзакций (чеков) из документов «Реализация товаров». Это первичная информация, на основе которой выявляются закономерности в поведении покупателей.

Выходными данными, отражающими результат работы системы, выступают веб-витрина с блоком рекомендаций для конечного пользователя и сформированная база рекомендаций в 1С для внутреннего учета.

Управление процессом осуществляется на основе следующих регламентов:

- фз-152 «О персональных данных» и ФЗ-149 «Об информации, информационных технологиях и о защите информации»: регулируют правила обработки и защиты информации о покупках клиентов;
- внутренний регламент предприятия (правила работы магазина): определяет периодичность актуализации данных (регламентные задания);
- методология построения моделей машинного обучения (математическая логика алгоритмов машинного обучения).

Ресурсами (механизмами) системы являются:

- менеджер;
- платформа «1С:Предприятие»;
- аналитический модуль машинного обучения на Python;
- Веб-приложение на Java-серверная часть, обеспечивающая визуализацию данных.

Для более глубокого понимания логики работы была проведена декомпозиция основного процесса. Декомпозиция представляет собой разделение общей сложной задачи на несколько более простых и понятных этапов. Это необходимо для того, чтобы увидеть все внутренние связи между программными модулями, которые скрыты на общей схеме. Такой подход позволяет пошагово проследить путь данных. Детализирующая диаграмма представлена на рисунке 7.

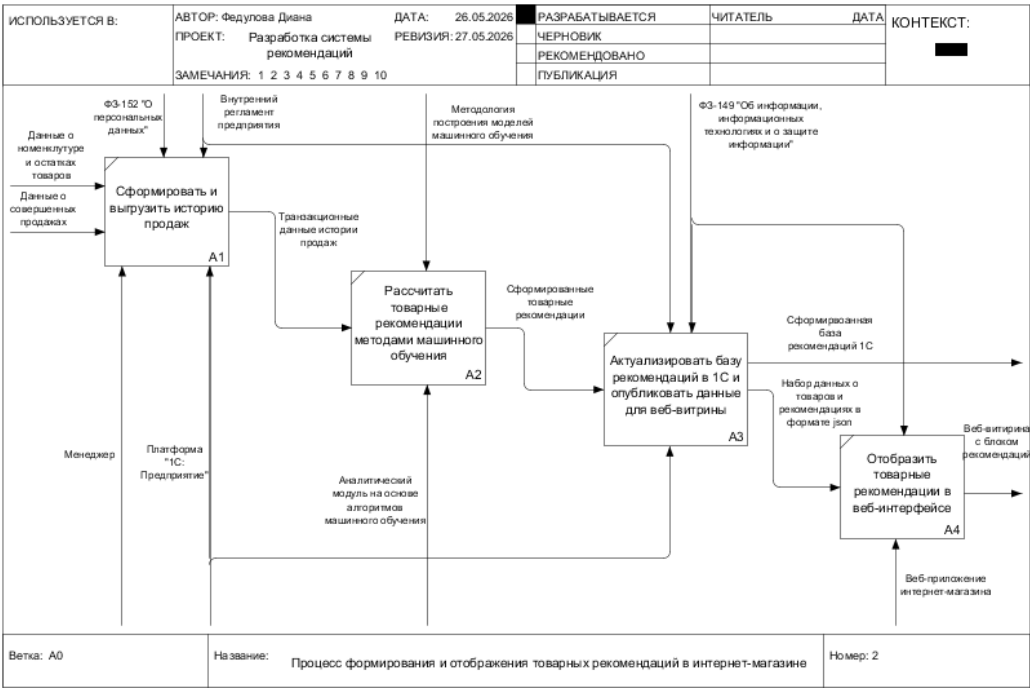


Рисунок 7 – Диаграмма декомпозиции IDEF0

В результате декомпозиции были выделены четыре ключевых этапа:

- сформировать и выгрузить историю продаж (A1): на этом этапе в среде 1С подготавливаются транзакционные данные за отчетный период и экспортируются во внешний формат CSV;
- рассчитать товарные рекомендации методами машинного обучения (A2): аналитический модуль Python загружает данные и строит модель ассоциативных связей, вычисляя коэффициенты сходства товаров;
- актуализировать базу рекомендаций в 1С и опубликовать данные для веб-витрины (A3): происходит импорт результатов анализа в учетную систему и подготовка JSON-пакетов для передачи на сайт через программный интерфейс (API);
- отобразить товарные рекомендации в веб-интерфейсе (A4): финальный этап визуализации, где сайт на Java получает данные и отрисовывает карточку товара с персональными предложениями.

Таким образом, построение модели в нотации IDEF0 позволило пошагово описать весь процесс работы: от сбора истории продаж в 1С до вывода готовых рекомендаций на страницы интернет-магазина.

2.2 Проектирование структуры хранения данных

После описания процессов системы в нотации IDEF0 необходимо определить структуру хранения информации. Для систем на базе машинного обучения правильная организация данных критична, так как от качества и связей этой информации напрямую зависит точность будущих рекомендаций.

Проектирование базы данных проводилось в среде Visual Paradigm Community Edition. Этот инструмент позволяет строить детальные диаграммы «сущность-связь» (ER-диаграммы), которые наглядно описывают структуру базы. Для описания информационной базы интернет-магазина была выбрана логическая модель данных [11]. Она показывает связь между реальными объектами (товарами, продажами) и их представлением в системе.

Поскольку система строится на платформе «1С:Предприятие 8.3», физические таблицы в SQL-базе создаются платформой автоматически на основе метаданных. Поэтому проектировать отдельную физическую модель и писать SQL-скрипты не требуется. Логической модели достаточно для настройки базы данных в конфигураторе 1С.

В итоге была разработана модель, охватывающая 13 ключевых элементов системы. В неё вошли справочники номенклатуры и контрагентов, документы для регистрации продаж и аналитические регистры, в которых хранятся результаты работы ML-модели. Полная логическая схема базы данных представлена на рисунке 8.

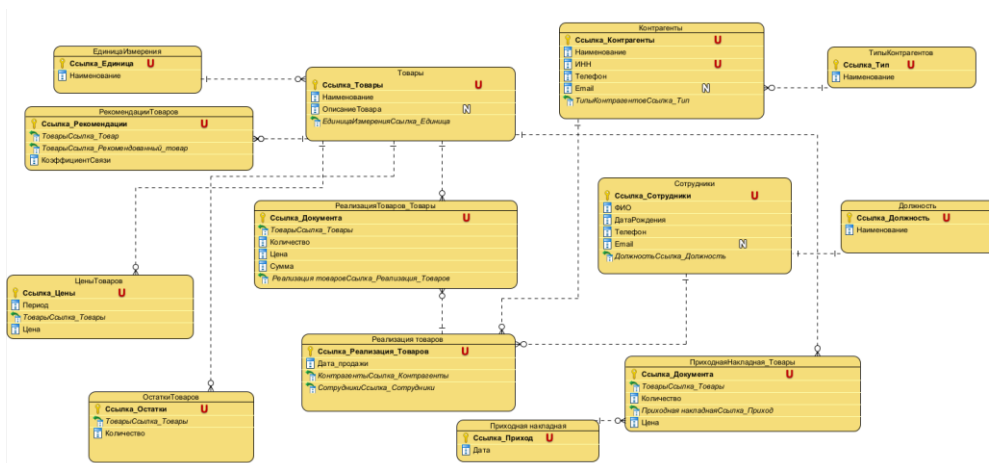


Рисунок 8 – Логическая модель системы

Описание атрибутов логической модели приведено в таблице 4, в которой представлены атрибуты сущности «Товары».

Таблица 4 – Описание атрибутов сущности «Товары»

Атрибуты	Нулевое значение	Ключевое значение
Ссылка_Товары	Не допустимо	РК
Наименование	Не допустимо	
ОписаниеТовара	Допустимо	
ЕдиницаИзмеренияСсылка_Единица	Не допустимо	FK

Для обеспечения корректного складского учета, автоматизации количественного контроля и исключения ошибок при оформлении первичной документации каждой товарной позиции необходимо задать соответствующие единицы измерения (штуки, литры, упаковки). Структура данной справочной информации представлена в таблице 5.

Таблица 5 – Описание атрибутов сущности «Единица измерения»

Атрибуты	Нулевое значение	Ключевое значение
Ссылка_Единица	Не допустимо	РК
Наименование	Не допустимо	

Ключевым аналитическим элементом разрабатываемой системы является сущность, хранящая результаты работы ML-алгоритма. В таблице 6 описаны атрибуты, необходимые для хранения связей между основным и рекомендованным товаром, а также численных коэффициентов сходства (весов), которые определяют приоритетность вывода товаров в интерфейсе магазина.

Таблица 6 – Описание атрибутов сущности «РекомендацииТоваров»

Атрибуты	Нулевое значение	Ключевое значение
Ссылка_Рекомендации	Не допустимо	РК
ТоварыСсылка_Товар	Не допустимо	FK
ТоварыСсылка_Рекомендованный_товар	Не допустимо	FK
КоэффициентСвязи	Не допустимо	-

Помимо связей между товарами, система должна учитывать актуальную стоимость позиций для расчета итоговой суммы чека. В таблице 7 приведен перечень полей, обеспечивающих хранение цен в разрезе периодов времени.

Таблица 7 – Описание атрибутов сущности «ЦеныТоваров»

Атрибуты	Нулевое значение	Ключевое значение
Ссылка_Цены	Не допустимо	РК
Период	Не допустимо	-
ТоварыСсылка_Товары	Не допустимо	FK
Цена	Не допустимо	-

Для того чтобы система не рекомендовала к покупке отсутствующие на складе позиции, необходимо фиксировать текущее количество товара. Такой сущностью, которая фиксирует остатки товаров на складе, является «ОстаткиТоваров». Атрибуты сущности «ОстаткиТоваров» приведен в таблице 8.

Таблица 8 – Описание атрибутов сущности «ОстаткиТоваров»

Атрибуты	Нулевое значение	Ключевое значение
Ссылка_Остатки	Не допустимо	РК
ТоварыСсылка_Товары	Не допустимо	FK
Количество	Не допустимо	-

Основным источником данных для обучения модели машинного обучения являются факты совершения продаж. Заголовочная часть сущности реализации, содержащая общие сведения о сделке, описана в таблице 9.

Таблица 9 – Описание атрибутов сущности «Реализация Товаров»

Атрибуты	Нулевое значение	Ключевое значение
Ссылка_Реализация_Товаров	Не допустимо	РК
Дата_продажи	Не допустимо	-
КонтрагентыСсылка_Контрагенты	Не допустимо	FK
СотрудникиСсылка_Сотрудники	Не допустимо	FK

При проектировании базы данных важно предусмотреть, как именно будут храниться детали каждой продажи. Для детального отражения состава каждой продажи (перечня товаров, их количества и цен в конкретном чеке) используется табличная часть сущности реализации товаров. В ней каждая строка – это отдельный товар из чека. Подробное описание её атрибутов содержится в таблице 10.

Таблица 10 – Описание атрибутов сущности «РекомендацииТоваров_Товары»

Атрибуты	Нулевое значение	Ключевое значение
Ссылка_Документа	Не допустимо	РК
ТоварыСсылка_Товары	Не допустимо	FK
Количество	Не допустимо	-
Цена	Не допустимо	-
Сумма	Не допустимо	-
Реализация товаровСсылка_Реализация_Товаров	Не допустимо	FK

Для того чтобы магазин мог вести учет своих клиентов и понимать, кто именно совершает покупки, в модель данных включена сущность «Контрагенты». Планируется хранить здесь не только названия компаний или имена покупателей, но и их контактные данные, например, номера телефонов и адреса электронной почты. В таблице 11 представлены характеристики сущности «Контрагенты».

Таблица 11 – Описание атрибутов сущности «Контрагенты»

Атрибуты	Нулевое значение	Ключевое значение
Ссылка_Контрагенты	Не допустимо	РК
Наименование	Не допустимо	-
ИНН	Не допустимо	-

Продолжение таблицы 11

Атрибуты	Нулевое значение	Ключевое значение
Телефон	Не допустимо	FK
Email	Допустимо	-
ТипыКонтрагентовСсылка_Тип	Не допустимо	FK

С целью логического разделения клиентской базы в системе выделена отдельная сущность для хранения типов контрагентов, параметры которой представлены в таблице 12.

Таблица 12 – Описание атрибутов сущности «ТипыКонтрагенты»

Атрибуты	Нулевое значение	Ключевое значение
Ссылка_Тип	Не допустимо	PK
Наименование	Не допустимо	-

Для учета ответственных за проведение операций продаж в системе фиксируются данные о персонале организации. Описание атрибутов сущности сотрудники приведен в таблице 13.

Таблица 13 – Описание атрибутов сущности «Сотрудники»

Атрибуты	Нулевое значение	Ключевое значение
Ссылка_Сотрудники	Не допустимо	PK
ФИО	Не допустимо	-

Продолжение таблицы 13

Атрибуты	Нулевое значение	Ключевое значение
ДатаРождения	Не допустимо	-
Телефон	Не допустимо	FK
Email	Допустимо	-
ДолжностьСсылка_Должность	Не допустимо	FK

Иерархическая структура предприятия и роли сотрудников определяются в сущности должностей, описание которой содержится в таблице 14.

Таблица 14 – Описание атрибутов сущности «Должность»

Атрибуты	Нулевое значение	Ключевое значение
Ссылка_Должность	Не допустимо	PK
Наименование	Не допустимо	-

Процесс пополнения товарных запасов и регистрации поступлений от поставщиков осуществляется с помощью сущности «Приходная накладная». Подробная структура сущности заголовка приходной накладной описана в таблице 15.

Таблица 15 – Описание атрибутов сущности «Приходная накладная»

Атрибуты	Нулевое значение	Ключевое значение
Ссылка_Приход	Не допустимо	PK

Продолжение таблицы 15

Атрибуты	Нулевое значение	Ключевое значение
Дата	Не допустимо	-

Спецификация товаров, поступивших на склад по конкретной накладной, фиксируется в связанной сущности, состав атрибутов которой представлен в завершающей таблице 16.

Таблица 16 – Описание атрибутов сущности «ПриходнаяНакладная_Товары»

Атрибуты	Нулевое значение	Ключевое значение
Ссылка_Документа	Не допустимо	РК
ТоварыСсылка_Товары	Не допустимо	FK
Количество	Не допустимо	-
Цена	Не допустимо	-
Приходная накладнаяСсылка_Приход	Не допустимо	FK

Разработанная логическая модель базы данных охватывает 13 ключевых сущностей, которые полностью описывают предметную область интернет-магазина. В модель включены справочники (Товары, Контрагенты, Сотрудники, Единица измерения), документы (РеализацияТоваров, Приходная накладная), а также специализированные регистры сведений (ЦеныТоваров, ОстаткиТоваров) и регистр РекомендацииТоваров, предназначенный для хранения результатов работы аналитического модуля (товарных рекомендаций).

Такая структура полностью покрывает потребности системы. Товары, продажи, цены, остатки и рекомендации связаны в единую схему, позволяющую строить качественные модели машинного обучения.

2.3 Проектирование логики взаимодействия программных модулей

Для того чтобы наглядно описать, как именно работает система и как в ней перемещаются данные, была разработана диаграмма деятельности (Activity Diagram). В языке UML этот инструмент используется для моделирования алгоритмов и логики поведения системы [12].

Диаграмма позволяет показать не просто отдельные функции, а всю цепочку действий в динамике: от автоматической выгрузки информации из базы 1С до её интеллектуальной обработки в модуле Python и итогового отображения рекомендаций на сайте. На схеме последовательно отражены все ключевые этапы, включая моменты принятия решений и разветвления процессов. Это позволяет заранее увидеть, как система будет реагировать на те или иные события, и исключить логические ошибки еще до начала написания кода. Каждое действие на диаграмме представлено в виде прямоугольника со скругленными углами, а переходы между ними обозначены направленными стрелками, что позволяет четко проследить жизненный цикл данных внутри программного комплекса.

Главной особенностью этой модели является то, что процесс разделен на три функциональные области – дорожки (swimlanes). Каждая дорожка соответствует своей технологии (1С, Python или Java). Это помогает четко разграничить ответственность между разными частями разрабатываемой системы. Благодаря такому разделению становится понятно, в какой момент управление переходит от одной программной среды к другой. Какие именно данные передаются в качестве входных параметров для следующего шага. Визуальное представление взаимодействия модулей делает архитектуру проекта прозрачной и упрощает дальнейшую техническую реализацию каждого компонента. Использование стандартной нотации UML делает описание алгоритма универсальным и понятным для любого разработчика, независимо от языка программирования. Спроектированная диаграмма деятельности представлена на рисунке 8.

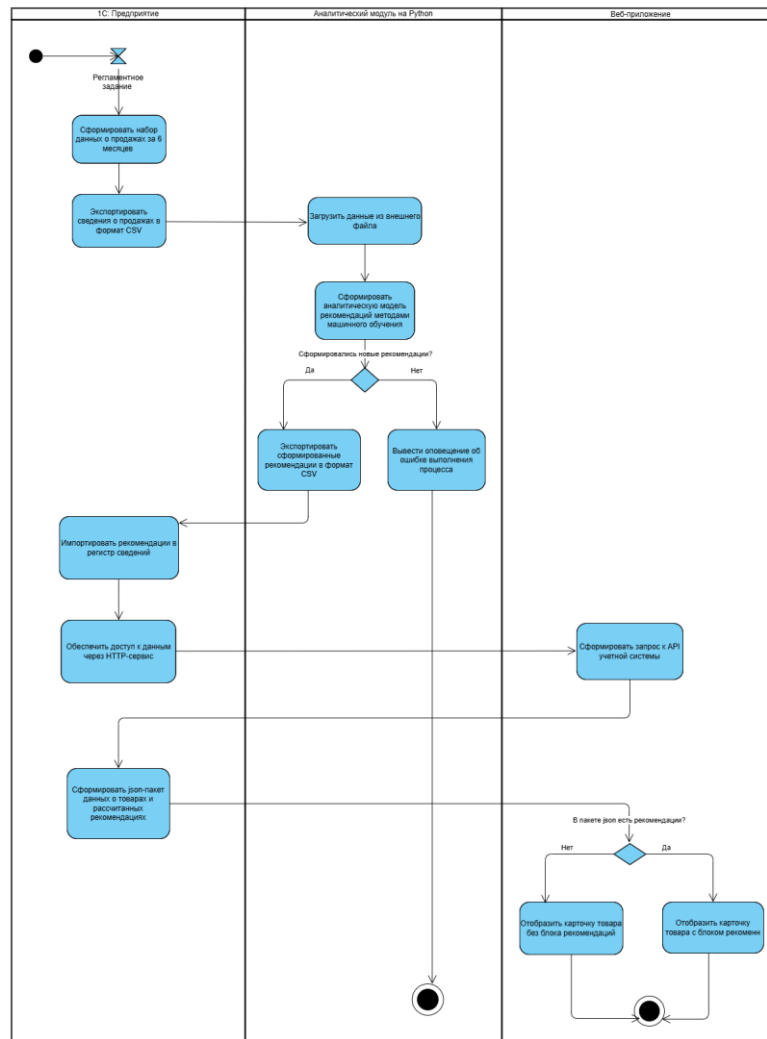


Рисунок 8 – Диаграмма деятельности

Логику работы системы, изображенную на диаграмме, можно разделить на несколько основных этапов.

Подготовка данных и расчет рекомендаций. Этот процесс происходит «в фоновом режиме» без участия покупателя. Всё начинается в дорожке «1С:Предприятие» с регламентного задания. Система сама собирает данные о продажах за последние 6 месяцев и выгружает их в CSV-файл с периодичностью в неделю. Затем управление переходит к «Аналитическому модулю» на Python. Скрипт загружает файл и строит математическую модель.

На схеме видно разветвление: если расчет прошел успешно и рекомендации сформированы, Python создает файл с результатами. Если же произошла ошибка (например, данных слишком мало), система выводит

оповещение об ошибке. Завершается этот этап в 1С, где новые данные записываются в регистр сведений.

Работа сайта и отображение результатов. Этот этап происходит в реальном времени, когда человек заходит в интернет-магазин. «Веб-приложение» на Java отправляет запрос к API учетной базы. 1С мгновенно обрабатывает этот запрос и формирует пакет данных в формате JSON, где собраны актуальные цены, остатки и рассчитанные ранее рекомендации.

На стороне сайта стоит еще один узел принятия решения. Программа проверяет: если в полученном пакете есть рекомендации для этого товара, то под основной карточкой отрисовывается блок «С этим товаром покупают». Если рекомендаций нет, блок рекомендаций скрыт.

Такое разделение процессов позволяет сайту работать очень быстро. Покупателю не нужно ждать, пока Python закончит сложные вычисления – он мгновенно получает уже готовый результат из базы 1С.

2.4 Проектирование сценариев обмена данными

Для того чтобы детально показать, как сайт на Java и учетная система 1С общаются между собой, была разработана диаграмма последовательности (Sequence Diagram). В языке UML этот инструмент используется для визуализации диалога между частями системы [13]. В рамках моей темы это позволяет пошагово проследить, как результаты машинного обучения передаются из базы данных 1С и превращаются в товарные рекомендации на экране пользователя.

Также она позволяет пошагово проследить, какие запросы отправляет сайт и какие ответы возвращает 1С в ответ на действия пользователя.

В моей работе это особенно важно, так как нужно было четко спроектировать взаимодействие двух разных программных сред через API. Это помогает заранее проверить всю логику обмена информацией и убедиться, что запросы к API будут выполняться без ошибок. Спроектированная диаграмма представлена на рисунке 9.

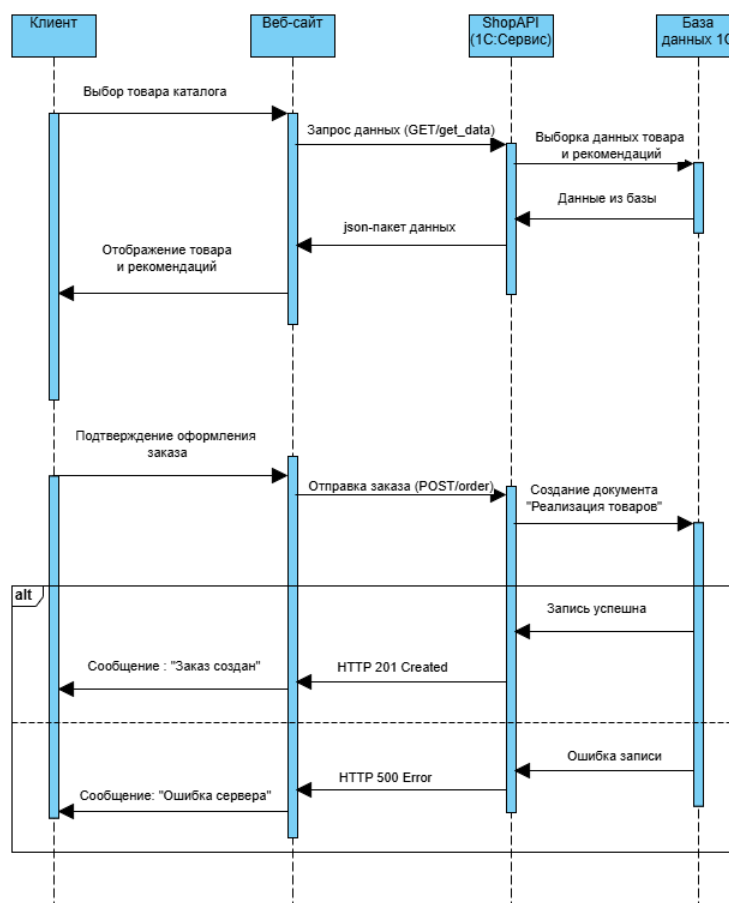


Рисунок 9 – Диаграмма последовательности

На представленной схеме выделены четыре основных объекта (линии жизни), которые взаимодействуют друг с другом для реализации функций системы.

Клиент – внешний актер (покупатель), который инициирует запросы к системе через интерфейс браузера.

Веб-сайт – серверное приложение на языке Java (Spring Boot), которое отвечает за логику отображения данных и формирование запросов к учетной системе. Его главная задача – получить технический ответ от учетной системы и превратить его в красивую и удобную страницу, которую увидит человек

ShopAPI (1C:Сервис) – промежуточный программный шлюз (HTTP-сервис), созданный внутри 1С. Он принимает запросы от сайта, проверяет их и «общается» с базой данных 1С. Использование такого шлюза обеспечивает безопасность, ведь сайт запрашивает те данные, которые разрешено видеть.

База данных 1С – уровень хранения информации, включающий в себя справочники товаров, регистры цен, регистры с рекомендациями, которые являются результатами машинного обучения.

На схеме детально описаны два ключевых сценария взаимодействия.

Получение данных витрины и рекомендаций. При обращении пользователя к странице товара веб-приложение отправляет GET-запрос к сервису ShopAPI. В этот момент ShopAPI (1С) выполняет выборку данных из регистров цен и остатков, а также извлекает блоки рекомендованных товаров, рассчитанные модулем машинного обучения. Итоговый массив данных упаковывается в пакет формата JSON и передается на сайт для отображения карточки товара и блока рекомендаций.

Оформление заказа. Когда покупатель подтверждает покупку, веб-сайт отправляет POST-запрос с контактными данными и составом корзины в 1С. Получив этот запрос, сервис ShopAPI обрабатывает данные и инициирует создание нового документа «Реализация товаров» в базе данных 1С.

Для обеспечения надежности процесса на диаграмме использован фрагмент alt (alternative), описывающий обработку результата записи: в случае успеха база данных 1С подтверждает создание документа, и сервис возвращает статус HTTP 201 Created.

В случае возникновения ошибки система возвращает статус HTTP 500 Error.

Таким образом, разработанная диаграмма последовательности наглядно показывает, как именно происходит обмен данными между сайтом и учетной базой в режиме реального времени. Спроектированная логика гарантирует, что покупатель всегда видит только актуальные рекомендации и цены. А использование блока альтернативы (alt) обеспечивает надежность системы при оформлении заказов. Такой подход к проектированию позволяет заранее продумать все нюансы общения между Java и 1С, чтобы в процессе реальной работы сайта не возникало задержек или потерь информации.

2.5 Проектирование архитектуры развертывания и логической структуры системы

Для определения физической структуры будущего программного комплекса и планирования размещения его компонентов на аппаратных ресурсах разработана диаграмма развертывания, представленная на рисунке 10. На этапе проектирования эта модель позволяет заранее предусмотреть распределение программных сред и артефактов, необходимых для стабильной работы системы [14].

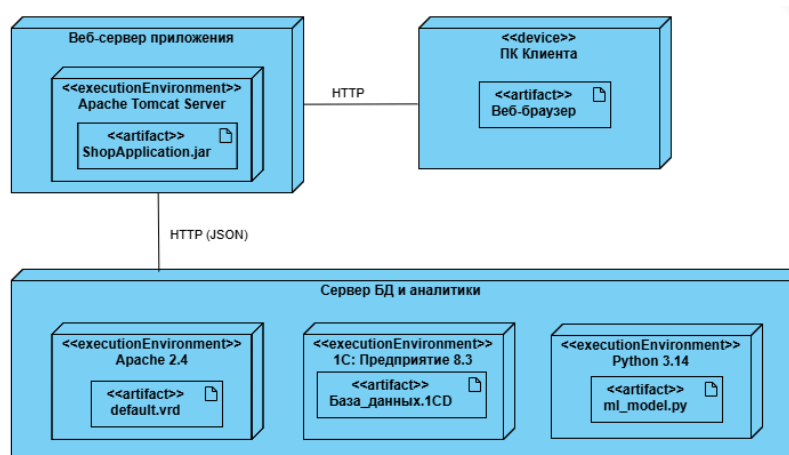


Рисунок 10 – Диаграмма развертывания

Архитектура системы проектируется как многоуровневая и включает три ключевых узла, между которыми разграничены зоны ответственности.

ПК Клиента (<<device>>). Доступ пользователя к системе будет осуществляться через стандартный Веб-браузер. Взаимодействие с интернет-магазином должно происходить по протоколу HTTP, что делает систему доступной с любого устройства.

Веб-сервер приложения. Данный узел проектируется как связующее звено между интерфейсом и данными. На нем планируется развертывание среды выполнения Java. Основным рабочим артефактом здесь должен стать исполняемый модуль ShopApplication.jar. Проект предусматривает использование встроенного сервера приложений Apache Tomcat, что позволит

обеспечить автономность сайта и высокую скорость обработки входящих запросов.

Сервер БД и аналитики. В рамках разрабатываемой системы этот узел является хранилищем данных и вычислительным центром. Проектируется совместное размещение трех программных сред на одном сервере для минимизации задержек.

1С:Предприятие 8.3: среда, предназначенная для управления артефактом База_данных.1CD (хранение номенклатуры и истории продаж).

Python 3.14: среда, предназначенная для запуска скрипта машинного обучения `ml_model.py`, где должны происходить все математические расчеты и формироваться рекомендации.

Apache 2.4: программный веб-сервер, который должен выступать в роли «шлюза». В его среде планируется размещение артефакта `default.vrd` (дескриптора публикации) для организации доступа к ShopAPI.

Связь между всеми частями системы проектируется на основе протокола HTTP. Он выбран как наиболее надежный и универсальный способ обмена данными.

Обмен данными между сайтом и учетной базой планируется реализовать через REST-интерфейс. Сведения о товарах, ценах и рассчитанных рекомендациях планируется передавать в формате JSON. Выбор формата JSON в качестве стандарта обмена на этапе проектирования обусловлен необходимостью быстрой передачи результатов интеллектуального анализа из 1С в пользовательский интерфейс. Это должно обеспечить высокую скорость загрузки страниц магазина, чтобы покупатель мог мгновенно увидеть блок «С этим товаром покупают» после открытия карточки товара.

В дополнение к физической схеме серверов нужно описать внутреннюю логическую архитектуру.

Это помогает сделать диаграмма компонентов, представленная на рисунке 11. На ней видно, из каких модулей и баз данных состоит система. И как между ними распределены основные функции.

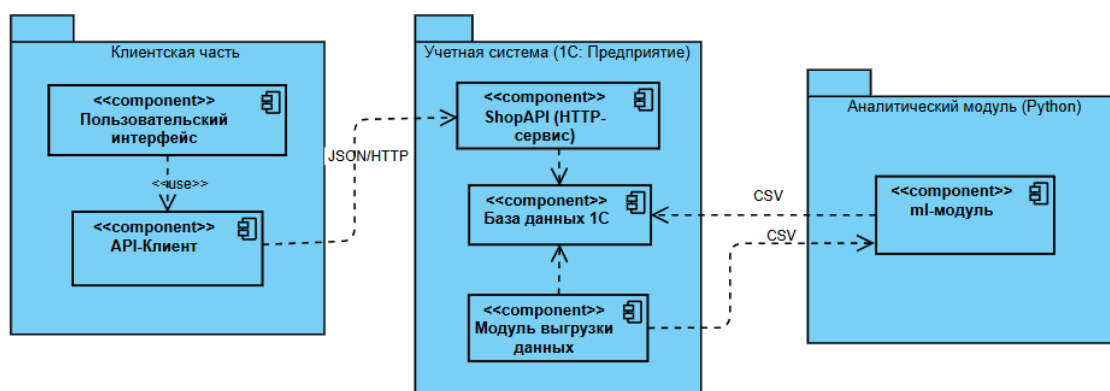


Рисунок 11 – Диаграмма компонентов системы

Клиентская часть включает в себя компоненты интерфейса и API-клиента для связи с учетной системой.

Учетная система (1С: Предприятие) выступает центром хранения данных. Здесь спроектированы компоненты для работы с базой данных, модуль для подготовки выгрузок и HTTP-сервис ShopAPI для обработки внешних запросов.

Аналитический модуль (Python) содержит компонент «ml-модуль», в котором сосредоточена математическая логика расчетов рекомендаций.

Взаимодействие компонентов через JSON/HTTP и CSV, показанное на диаграмме, обеспечивает независимость модулей друг от друга.

2.6 Проектирование ML-алгоритма товарных рекомендаций

Фундаментом разрабатываемой системы является аналитический модуль. Его главная задача – превратить «сырой» массив чеков из 1С в знания, полезные для увеличения продаж. Вместо ручного подбора сопутствующих товаров, система должна самостоятельно находить скрытые ассоциативные закономерности в поведении покупателей и формировать товарные рекомендации.

На этапе проектирования интеллектуального модуля основным решением является выбор алгоритма, который сможет автоматически находить связи между товарами. В качестве такого алгоритма планируется использовать коллаборативную фильтрацию (Collaborative Filtering). Этот метод считается

одним из фундаментальных в машинном обучении, в частности для построения рекомендательных систем [15].

В классификации задач машинного обучения данный метод относится к обучению без учителя (Unsupervised Learning). Это означает, что система проектируется для самостоятельного поиска закономерностей в данных без заранее подготовленных ответов. Программе не нужно объяснять, что, например, товары «чай» и «лимон» связаны – она сама сделает этот вывод, проанализировав историю чеков в 1С.

Архитектура системы предполагает использование подхода на основе памяти (Memory-based), реализующего принцип «ленивого обучения» (Lazy Learning). «Ленивый» алгоритм будет пересчитывать связи каждый раз при обновлении данных [16].

Благодаря этому рекомендации всегда будут актуальными и будут подстраиваться под то, что люди покупают именно сейчас.

Выделяются два основных подхода к реализации коллаборативной фильтрации.

Пользователь-ориентированный подход (User-based CF): направлен на поиск сходства между покупателями. Система проектируется так, чтобы рекомендовать пользователю товары, которые приобрели люди со схожими интересами («соседи»). Главным минусом этого метода является необходимость обязательной накопления истории покупок каждого клиента.

Объектно-ориентированный подход (Item-based CF): основывается на анализе связей между самими товарами. Математическая модель определяет, насколько часто определенные позиции товаров встречаются в общих заказах.

Для проектируемой системы был выбран именно Item-based подход [17]. Это решение обусловлено несколькими причинами.

Решение проблемы «холодного старта» для пользователя. Система способна выдавать рекомендации даже анонимным (неавторизованным) посетителям сайта, так как для расчета достаточно знать текущий выбор -товар, который пользователь просматривает в данный момент.

Стабильность связей: ассоциативные связи между товарами (например, ноутбук и мышь) меняются гораздо реже, чем личные предпочтения покупателей, что делает модель более надежной.

В основе проектируемого алгоритма лежит математическая метрика косинусного сходства (Cosine Similarity) [18]. Она выбрана как инструмент для измерения близости объектов. Для применения этой метрики необходимо перевести информацию о продажах из 1С в математический вид с помощью процесса векторизации.

Каждый товар в системе проектируется как вектор в многомерном пространстве. Размерность пространства определяется общим количеством заказов (чеков) в базе данных. Если в системе накоплено n заказов, то каждый товар представляется вектором с n координатами.

Координаты вектора принимают бинарные значения: 1, если товар был куплен в данном заказе, и 0, если покупка не совершалась.

Таким образом, сходство товаров определяется не по их названиям, а по «направлению» их векторов в пространстве продаж. Если два товара часто вместе покупают, угол между их векторами будет минимальным, а косинус этого угла – близким к единице.

Для вычисления коэффициента сходства между товаром и товаром используется следующая формула:

$$\text{sim}(A,B)=\frac{\sum_{i=1}^n A_i * B_i}{\sqrt{\sum_{i=1}^n A_i^2} * \sqrt{\sum_{i=1}^n B_i^2}},$$

где $\text{sim}(A,B)$ – коэффициент косинусного сходства между товаром;

A_i – координата вектора товара в транзакции (принимает значение 1, если товар присутствует в чеке, и 0 – если отсутствует);

B_i – координата вектора товара транзакции (принимает значение 1, если товар присутствует в чеке, и 0 – если отсутствует);

n – общее количество транзакций (чеков) в исследуемой выборке данных;

i – порядковый номер транзакции в базе данных.

На этапе проектирования этот числитель формулы интерпретируется как количество совместных покупок. Так как векторы состоят из нулей и единиц, произведение их координат даст «1» только в тех позициях, где оба товара присутствуют в одном чеке. Сумма этих произведений показывает, сколько раз данные товары встретились в истории продаж одновременно.

Знаменатель формулы – это ключевой элемент проектируемой модели, отвечающий за нормализацию данных. Знаменатель представляет собой произведение длин (норм) векторов товаров.

Логика этого решения заключается в уменьшении веса избыточно популярных товаров. Без такой нормировки товары-лидеры продаж (например, «Пакет» или «Хлеб») имели бы высокие показатели сходства со всем ассортиментом магазина. Деление на длину вектора позволяет системе находить уникальные ассоциативные связи и предлагать покупателю действительно релевантные товары, исходя из его текущего выбора.

При проектировании архитектуры была выявлена проблема масштабируемости. В типичном интернет-магазине матрица покупок на 99% состоит из нулей, так как один человек покупает лишь малую часть ассортимента. Хранение такой «пустой» таблицы в памяти компьютера неэффективно.

В связи с этим спроектировано использование формата CSR (Compressed Sparse Row). CSR – то один из стандартных форматов хранения разреженной матрицы. Это метод сжатия данных, при котором система полностью игнорирует нулевые значения и хранит в оперативной памяти только три массива данных: значения единиц, индексы товаров и указатели на начало заказов [19]. Это решение позволяет:

- снизить потребление памяти в десятки раз;
- ускорить математические операции.

Алгоритм косинусного сходства будет выполнять вычисления только там, где есть реальные продажи, «пролетая» мимо пустых нулей.

Логика действий аналитического модуля представлена на рисунке 12.



Рисунок 12 – Блок-схема алгоритма

Логика работы аналитического модуля, представленная на блок-схеме, разделена на несколько ключевых этапов обработки информации.

Подготовка и кодирование данных. На вход модуля поступает массив продаж из 1С. Поскольку алгоритмы машинного обучения не могут напрямую работать с текстовыми названиями, всем уникальным товарам и номерам заказов присваиваются целочисленные индексы. Это превращает текстовые данные в цифровой набор, пригодный для математических вычислений.

Формирование матрицы взаимодействий. На основе полученных индексов строится разреженная матрица в формате CSR. В этой структуре строки

соответствуют уникальным заказам, а столбцы – товарам из каталога. На пересечении ставится единица, если товар присутствует в чеке, и ноль – если покупка не совершалась. Использование разреженного формата позволяет системе игнорировать пустые ячейки, что важно для экономии памяти.

Транспонирование матрицы. Для перехода к анализу сходства объектов выполняется операция транспонирования. В результате строками матрицы становятся товары, а столбцами заказы. Теперь каждая строка представляет собой бинарный вектор, который описывает всю историю продаж конкретной позиции товара.

Математический расчет и ранжирование. К векторам товаров применяется метрика косинусного сходства. Алгоритм попарно сравнивает товары и вычисляет коэффициенты их близости от 0 до 1. После этого для каждой позиции товара выполняется сортировка найденных связей. Система отсекает совпадение товара самим с собой и выбирает топ-3 наиболее подходящих сопутствующих товара.

Экспорт модели. Финальным шагом является сохранение рекомендаций в файл формата CSV.

Таким образом, спроектированный алгоритм позволяет полностью автоматизировать процесс выявления скрытых связей между товарами.

2.7 Проектирование пользовательского интерфейса

2.7.1 Проектирование интерфейса управления в 1С

Интерфейс для сотрудников интернет-магазина проектируется средствами платформы «1С:Предприятие 8.3».

Логика управления строится на разделении функций по подсистемам, что позволяет сотрудникам быстро находить нужные инструменты.

Согласно проекту, командный интерфейс должен включать несколько функциональных разделов и объектов.

Подсистема «Складской учет» – предназначена для управления товарными запасами. В неё планируется включить:

- справочник «Товары» (каталог продукции);
- документ «Приходная накладная» (регистрация поступлений);
- регистр накопления «Остатки товаров» (контроль наличия).

Подсистема «Продажи и клиенты» – основной блок для регистрации торговых операций. Включает:

- документ «Реализация товаров» (фиксация продаж для накопления истории);
- справочник «Контрагенты» (база покупателей с их контактами);
- справочник «Сотрудники» (список персонала);
- регистр сведений «Цены товаров» (установка актуальной стоимости).

Подсистема «Интеллектуальный сервис» – технический раздел для взаимодействия с ML-модулем. Включает:

- обработку «Интеллектуальный анализ» (форма с кнопкой запуска внешнего скрипта Python);
- регистр сведений «Рекомендации товаров» (хранилище рассчитанных связей).

Центральным элементом управления должна стать форма обработки «Интеллектуальный анализ». Проектируемые требования к форме включают наличие управляющей кнопки для ручного запуска расчета рекомендаций.

Система предполагает, что информация о рекомендациях должна быть доступна менеджеру не только в виде записей регистра, но и непосредственно в карточке каждого товара в справочнике «Товары».

Для этого в нижней части формы проектируется создание динамической таблицы. Требования к этому элементу: отображение списка товаров, которые алгоритм предлагает покупать вместе с текущим и вывод рассчитанного модулем коэффициента сходства для каждой пары. Такое построение интерфейса позволяет менеджеру видеть полную картину по каждому товару: от текущих остатков и цены до готовых рекомендаций, рассчитанных алгоритмом.

2.7.2 Проектирование интерфейса веб-приложения

Проектируемый веб-интерфейс должен обеспечивать покупателю простую навигацию и наглядное представление результатов интеллектуального анализа. Система должна включать следующие основные разделы и страницы:

- «главная» – входная страница сайта;
- «каталог» – список товаров, доступных к покупке;
- «карточка товара» – страница с подробным описанием и рекомендациями;
- «оформление заказа» – модальное окно для ввода контактных данных.

На главной странице планируется размещение приветственного баннера с информацией о магазине, а также кнопку для быстрого перехода в каталог.

На странице каталога должен отображаться динамический список товаров. Проектируемая логика предполагает, что карточки товаров выводятся только при наличии остатка в учетной системе. Каждая карточка должна содержать наименование, цену и кнопку перехода к детальному просмотру.

На странице карточки товара проектируется разделение на две зоны: информационную и рекомендательную. В информационной части указываются характеристики товара и кнопка добавления в корзину. В рекомендательной части должен располагаться блок «С этим товаром покупают». Требование к этому блоку – вывод до трех товаров с наиболее высокими коэффициентами ассоциативной связи, полученными в ходе работы ML-модели.

Интерфейс оформления заказа проектируется в виде модального окна. Форма должна поддерживать «гостевой заказ», запрашивая только имя и контактный номер телефона без необходимости регистрации. Такой подход снижает порог входа для новых покупателей и ускоряет процесс оформления. Переходы между разделами предполагаются интуитивно понятными, без лишних действий. Визуальное оформление выдерживается в едином стиле.

3 Реализация системы

3.1 Разработка структуры объектов информационной базы 1С

Процесс создания объектов в системе «1С:Предприятие 8.3» был направлен на формирование надежной базы данных, которая служит фундаментом для последующего интеллектуального анализа продаж и выявления рекомендаций. Чтобы система могла автоматически выдавать рекомендации, необходимо было не просто организовать хранение товаров, но и обеспечить фиксацию каждой торговой операции.

Разработка велась в среде «Конфигуратор». В ходе работы были созданы и настроены следующие объекты метаданных:

- справочники – для хранения постоянной информации о товарах, клиентах и сотрудниках компании;
- документы – для регистрации реальных событий (поступление товаров и их продажа), на основе которых формируется история покупок;
- регистры накопления – для автоматического учета количества товаров, оставшихся на складах;
- регистры сведений – для хранения актуальных цен и готовых результатов расчетов машинного обучения;
- http-сервисы – для создания программного интерфейса, через который сайт на Java получает данные из 1С;
- перечисления – для хранения фиксированных списков значений (например, единиц измерения);
- регламентное задание – для автоматизации периодического пересчёта рекомендаций;
- обработки – для реализации сервисных функций, для управления процессом обмена данными с аналитическим модулем Python.

Для того чтобы наглядно представить внутреннее устройство разработанной конфигурации, была составлена диаграмма компонентов. Она

показывает, из каких объектов состоит система и как они связаны между собой. Диаграмма представлена на рисунке 13.

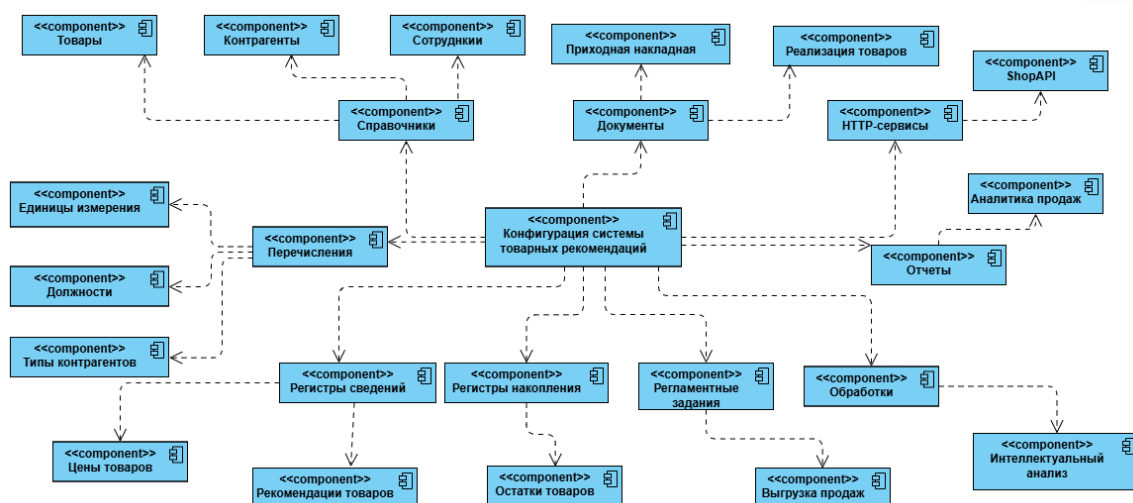


Рисунок 13 – Диаграмма компонентов конфигурации

Разработанная диаграмма наглядно показывает, как устроена конфигурация изнутри. Далее рассмотрено назначение и структура основных объектов более подробно.

Справочники. Справочник является объектом ссылочного типа, который позволяет хранить в системе списки данных. Главным отличие справочника от документа является наличие или отсутствие движений по регистрам. Каждому элементу в справочнике система автоматически присваивает уникальный код.

Для разрабатываемой системы были созданы следующие справочники:

- товары;
- контрагенты;
- сотрудники.

«Сотрудники» – справочник фиксирует список персонала магазина. Объект нужен для того, чтобы закрепить за каждой продажей ответственного менеджера. Для полноты данных были добавлены поля: ФИО, Должность, Дата рождения и контакты. Наличие этого справочника в системе позволяет в будущем анализировать работу персонала и видеть, кто из менеджеров совершает больше всего успешных продаж. Форма справочника представлена на рисунке 14.

Наименование	Код	Должность	Телефон	Email	Дата рождения
Алексеева О.В.	000000001	Менеджер продаж	+7 (911) 222-33-44	alex123@mail.ru	06.06.1987 г.
Петров И.С.	000000002	Менеджер продаж	+7 (499) 555-44-33	petrov@office-ld.ru	12.09.2000 г.
Сидорова М.И.	000000003	Менеджер продаж	+7 (933) 444-66-77	marina_sid@bk.ru	05.07.2004 г.
Иванова И.Р.	000000004	Администратор	+7 (892) 444-32-12	ivanovi880@mail.ru	03.08.1988 г.
Козлов Д.Д.	000000005	Зав. складом	+7 (123) 987-34-66	kozloww@office-ld.ru	13.12.1976 г.
Ковалева С.Г.	000000006	Товаровед	+7 (499) 555-44-33	kovaleva2000@mail.ru	21.06.2000 г.
Гурьева Н.Я.	000000007	Бухгалтер	+7 (999) 432-76-28	gureva.natalia@mail.ru	18.02.1980 г.
Арефьев В.В.	000000008	Товаровед	+7 (854) 382-12-37	arefew@office-ld.ru	04.05.2001 г.
Ильин И.Ф.	000000009	Директор	+7 (927) 123-77-11	iliniya1970@mail.ru	01.03.1970 г.

Рисунок 14 – Форма справочника «Сотрудники»

«Товары» – это центральный объект всей системы. В нем хранится каталог продукции магазина. Справочник является основой для работы модуля рекомендаций: именно эти товары модуль анализирует и связывает между собой. Помимо наименования, в справочнике есть поля «Описание товара» и «Единица измерения». Форма справочника «Товары» представлена на рисунке 15.

Наименование	Код	Описание	Единица измерения
Лимон	000000005	Свежие отборные лимоны с тонкой кожцей. Богаты ви...	Кг
Печенье «Юбилейное»	000000006	Классическое сахарное печенье, изготовленное по тра...	Шт
Шоколад "Аленка"	000000013	Легендарный молочный шоколад с нежным вкусом. Пл...	Шт
Чай черный	000000004	Насыщенный байховый чай высшего сорта. Упаковка 1...	Шт
Сахар (песок)	000000002	Белый кристаллический сахар, идеально подходит для...	Кг
Сливки 10%	000000003	Натуральные питьевые сливки. Отличное дополнение ...	Шт
Кофе (в зернах)	000000001	Смесь арабики и робусты средней обжарки с ярким аро...	Шт
Конфеты (ромашка)	000000011	Помадные глазированные конфеты с нежным вкусом кр...	Кг
Мед(в банке)	000000012	Натуральный цветочный мед урожая этого года. Стекл...	Шт
Коврик для мыши	000000009	Износостойкий коврик с нескользящим основанием дл...	Шт
Мышь беспроводная	000000008	Оптическая беспроводная мышь с эргономичным диза...	Шт
Кабель HDMI	000000010	Высокоскоростной кабель для передачи видео и ауд...	Шт
Ноутбук	000000007	Производительный ноутбук для работы и учебы. 8 ГБ ...	Шт

Рисунок 15 – Форма справочника «Товары»

«Контрагенты» – база данных всех покупателей. Был расширен стандартный список полей реквизитами ИНН, Телефон и Email. Также реквизит

«Тип контрагента» позволяет разделять клиентов на физлиц и юрлица. Форма справочника представлена на рисунке 16.

Наименование	Код	Тип контрагента	Телефон	Email	ИНН
Иванов Иван	000000001	Физическое лицо	+7 (900) 123-45-67	ivanov_ivan@mail.ru	770102030405
ИП Алексеев	000000005	Физическое лицо	+7 (911) 222-33-44	alekseev_work@yandex	500100200300
Магазин «Ромашка»	000000004	Юридическое лицо	+7 (495) 777-88-99	shop@romashka.ru	7712345678
ООО "ОфисМаг"	000000007	Юридическое лицо	+7 (499) 555-44-33	info@office-ltd.ru	7789765466
ООО «Офис»	000000006	Юридическое лицо	+7 (499) 555-44-33	info@office-ltd.ru	7709876543
Петров Петр	000000002	Физическое лицо	+7 (922) 333-55-66	petr_p88@gmail.com	660504030201
Сидорова Анна	000000003	Физическое лицо	+7 (933) 444-66-77	anna_sid@bk.ru	340607080900

Рисунок 16 – Форма справочника «Контрагенты»

Документы. Если справочники отвечают за хранение данных, то документы предназначены для фиксации конкретных событий в деятельности магазина. Каждый документ привязан к определенному моменту времени. В рамках проекта разработаны два вида документов, обеспечивающих полный цикл движения товара, а именно приход товаров и их реализацию.

«Приходная накладная» — используется для регистрации поступления товаров от поставщиков на склад. В табличной части документа реализованы реквизиты для учета не только количества, но и цен закупки. Данный объект конфигурации является источником информации для формирования складских остатков. Документ «Приходная накладная» служит основой для ведения складского учёта. При его проведении система автоматически изменяет данные в регистре накопления «Остатки товаров», увеличивая количество поступивших позиций. Это позволяет всегда иметь точную информацию о наличии товаров на складе. Таким образом, документ обеспечивает приход товаров и служит источником данных для складского учёта. Форма документа «Приходная накладная» представлена на рисунке 17.

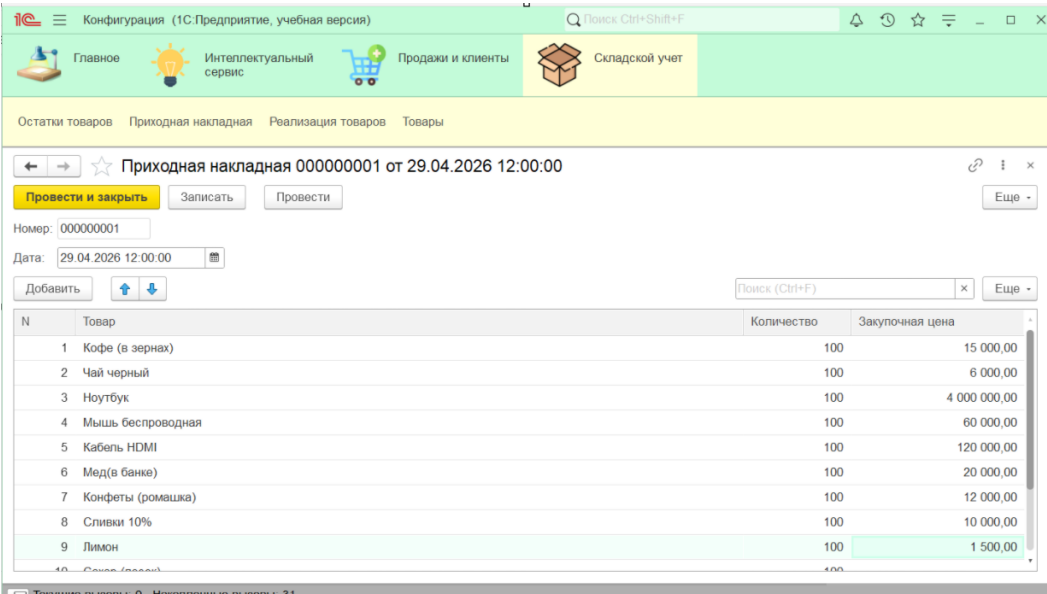


Рисунок 17 – Форма документа «Приходная накладная»

«Реализация товаров» – документ, фиксирующий факт продажи товара покупателю. Именно данные этого документа служат источником информации для работы алгоритмов машинного обучения. На основе истории проведенных реализаций модуль на Python выявляет ассоциативные связи между товарами. В форме документа реализована автоматизация: при выборе товара система автоматически подставляет актуальную цену продажи из регистров сведений. Также предусмотрен механизм создания этого документа при поступлении заказа с веб-сайта. Форма документа «Реализация товаров» представлена на рисунке 18.

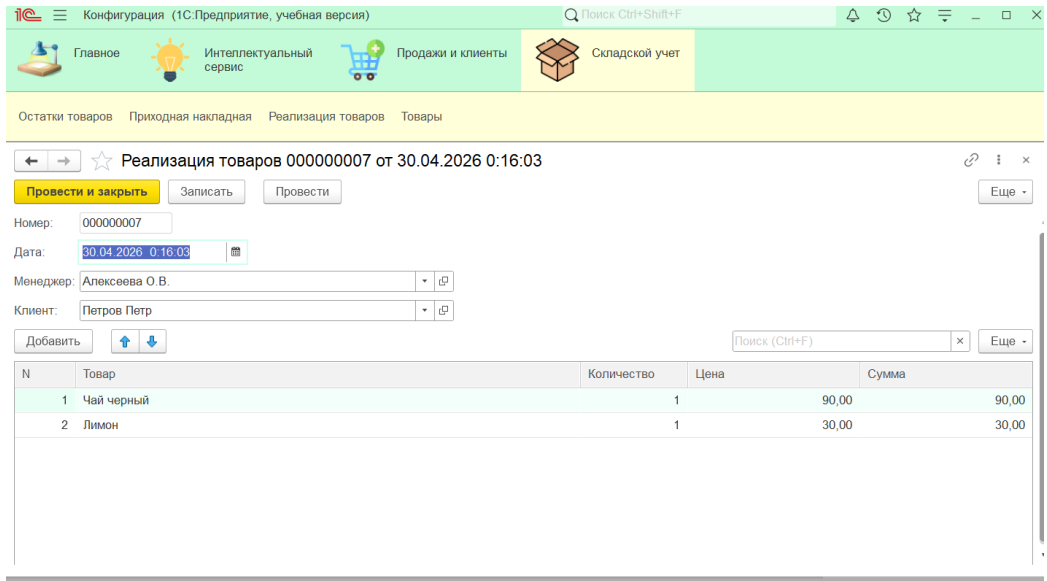


Рисунок 18 – Форма документа «Реализация товаров»

Регистры. Для хранения итоговой информации и быстрого доступа к данным в системе реализованы регистры накопления и регистры сведений. Эти объекты позволяют сайту мгновенно получать нужные сведения без повторного анализа всех документов базы.

Регистр сведений «Цены товаров». Хранит историю изменения стоимости товаров. Регистр настроен на выдачу актуальной цены на текущий момент времени. Это гарантирует, что покупатель на сайте всегда видит верную стоимость, зафиксированную в учетной системе. Форма регистра сведений «Цены товаров» представлена на рисунке 19.

Период	Товар	Цена
12.05.2026	Сахар (песок)	100,00
12.05.2026	Сливки 10%	150,00
12.05.2026	Чай черный	90,00
12.05.2026	Лимон	30,00
12.05.2026	Печенье «Юбилейное»	50,00
12.05.2026	Ноутбук	50 000,00
12.05.2026	Мышь беспроводная	1 000,00
12.05.2026	Коврик для мыши	500,00
12.05.2026	Кабель HDMI	2 000,00
12.05.2026	Конфеты (ромашка)	200,00
12.05.2026	Мед(в банке)	300,00
12.05.2026	Шоколад "Аленка"	80,00

Рисунок 19 – Форма регистра сведений «Цены товаров»

Регистр сведений «Рекомендации товаров». Является основным объектом системы, в который записываются результаты работы машинного обучения. В регистре хранятся пары связанных товаров и рассчитанные модулем коэффициенты сходства. Эти данные используются при формировании блока «С этим товаром покупают» на сайте. Обновление записей происходит автоматически при каждом пересчёте рекомендаций. Благодаря этому покупатель всегда видит актуальные предложения, основанные на последних продажах. Такой способ хранения данных позволяет сайту работать очень быстро, так как ему не нужно каждый раз делать сложные вычисления. Форма регистра сведений «Рекомендации товаров» представлена на рисунке 20.

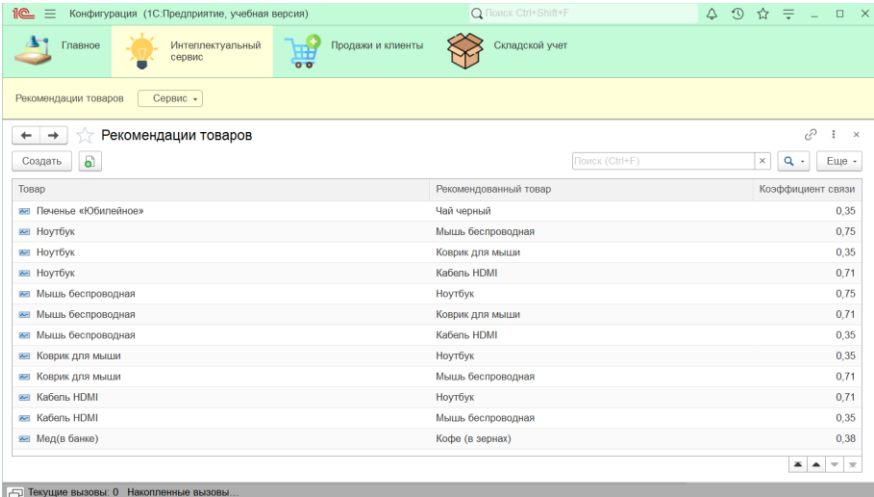


Рисунок 20 – Форма регистра сведений «Рекомендации товаров»

Регистр накопления «Остатки товаров». Обеспечивает учет наличия продукции на складах. Изменение данных (движения) в этом регистре происходит автоматически при проведении документов. Документ «Приходная накладная» выполняет движение «Приход», увеличивая количество товара на складе. Документ «Реализация товаров» выполняет движение «Расход», списывая проданные товары со склада.

Наличие актуальных остатков позволяет ShopAPI фильтровать ассортимент для сайта, чтобы покупатель видел только те товары, которые реально можно заказать. Форма регистра накопления «Остатки товаров» представлена на рисунке 21.

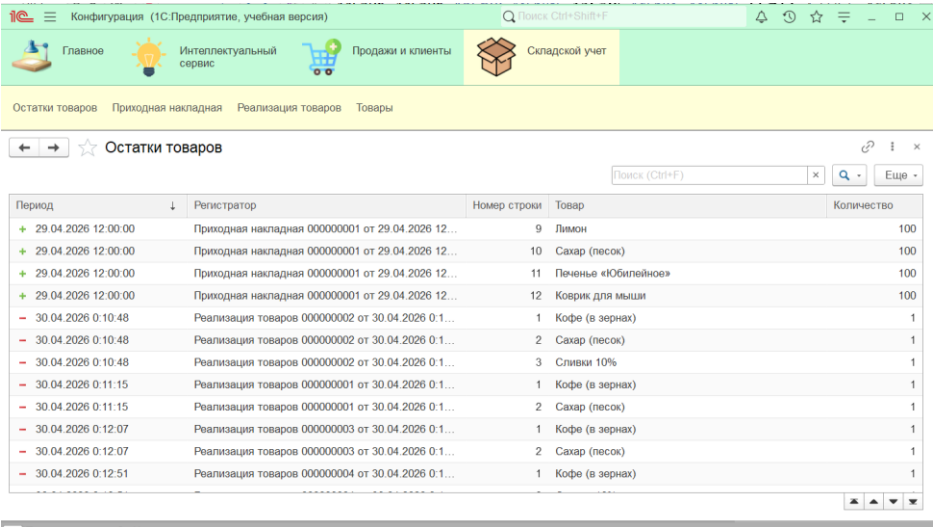


Рисунок 21 – Форма регистра накопления «Остатки товаров»

Регламентное задание. Для автоматического, не требующего участия менеджера, обновления товарных рекомендаций в системе настроено регламентное задание «Выгрузка продаж». Оно инициирует процедуру выгрузки данных из документов «Реализация товаров», запуск аналитического модуля на Python и последующую загрузку результатов в регистр сведений «Рекомендации товаров». Периодичность выполнения задания установлена равной 7 дням. Настройка расписания регламентного задания представлена на рисунке 22.

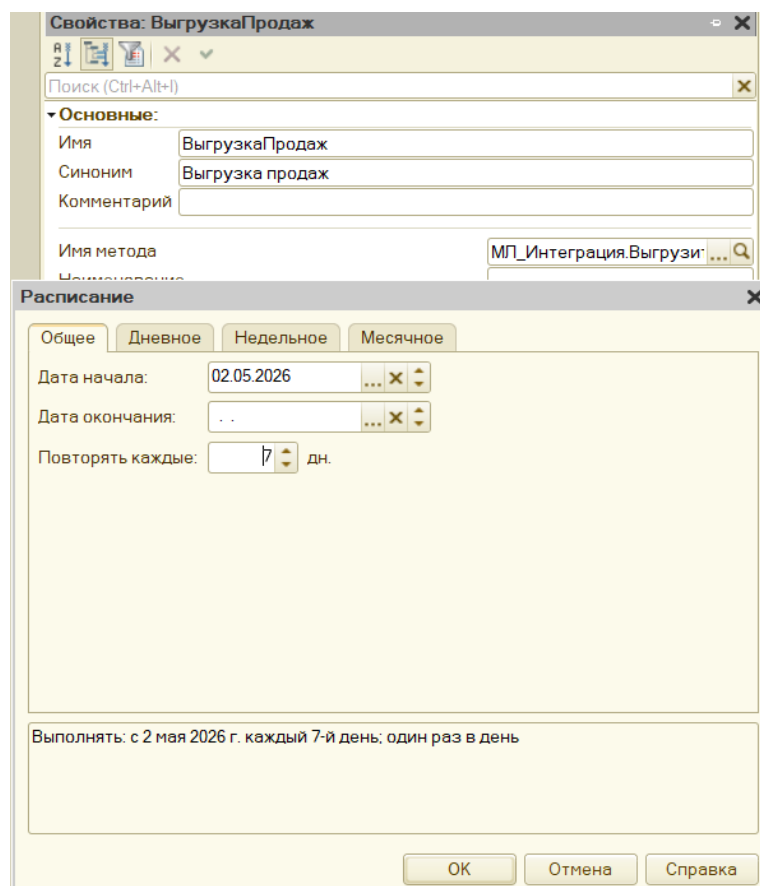


Рисунок 22 – Настройка расписания регламентного задания

Отчеты. Для визуализации накопленной информации и оценки деятельности магазина реализован отчет «Аналитика продаж». Отчет создан с помощью схемы компоновки данных. Он позволяет менеджеру увидеть, какие товары пользуются спросом. Также пользователь может самостоятельно настраивать фильтры, например, чтобы посмотреть продажи за конкретный месяц. Пример реализации отчета представлен на рисунке 23.

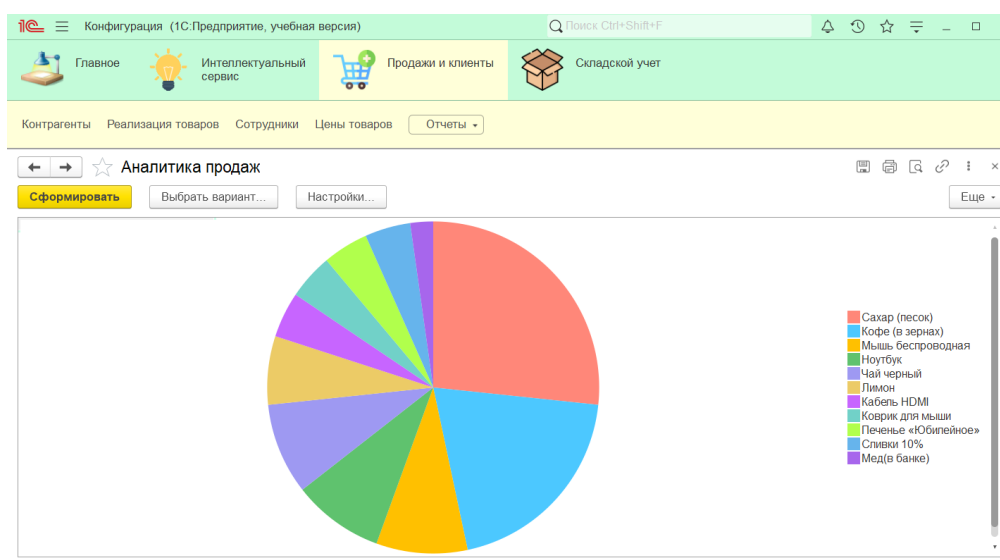


Рисунок 23 – Отчет «Аналитика продаж»

Таким образом, в среде «Конфигуратор» 1С:Предприятие 8.3 была реализована структура объектов, полностью покрывающая потребности интернет-магазина и системы товарных рекомендаций.

3.2 Реализация программных средств для системы рекомендаций

3.2.1 Реализация алгоритмов учета и обмена данными на языке 1С

Разработка логики программного комплекса велась на трех языках программирования: встроенном языке «1С», Python и Java. Это позволило использовать преимущества каждой технологии: надежность учета 1С, аналитические мощности Python и быстрдействие интерфейса Java.

В среде «1С:Предприятие 8.3» программный код распределен по специализированным модулям. Каждый модуль отвечает за свой участок работы, что упрощает разработку и последующее сопровождение системы. Благодаря этому изменения в одном модуле не затрагивают другие, что снижает риск появления новых ошибок. Каждый модуль написан с учётом его функционального назначения и не содержит лишней логики. Это облегчает тестирование и модификацию кода в дальнейшем.

Описание основных модулей приведено в таблице 17.

Таблица 17 – Описание модулей программы

Название объекта конфигурации	Вид модуля	Описание
МЛ_Интеграция	Общий модуль	Содержит серверные процедуры для выгрузки данных в CSV, вызова внешнего скрипта Python и загрузки результатов анализа в базу (Регистр сведений).
ShopAPI	Модуль HTTP-сервиса	Реализует программный интерфейс (REST API) для обеспечения двустороннего обмена данными с веб-приложением. Модуль содержит методы динамической сериализации данных (перевод цен и рекомендаций в формат JSON), а также процедуры десериализации входящих запросов для автоматизированной генерации документов «Реализация товаров»
ИнтеллектуальныйАнализ	Модуль формы	Содержит код вызова процедур интеграции при нажатии пользователем кнопки управления.
РеализацияТоваров	Модуль формы	Обеспечивает автоматизацию интерфейса менеджера (подстановка цен и расчет сумм).

Общий модуль «МЛ_Интеграция» обеспечивает взаимодействие базы «1С» с внешней средой Python. В нем реализована общая серверная процедура «ВыгрузитьДляОбученияНаСервере», которая предназначена для подготовки данных к анализу команды для запуска самого процесса расчета.

Работа модуля разделена на три этапа.

Программа формирует выборку продаж из документа «Реализация товаров» за последние 6 месяцев. Полученные сведения очищаются от спецсимволов и сохраняются в файл `sales_data.csv` для передачи в аналитический модуль. Структура этого файла организована таким образом, что первая колонка содержит номер заказа, а вторая – наименование купленного товара. Это позволяет аналитическому модулю определять, какие товары находились в одном чеке.

Для выполнения расчетов используется системный объект `WScript.Shell`. С его помощью 1С отправляет команду операционной системе на запуск Python в фоновом режиме. Такой подход позволяет проводить сложные вычисления незаметно для пользователя, не вызывая зависания интерфейса 1С.

После завершения работы аналитического модуля общий модуль считывает полученный файл `recommendations.csv`. Данный файл содержит три столбца: основной товар, рекомендованный к нему товар и рассчитанный коэффициент связи (степень сходства). Полученные данные автоматически записываются в регистр сведений «Рекомендации товаров». Код метода представлен в листинге 1 Приложения Б. С полным кодом программы можно ознакомиться в GitHub [20].

Для организации обмена данными между базой «1С» и веб-сайтом в конфигурации реализован HTTP-сервис ShopAPI. Он функционирует по принципам REST-архитектуры и предоставляет два метода для взаимодействия с внешним приложением.

Для того чтобы сайт мог отображать актуальную информацию, в HTTP-сервисе ShopAPI был реализован метод `get_dataGetProduct`

Алгоритм данного метода обеспечивает подготовку актуального каталога товаров для сайта. Главная задача – собрать из разных частей базы 1С (справочников и регистров) единый пакет данных, который содержит всё необходимое для показа товара и его рекомендаций.

Программный код выполняет последовательную выборку информации.

С помощью запроса определяются позиции номенклатуры, имеющие положительный остаток на складе.

Из соответствующего регистра сведений извлекаются актуальные цены товаров.

Система сопоставляет каждый товар с его персональными рекомендациями, которые были ранее рассчитаны модулем Python и сохранены в регистре рекомендаций.

Итоговый массив данных программно преобразуется в объектную структуру и сериализуется в формат JSON. Это позволяет сайту на Java получать готовую информационную выдачу, объединяющую учетные данные и результаты машинного обучения. Код метода представлен в листинге 2 Приложения Б.

Для обеспечения обратной связи в сервисе ShopAPI реализован метод `orderCreateOrder`. Его основной задачей является автоматический прием данных о покупке и их регистрация в учетной системе.

Процесс обработки входящего запроса состоит из следующих этапов:

Парсинг данных. Система получает запрос от сайта и выполняет десериализацию JSON-пакета. Текстовая информация преобразуется в объект «Структура», это позволяет программно обращаться к данным о покупателе и списке выбранных товаров.

В базе 1С программно создается новый документ «Реализация товаров». Система заполняет обязательные реквизиты шапки (дата, ответственные лица) и переходит к формированию табличной части.

Обработка списка товаров из заказа выполняется в цикле. Система сопоставляет идентификаторы товаров (GUID), пришедшие с сайта, с объектами в справочнике «Товары». Для каждой строки автоматически рассчитывается итоговая сумма.

Код обернут в блок обработки исключений. Если документ успешно записан в базу, сервис возвращает сайту статус HTTP 201 Created и номер заказа. В случае технического сбоя система возвращает код 500 Error, предотвращая

некорректное завершение операции. Код метода представлен в листинге 3 Приложения Б. С полным кодом программы можно ознакомиться в GitHub [20].

В модуле формы обработки «Интеллектуальный анализ» прописан код вызова интеграционных процедур. При нажатии пользователем кнопки «Выгрузить для обучения» система обращается к описанному ранее общему модулю «МЛ_Интеграции» и запускает его метод. Это позволяет менеджеру обновлять базу знаний в ручном режиме, не дожидаясь срабатывания регламентного задания.

В форме документа «Реализация товаров» реализованы методы для автоматизации рутинных операций менеджера. Например, при выборе товара в табличной части срабатывает процедура «ТоварыТоварПриИзменении», которая обращается к регистру цен и мгновенно подставляет актуальную стоимость в документ. Также при любом изменении количества или цены товара срабатывают соответствующие обработчики событий «ТоварыКоличествоПриИзменении» или «ТоварыЦенаПриИзменении», которые пересчитывают результат. Такой подход полностью исключает риск арифметических ошибок при ручном вводе.

3.2.2 Реализация аналитического модуля на языке Python

Для формирования товарных рекомендаций был разработан аналитический модуль, использующий методы машинного обучения. Программная реализация модуля выполнена на языке Python, так как он обладает мощными инструментами для векторизации данных и расчета модели коллаборативной фильтрации.

В процессе разработки были выбраны следующие библиотеки:

Pandas – используется для загрузки транзакционных данных из CSV-файла, их предварительной очистки и преобразования в табличный вид [21].

SciPy (модуль sparse) – применяется для построения разреженных матриц в формате CSR. Это важное решение для обеспечения масштабируемости

системы, так как оно позволяет не хранить в памяти миллионы пустых нулевых значений [22].

Scikit-learn – предоставляет алгоритм вычисления косинусного сходства, используемый для работы с разреженными структурами данных.

Процесс начинается с чтения файла `sales_data.csv` средствами библиотеки Pandas (метод `read_csv`). На данном этапе данные представлены в виде текстовых строк (имена товаров и номера заказов), которые не пригодны для математических операций.

Для подготовки данных к расчету к колонкам с номерами заказов и названиями товаров применяется метод `astype('category')`. После его выполнения тип данных становится категориальным: это означает, что каждому уникальному значению в списке автоматически присваивается свой целочисленный индекс. Чтобы извлечь эти индексы и использовать их в построении модели, применяется команда `.cat.codes`. Это позволяет заменить тяжелые текстовые названия короткими числами, что ускоряет работу программы и подготавливает данные к векторизации.

Так как в магазине много товаров, а в одном чеке их обычно всего несколько, итоговая таблица покупок будет почти полностью состоять из нулей. Чтобы не перегружать память компьютера хранением нулей, используется конструктор `sparse.csr_matrix` из библиотеки SciPy.

Этот метод позволяет создать разреженную матрицу, записывая только реальные факты продаж. Для этого в функцию передаются три параметра:

Массив значений `[1] * len(df)`, где каждая единица означает наличие товара в чеке.

Координаты строк, полученные из индексов заказов (`.cat.codes`).

Координаты столбцов, полученные из индексов товаров (`.cat.codes`).

В результате формируется матрица, в которой строки соответствуют уникальным заказам, а столбцы – наименованиям товаров, в формате CSR, и которая хранит только адреса единиц. Это техническое решение позволяет системе легко обрабатывать даже очень большие объемы продаж из 1С.

Для того чтобы система могла сравнивать товары между собой, выполняется операция транспонирования (метод `.T`). В результате этой операции структура данных меняется: строками становятся товары, а столбцами – заказы.

Теперь каждая строка в таблице -это бинарный вектор, который описывает историю продаж конкретного товара. Для того чтобы создать рекомендации, необходимо вычислить, насколько эти векторы близки друг к другу. Для этого в модуле применяется функция `cosine_similarity` библиотеки `Scikit-learn`. Она попарно сравнивает векторы всех товаров и вычисляет косинус угла между ними.

Итогом этого этапа является матрица сходства, в которой на пересечении каждой пары товаров стоит число от 0 до 1. Чем чаще товары оказывались в одних и тех же заказах, тем выше этот коэффициент.

На финальном этапе выполняется программная обработка полученных коэффициентов. Чтобы покупатель на сайте видел только самые подходящие товары, в коде реализован цикл, который для каждой позиции номенклатуры выполняет следующие действия.

С помощью метода `sort_values(ascending=False)` все найденные связи сортируются от самых сильных к самым слабым.

Методом `drop` из списка удаляется сам анализируемый товар, так как его сходство с самим собой всегда максимально, но не имеет смысла для рекомендаций.

Выполняется отбор Топ-3 наиболее релевантных позиций.

Полученные данные сохраняются в файл `recommendations.csv` с помощью метода `to_csv`, где первый столбец- товар, второй-рекомендованный товар и третий-коэффициент связи. Код алгоритма представлен в листинге 4 Приложения В.

В итоге, этот код на Python позволяет не подбирать рекомендуемые товары вручную, а делать это автоматически на основе реальных продаж. Программа сама находит скрытые связи между покупками и определяет, что именно стоит предложить клиенту. Благодаря переходу на разреженные матрицы, расчеты

проходят быстро, и система не будет зависать даже при большом количестве заказов.

3.2.3 Реализация серверной и клиентской логики веб-приложения

Чтобы покупатели могли взаимодействовать с системой и видеть результаты работы машинного обучения, разработан веб-интерфейс. Он выполняет роль «витрины», которая запрашивает информацию у учетной системы и показывает её пользователю.

В качестве основного инструмента был выбран фреймворк Spring Boot на языке Java. Он обеспечил стабильность приложения, а встроенный сервер Tomcat избавил от лишних настроек при запуске сайта.

Для того чтобы код был организованным, была использована паттерн проектирования MVC (Model-View-Controller). Использование этого подхода позволило разделить внешний вид страниц от внутренней логики программы. Весь программный код распределен по функциональным модулям, описание которых приведено в таблице 18.

Таблица 18 – Описание программных модулей Java-приложения

Название объекта	Вид модуля	Описание
ShopApplication.java	Главный класс	Точка входа в программу. Отвечает за запуск Spring-контекста и веб-сервера.
ShopController.java	Контроллер	Обрабатывает HTTP-запросы от клиента (браузера). Использует аннотации @GetMapping, @PostMapping для маршрутизации. Передаёт данные из модели в Thymeleaf-шаблоны, а также принимает JSON заказа и перенаправляет его в сервис.

Продолжение таблицы 18

Название объекта	Вид модуля	Описание
ShopService.java	Сервис	Реализует интеграцию с 1С. Выполняет HTTP-запросы к ShopAPI и преобразует JSON-ответы в объекты Java.
Product.java	Модель данных	Описывает структуру товара: уникальный идентификатор, наименование, цена, а также список рекомендованных товаров. Использует аннотацию @Data из библиотеки Lombok для автоматической генерации геттеров, сеттеров и конструкторов.

Веб-приложение не имеет собственной базы данных. Вся информация подгружается в реальном времени из «1С:Предприятия». Для этого в модуле ShopService используется класс RestTemplate. Когда пользователь переходит в каталог или карточку товара, сервис отправляет GET-запрос к ShopAPI. Полученный JSON-пакет содержит не только товары и цены, но и список рекомендованных товаров, которые ранее рассчитал модуль на Python.

Для отправки оформленного заказа в 1С используется POST-запрос. Когда пользователь подтверждает заказ (вводит имя и телефон), JavaScript собирает данные корзины и отправляет их на серверное Java-приложение. Контроллер ShopController передаёт JSON-строку в метод sendOrderTo1C(jsonOrder) сервиса ShopService. Внутри этого метода RestTemplate выполняет POST-запрос к соответствующему HTTP-сервису 1С (/shop/hs/v1/order), передавая JSON с контактными данными и списком товаров. Ответ от 1С (статус выполнения и номер заказа) возвращается обратно в браузер. Описание методов представлены в таблице 19.

Таблица 19 – Методы сервиса для работы с 1С

Метод	Тип запроса	Назначение
<code>getAllProducts()</code>	GET	Получает список всех товаров с ценами и рекомендациями
<code>getProductById(id)</code>	GET	Получает конкретный товар с ценами и рекомендациями.
<code>sendOrderTo1C(jsonOrder)</code>	POST	Передаёт оформленный заказ в 1С, возвращает номер документа

Ниже приведено подробное описание каждого метода сервиса ShopService, их параметров и возвращаемых значений.

Метод `getAllProducts()`.

Параметры: отсутствуют.

Тип запроса к 1С: GET.

Что делает: отправляет GET-запрос к `http://localhost:8080/shop/hs/v1/data`. Полученный JSON-массив автоматически десериализуется в массив объектов `Product` с помощью библиотеки Jackson.

Возвращает: `List<Product>` – список всех товаров (с полями `id`, `name`, `price`, `recommendations`). В случае ошибки соединения возвращает пустой список. Код метода представлен в листинге 5 Приложения Г.

Метод `getProductById(String id)`.

Параметры: `id` – уникальный идентификатор товара.

Что делает: вызывает `getAllProducts()` для получения полного списка. Затем с помощью `Stream API` отфильтровывает товар, у которого `id` совпадает с переданным.

Возвращает: объект `Product` с найденным товаром; если товар не найден – `null`. Такой подход позволяет избежать дополнительного запроса к `1C` для каждого вызова. Код метода представлен в листинге 6 Приложения Г.

Метод `sendOrderTo1C(String jsonOrder)`.

Параметры: `jsonOrder` – строка в формате `JSON`, содержащая данные заказа.

Тип запроса к `1C`: `POST`.

Что делает: с помощью `RestTemplate.postForObject` отправляет `JSON` на URL `http://localhost:8080/shop/hs/v1/order`.

Возвращает: В случае успешного выполнения `POST`-запроса метод возвращает ответ `1C`, содержащий статус и номер заказа. При возникновении исключения метод возвращает `JSON` с признаком ошибки. Код метода представлен в листинге 7 Приложения Г. С полным кодом программы можно ознакомиться в `GitHub` [20].

Для создания интерфейса использовался шаблонизатор `Thymeleaf` и библиотека `Bootstrap`. Главной задачей было наглядно показать покупателю блок «С этим товаром покупают». Логика страницы настроена так, что если `1C` присылает для товара рассчитанные ассоциативные связи, сайт динамически отрисовывает их под основным описанием. Это позволяет пользователю быстро добавить в корзину рекомендованный товар, увеличивая общую сумму заказа.

Для того чтобы сайт работал быстро и не нагружал сервер `1C` лишними запросами, логика корзины реализована на стороне клиента с помощью `JavaScript`.

Хранение: выбранные товары сохраняются в памяти браузера (`localStorage`). Это позволяет покупателю перемещаться по сайту, не теряя выбранные позиции.

Оформление: реализован механизм гостевого заказа. Покупателю не нужно регистрироваться. Достаточно ввести имя и телефон.

3.3 Реализация пользовательского интерфейса

3.3.1 Реализация интерфейса в среде «1С: Предприятие 8.3»

Для административной части системы была использована платформа «1С:Предприятие 8.3», которая предоставляет готовую среду для построения командного интерфейса.

Общий вид начального окна интерфейса представлен на рисунке 24.

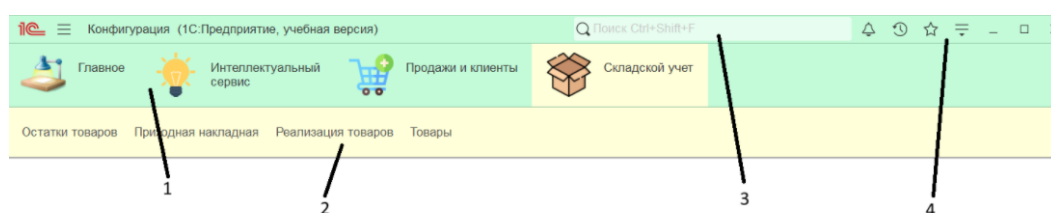


Рисунок 24 – Начальное окно интерфейса конфигурации

На рисунке представлены стандартные элементы управления 1С:

- 1 – панель разделов;
- 2 – область функций текущего разделов;
- 3 – строка глобального поиска;
- 4 – кнопки быстрого доступа к истории и избранному.

Для удобства работы менеджера весь функционал был разделен на логические блоки – подсистемы. Каждая подсистема объединяет объекты по их назначению.

«Складской учет» – предназначен для контроля запасов. Сюда были включены справочник «Товары», документы поступления и реализации, а также отчеты по остаткам. Интерфейс данной вкладки представлен на рисунке 25.

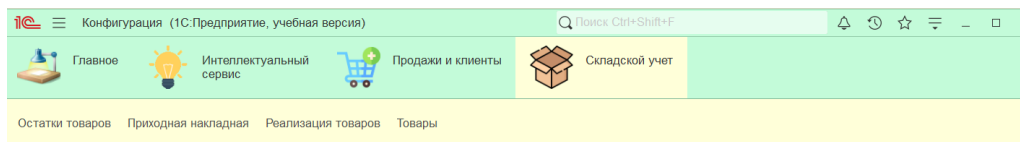


Рисунок 25 – Окно интерфейса подсистемы «Складской учет»

«Продажи и клиенты» – содержит базу контрагентов, сотрудников, регистры цен и документы по реализации товаров, а также функционал по созданию отчетов по продажам.

В подсистеме ведётся учёт контрагентов и сотрудников, хранятся текущие цены и фиксируются продажи. Документы реализации служат основой для анализа продаж и передачи данных в модуль машинного обучения. Также в этой части системы доступны отчёты, помогающие оценить объёмы продаж и популярность товаров. Интерфейс данной вкладки представлен на рисунке 26.

Эта подсистема была реализована как основное рабочее место менеджера, где собраны все необходимые инструменты для ежедневного обслуживания покупателей. Были объединены эти функции в один блок, чтобы сотруднику не приходилось переключаться между разными разделами программы при оформлении сделки. Именно здесь создается фундамент для всей аналитики. Чем больше реальных продаж будет зафиксировано в этой подсистеме, тем точнее станут рекомендации в будущем.

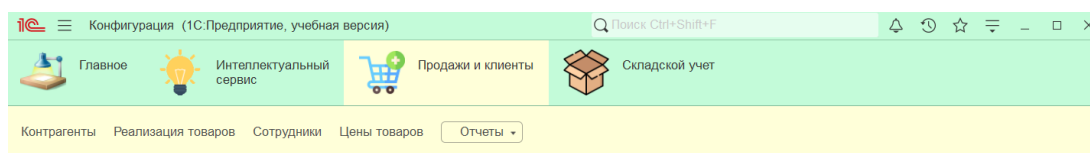


Рисунок 26 – Окно интерфейса подсистемы «Продажи и клиенты»

«Интеллектуальный сервис» – специализированный раздел для управления рекомендательной системой. Здесь хранятся результаты расчетов в регистре «Рекомендации товаров» и находится инструмент для запуска ML-модуля. Интерфейс данной вкладки представлен на рисунке 27.

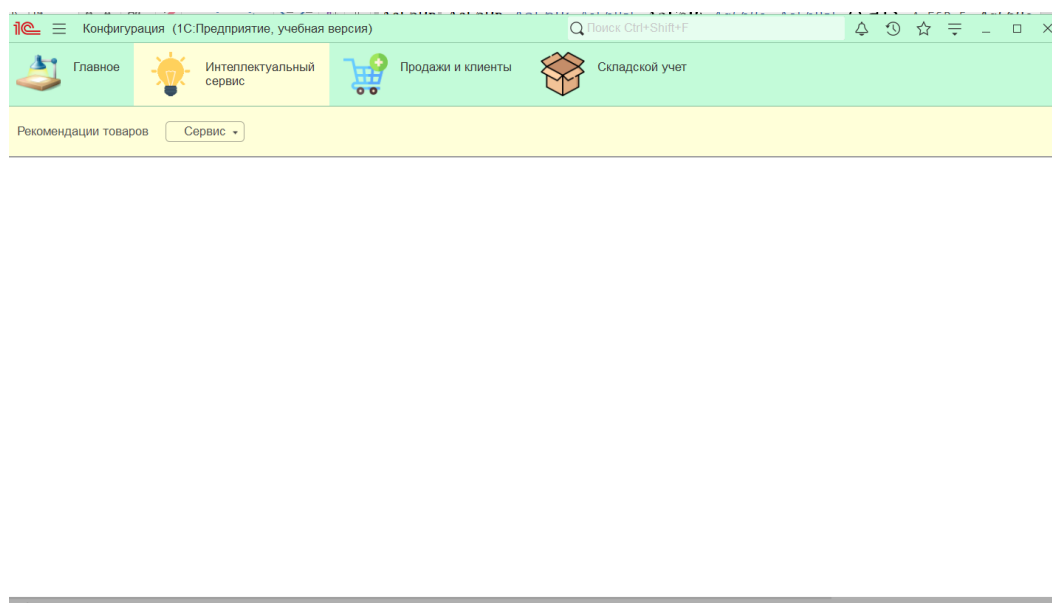


Рисунок 27 – Окно интерфейса подсистемы «Интеллектуальный сервис»

«Главное» – раздел для быстрого доступа к наиболее востребованным функциям системы.

Перейдя в конкретную подсистему, пользователь получает доступ к объектам (справочникам и документам). Все они оформлены по единому принципу, поэтому менеджеру не нужно заново разбираться в каждой новой форме. Это заметно ускоряет повседневную работу.

Например, интерфейс списка товаров представлен на рисунке 28.

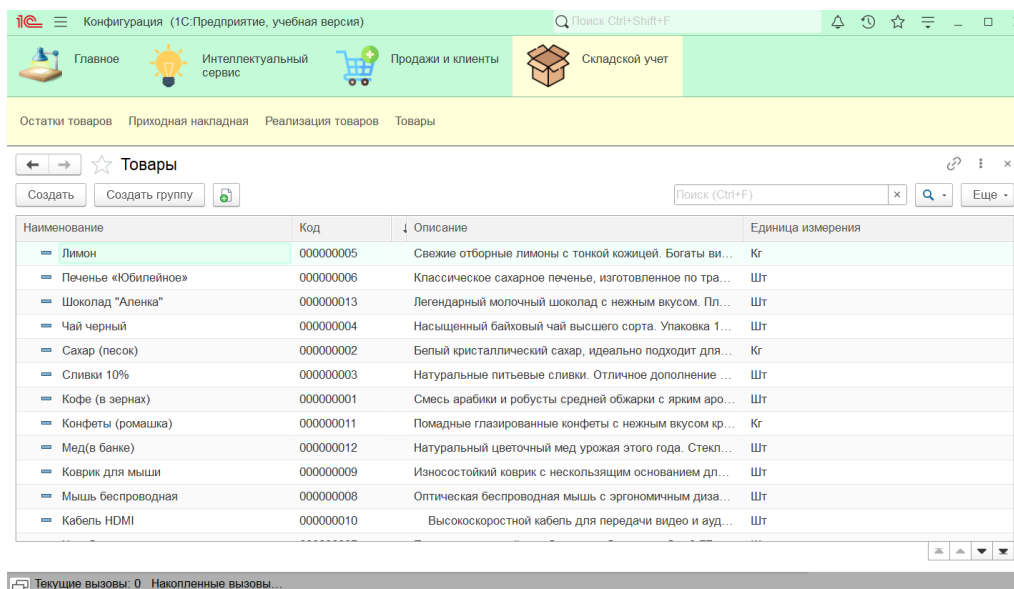


Рисунок 28 – Окно интерфейса справочника «Товары»

Данная форма позволяет менеджеру выполнять следующие действия:

- создавать и редактировать карточки номенклатуры;
- просматривать актуальные цены и описания;
- контролировать единицы измерения.

В списке товаров отображаются все позиции, заведённые в системе, с их основными реквизитами. Менеджер может открыть карточку любого товара двойным кликом мыши, чтобы просмотреть или изменить данные. Также в списке доступен быстрый переход к настройке цен и остатков. Это делает работу с номенклатурой удобной и наглядной.

Особое внимание на этапе реализации было уделено форме элемента справочника «Товары». Именно здесь менеджер может увидеть, какие товары алгоритм рекомендует друг к другу. Пример формы представлен на рисунке 29.

Товар	Рекомендованный товар	Коэффициент связи
Лимон	Сахар (песок)	0,24
Лимон	Чай черный	0,87

Рисунок 29 – Форма товара с блоком рекомендаций

Представленная форма содержит следующие элементы:

- поля основных данных – наименование, символьный код товара, описание товара;
- блок «Рекомендации товаров»: динамическая табличная часть, расположенная в нижней части формы.

Наличие этого блока позволяет менеджеру в режиме реального времени видеть, какие ассоциативные связи нашел алгоритм машинного обучения для данного товара. В таблице отображается наименование рекомендованного товара и рассчитанный коэффициент связи. Такое решение позволяет контролировать работу ML-модели прямо из карточки товара, не переходя в регистры.

При необходимости менеджер может сверить предложенные алгоритмом пары с реальной статистикой продаж. Это помогает вовремя заметить возможные ошибки в расчётах и принять решение о корректировке модели. Кроме того, отображение коэффициента связи даёт понимание, насколько сильной является рекомендация.

Для управления процессом обучения системы была реализована специализированная форма обработки «Интеллектуальный анализ», представленная на рисунке 30.

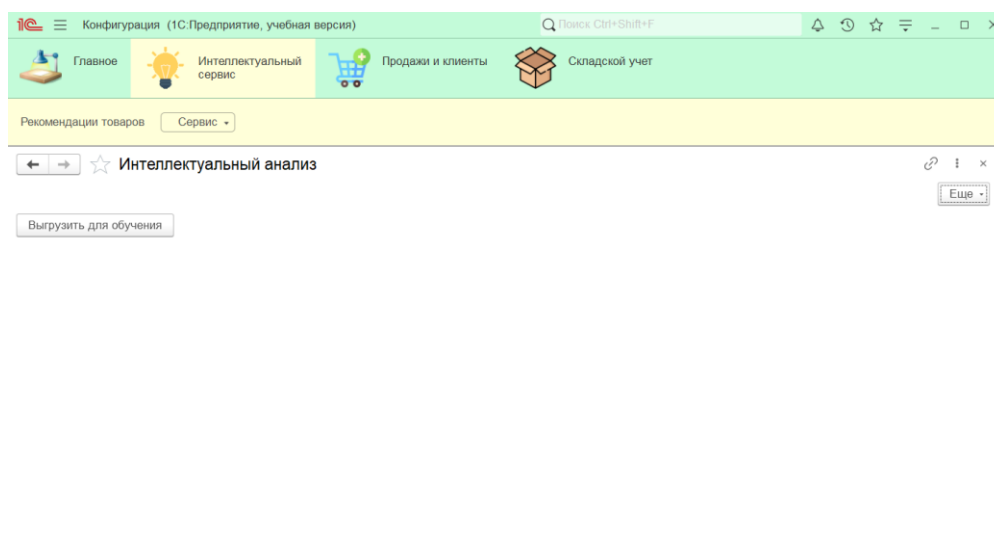


Рисунок 30 – Форма управления модулем машинного обучения

Интерфейс данной обработки максимально упрощен и работает по принципу одной кнопки. Это позволяет менеджеру, не имеющему специальных знаний в области Data Science, самостоятельно инициировать цикл актуализации рекомендаций.

3.3.2 Реализация интерфейса веб-приложения интернет-магазина

Интерфейс для покупателей реализован в виде веб-сайта на языке Java (Spring Boot). Это обеспечивает корректное отображение страниц на любых устройствах.

Главная страница сайта. Служит точкой входа для покупателя. На ней размещен приветственный баннер и кнопка для перехода в каталог товаров. Главная страница также содержит краткую информацию о преимуществах магазина, что помогает покупателю быстрее сориентироваться. Навигация по сайту интуитивно понятна и не требует дополнительных пояснений. Кнопка перехода в каталог выделена цветом, чтобы покупатель легко её нашёл. При реализации данной страницы основной акцент был сделан на лаконичности дизайна, что позволяет пользователю сразу сосредоточиться на выборе товаров. Интерфейс главной страницы представлен на рисунке 31.

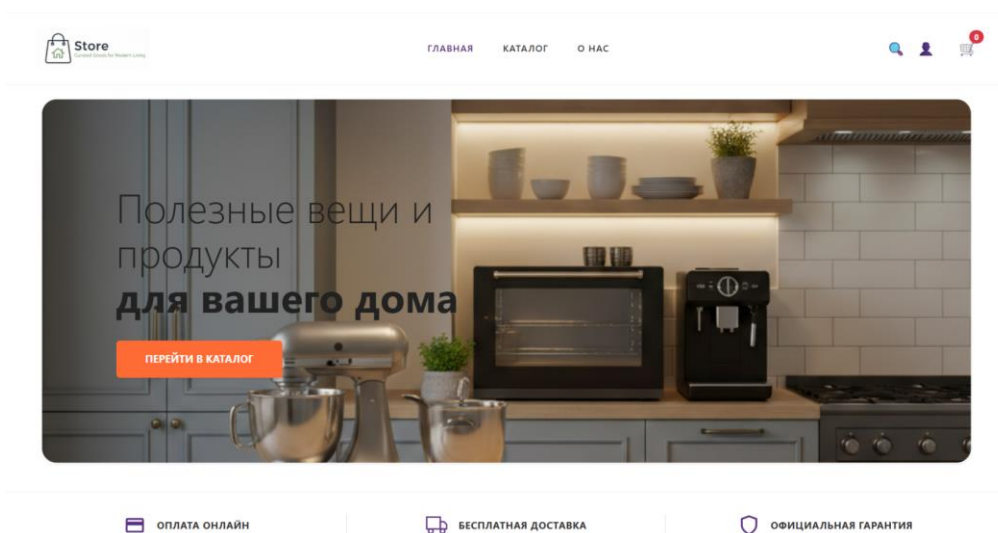


Рисунок 31 – Интерфейс главной страницы сайта

Динамический каталог товаров. Экран каталога выводит список всех позиций, которые есть в наличии на складе 1С. Каждая карточка содержит актуальную цену и кнопку перехода к детальному просмотру. Пример интерфейса каталога представлен на рисунке 32.

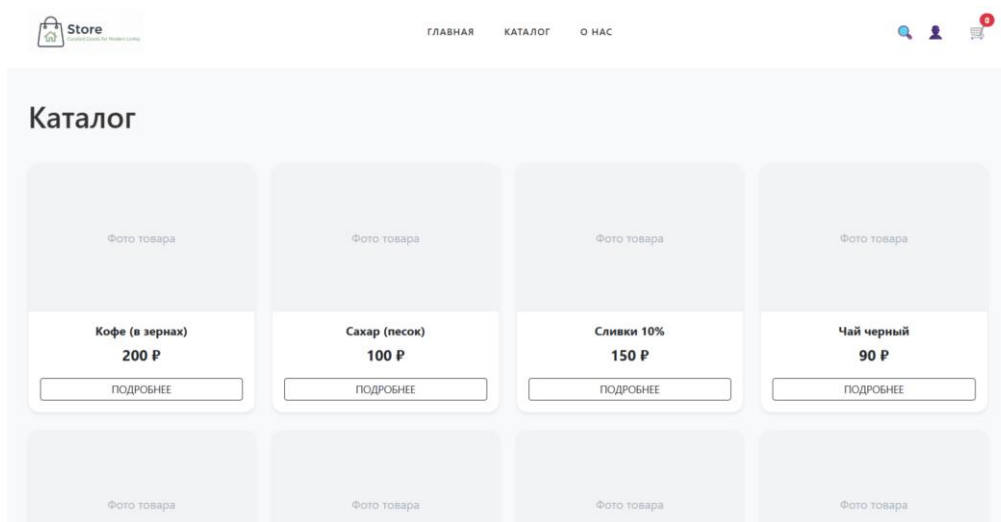


Рисунок 32 – Интерфейс каталога товаров на сайте

Карточка товара и блок рекомендаций. Это центральный элемент интерфейса. Под основным фото товара программно реализован блок «С этим товаром покупают».

Сайт автоматически запрашивает данные у 1С и отрисовывает рекомендации, которые рассчитал аналитический модуль Python. Пример интерфейса карточки товара представлен на рисунке 33.

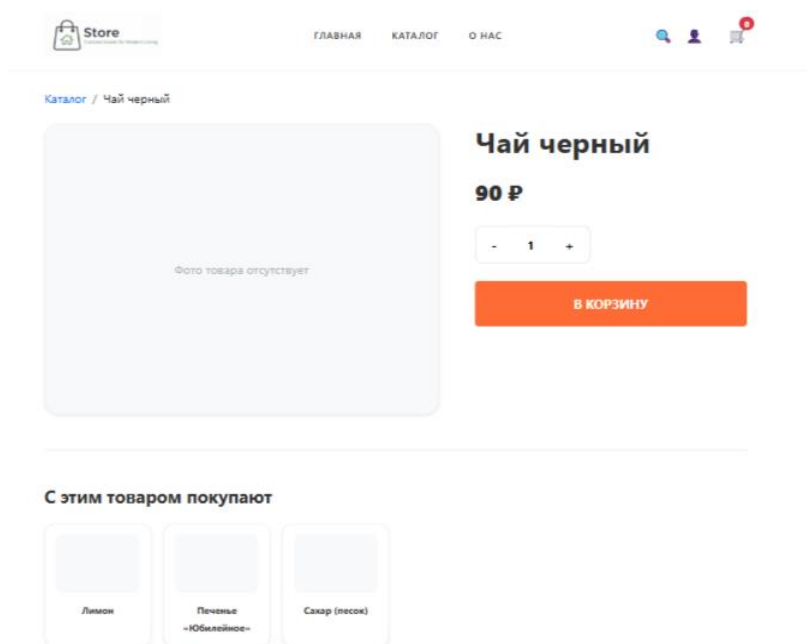


Рисунок 33 – Интерфейс карточки товара

Интерфейс окна реализованной корзины включает несколько основных элементов.

Табличная часть. Содержит наименования товаров, их количество и итоговую стоимость по каждой позиции.

Информационная панель. В нижней части окна крупным шрифтом выделена итоговая сумма всего заказа.

Управляющие кнопки. Реализованы кнопки «Очистить» для быстрого сброса всех позиций, «Закрыть» для закрытия окна и основная кнопка «Оформить» для возможности перехода к форме с оформлением заказа. При изменении количества товара в корзине итоговая сумма пересчитывается автоматически. Это позволяет покупателю видеть актуальную стоимость заказа сразу после добавления или удаления позиций. Пример формы корзины представлен на рисунке 34.

The screenshot shows a modal window titled 'Корзина' (Cart) with a close button (X) in the top right corner. Inside the window, there is a table with three columns: 'Товар' (Goods), 'Кол-во' (Quantity), and 'Сумма' (Sum). The table contains one row: 'Чай черный' (Black Tea) with a quantity of '2 шт.' (2 pieces) and a sum of '180 Р' (180 Rubles). Below the table, there is a section labeled 'Итого:' (Total:) with the value '180 Р'. At the bottom of the window, there are three buttons: 'Очистить' (Clear) in a red-outlined box, 'Закрыть' (Close) in a grey box, and 'Оформить' (Checkout) in a yellow box.

Товар	Кол-во	Сумма
Чай черный	2 шт.	180 Р

Итого: 180 Р

Очистить Закрыть Оформить

Рисунок 34 – Интерфейс корзины товаров

Для завершения процесса покупки реализована форма оформления заказа, представленная на рисунке 35. Она появляется в виде отдельного модального окна после нажатия кнопки оформления в корзине.

Интерфейс формы содержит текстовые поля ввода: «Ваше имя» и «Контактный телефон».

Так оформление заказа занимает минимум времени и не требует перехода на другие страницы сайта.

The screenshot shows a modal window titled 'Заказ' (Order) with a close button (X) in the top right corner. Inside the window, there are two text input fields. The first field contains the text 'Юлия' and the second field contains the text '+79272872070'. Below the input fields, there is a large blue button with the text 'ПОДТВЕРДИТЬ' (CONFIRM).

Заказ

Юлия

+79272872070

ПОДТВЕРДИТЬ

Рисунок 35 – Форма оформления заказа

Разработанный веб-интерфейс обеспечивает покупателю комфортное взаимодействия с системой товарных рекомендаций.

4 Контроль качества системы рекомендаций товаров

4.1 Обоснование выбора режима тестирования и метрик кода

Проверка качества – обязательная часть разработки. Моя система объединяет три технологии (1С, Python и Java), поэтому тестировать нужно не только каждую функцию по отдельности, но и то, как модули взаимодействуют друг с другом.

Для комплексной проверки системы были выбраны следующие методы:

Функциональное тестирование (классы эквивалентности и граничные условия) [23]. С помощью этого метода я проверю самый ответственный участок системы – передачу данных через CSV-файлы. Нужно убедиться, что Python-скрипт корректно читает информацию из 1С, даже если во входном файле встречаются ошибки, не тот разделитель или пустые поля.

Классы эквивалентности помогут выделить группы корректных и некорректных входных данных (например, правильное количество полей, неверный разделитель, пустые строки).

Граничные условия позволят проверить поведение модуля на краях допустимых значений. Например, при минимальном количестве строк в файле, , при пустом файле.

Этот вид тестирования показывает, как сайт ведёт себя, когда к нему одновременно обращаются много пользователей [24]. Магазин должен выдерживать наплыв покупателей, не замедляясь и не падая. Тестирование с помощью JMeter необходимо, чтобы понять, как быстро связка сайта и 1С выдает рекомендации при росте количества запросов. И где находится предел производительности текущего сервера.

Оценка метрик программного кода (SLOC, метрики Холстеда и Чепина). Эти методы выбраны для анализа качества кода на языке Python. Так как этот модуль отвечает за сложные математические расчеты, важно подтвердить, что программа написана аккуратно и без лишних усложнений.

Метрика SLOC. Это метод прямого подсчета количества строк в программе. С её помощью можно оценить реальный объем написанного модуля, снабжен ли код снабжен комментариями. Это помогает понять, не перегружен ли код лишними строками.

Метрика Холстеда. Основывается на анализе числа команд (операторов) и данных (операндов) в коде. Она позволяет рассчитать логическую сложность алгоритма и оценить усилия, затраченные на разработку. В данном проекте метрика помогает доказать, что благодаря библиотекам машинного обучения удалось реализовать сложную математику коротким и эффективным путем.

Метрика Чепина. Позволяет оценить сложность программы по тому, какую роль играют переменные в расчетах или управлении логикой. Это важно для подтверждения того, что модуль на Python имеет понятную структуру, в которой легко разобраться.

Такой набор методов позволяет оценить проект с разных сторон. Проверить точность алгоритмов, устойчивость системы под нагрузкой и чистоту написанного кода.

4.2 Функциональное тестирование

4.2.1 Классы эквивалентности

Для того чтобы аналитический модуль на Python мог корректно построить модель, входной файл `sales_data.csv` с данными продаж должен соответствовать установленному формату. Тестирование методом классов эквивалентности позволяет проверить, как система реагирует на корректные и некорректные данные в этом файле.

Правила для файла `sales_data.csv`:

- файл должен быть в кодировке UTF-8;
- разделителем должна быть запятая;
- в каждой строке должно быть ровно 2 поля – `order_id` и `item_name`;
- поля не должны быть пустыми.

Классы эквивалентности представлены в таблице 20.

Таблица 20 – Классы эквивалентности

Критерий	Правильные КЭ	Неправильные КЭ
Структура строки	2 поля через запятую (1)	1 поле (2); Более 2-х полей (3)
Разделитель	Запятая (4)	Точка с запятой (5); Табуляция (6)
Заполнение полей	Оба поля заполнены (7)	Пустой order_id (8); Пустой item_name (9)
Наличие данных	В файле есть строки (10)	Файл содержит только заголовок (11)
Кодировка текста	UTF-8 (12)	Windows-1251 или другая (13)
Заголовки полей	order_id,item_name (14)	Заголовки отсутствуют или изменены (15)

Выделенные классы эквивалентности служат фундаментом для проведения практических проверок системы. На основе этих классов подготовлены детальные тест-кейсы, представленные в таблице 21.

Таблица 21 – Тест-кейсы для проверки интеграционного файла

№	КЭ	Входные данные	Выходные данные
1	1, 4, 7, 10, 12, 14, 16	00000001,Лимон	Успешно. Файл полностью корректен, запущен расчет модели.
2	2	Лимон	Ошибка. Строка содержит только один столбец.
3	3	00000001,Лимон,30.00	Ошибка. Обнаружено 3 поля вместо 2. Строка не принята в расчет.
4	5	00000001;Лимон	Ошибка. Разделитель (;) не распознан, данные слились в одну строку.
5	6	00000001 Лимон(через Tab)	Ошибка. Табуляция не является стандартным разделителем для данного процесса.

Продолжение таблицы 21

№	КЭ	Входные данные	Выходные данные
6	8	,Лимон	Ошибка. Отсутствует идентификатор заказа.
7	9	00000001,	Ошибка. Пустое название товара.
8	11	order_id,item_name (больше ничего)	Ошибка. Система сообщает о нехватке данных для обучения модели.
9	13	00000001, Лимон (кодировка Windows-1251)	Ошибка. Неверная кодировка файла
10	15	ID_ZAKAZA,TOVAR 000000002,Лимон	Ошибка. Неверный формат столбцов.

Разработанные классы эквивалентности и тест-кейсы покрывают все возможные сценарии отклонения входного CSV-файла от заданного формата. А именно неверное количество полей, неправильный разделитель, пустые значения, недопустимую кодировку, отсутствие данных или неверные заголовки.

4.2.2 Граничные условия

Метод граничных условий позволяет проверить, как ведет себя система при достижении предельных значений данных. Для модуля машинного обучения важно убедиться, что программа не «зависнет» при большом объеме продаж и корректно обработает минимальные наборы данных. Крайние значения входных данных могут возникать в реальной работе системы, поэтому их проверка обязательна. Например, это могут быть единичные чеки или, наоборот, массивы данных за несколько лет. Тестирование в таких условиях показывает, насколько надёжно работает аналитический модуль в нестандартных ситуациях.

Объектами тестирования являются: входной файл истории продаж (sales_data.csv), процесс построения разреженной матрицы. Выделенные граничные условия представлены в таблице 22.

Таблица 22 – Граничные условия

Границы	Условия
Логические	Минимально допустимый объем входного файла (Л1)
Логические	Минимальное количество товаров в одном чеке для поиска связи (Л2)
Логические	Минимальный набор данных для расчетов (2 товара в базе) (Л3)
Технологические	Максимальное количество отображаемых рекомендаций на сайте (Т1)
Технологические	Рост номенклатуры до 10 000+ позиций (нагрузка на память)

Для практической проверки того, как система справляется с достижением этих границ, были составлены тест-кейсы, представленные в таблице 23. Это позволяет на конкретных примерах увидеть реакцию алгоритма на минимальные и максимальные значения.

Таблица 23 – Тест-кейсы граничных условий

№	Границы	Входные данные	Выходные данные
1	Л1	В файле одна строка данных (например: 000000001, Товар1)	Успешно. Модель загружается, но так как пара для товара отсутствует, файл рекомендаций будет пустым. Критических ошибок нет.
2	Л2	В базе 100 чеков, но в каждом только по 1 товару	Успешно. Алгоритм не находит пар, список рекомендаций пуст.
3	Т1	Модуль рассчитал 10 рекомендованных товаров для ноутбука	Успешно. На сайте отображаются только первые 3 (Тор-3).
4	Л3	В файле всего 2 товара и 1 общий чек	Успешно. Модель успешно находит связь и выдает коэффициент сходства (минимальный порог данных).
5	Т2	Загрузка данных о 15 000 товарах и 50 000 продажах	Успешно. Благодаря использованию разреженных матриц (CSR), расчет проходит без перегрузки оперативной памяти.

Проведенное тестирование по методам классов эквивалентности и граничных условий подтвердило стабильность работы разрабатываемой системы. Система справляется с нагрузкой при росте ассортимента и защищена от ошибок ввода.

4.3 Нефункциональное тестирование

В качестве нефункционального тестирования было выбрано нагрузочное тестирование. Оно проверяет работоспособность системы при заданной нагрузке. С его помощью можно понять, сможет ли система за определенное время обработать желаемое количество запросов.

В ходе тестирования оценивалось, как веб-сервер и интеграция с 1С выдерживают нагрузку при запросах к странице каталога. Этот сценарий наиболее требователен, так как требует формирования списка товаров с рекомендациями, рассчитанными Python-модулем и переданными через 1С.

Для создания нагрузки использовался Apache JMeter – инструмент с поддержкой имитации HTTP-запросов, измерения времени ответа и подсчёта пропускной способности.

JMeter делал одновременные GET-запросы к URL `http://localhost:5000/catalog`. Количество виртуальных пользователей последовательно увеличивалось: 50, 100, 200 и 500. Каждый пользователь выполнял ровно один запрос. Такой подход позволяет постепенно наращивать нагрузку и наблюдать за изменением времени ответа. При этом фиксируется не только среднее время, но и максимальное, а также количество успешных запросов. Это даёт полную картину производительности системы при разных уровнях одновременной работы пользователей. Время нарастания нагрузки составляло 5 секунд, чтобы пользователи подключались постепенно.

Результаты тестирования позволят выявить, при каком количестве пользователей время отклика остаётся приемлемым, а когда начинаются заметные задержки.

Результаты тестирования при нагрузке в 50 пользователей представлены на рисунке 36.

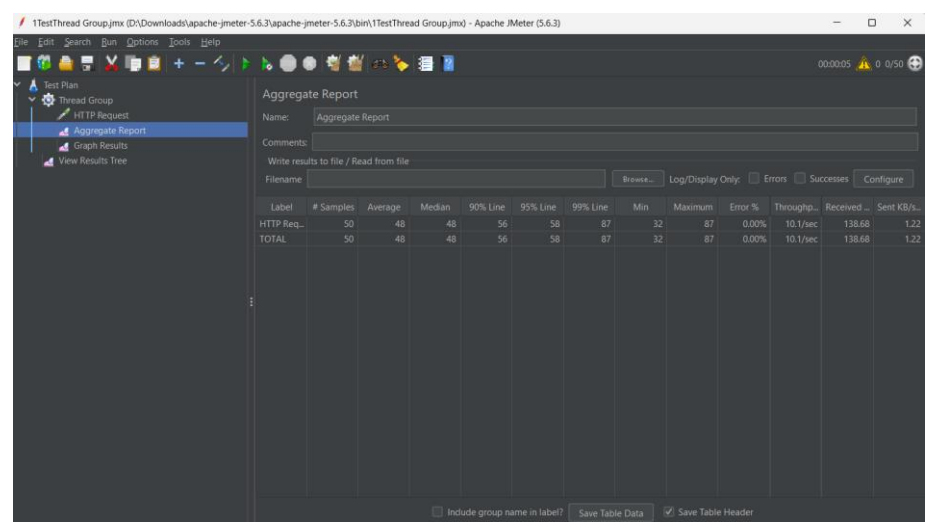


Рисунок 36 – Результаты нагрузочного тестирования при 50 пользователях

Анализ данных на рисунке 36 показывает высокую производительность: среднее время ответа составляет всего 48 мс, а ошибки полностью отсутствуют. На следующем шаге нагрузка была увеличена в два раза. Результаты тестирования при 100 одновременных пользователях представлены на рисунке 37.

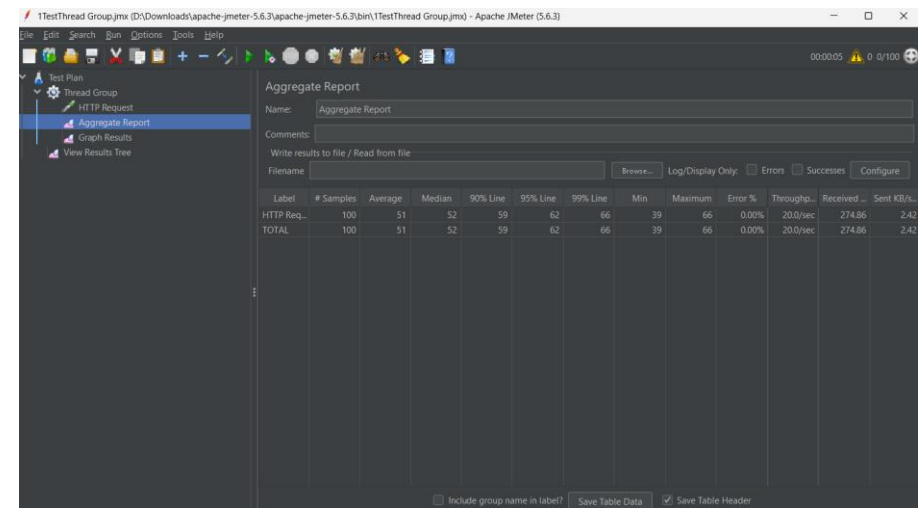


Рисунок 37 – Результаты нагрузочного тестирования при 100 пользователях

При 100 пользователях система также демонстрирует стабильно высокие показатели, сохраняя среднее время ответа в пределах 51 мс без возникновения

ошибок. Для определения пределов производительности системы нагрузка была увеличена до 200 пользователей. Соответствующие результаты приведены на рисунке 38.

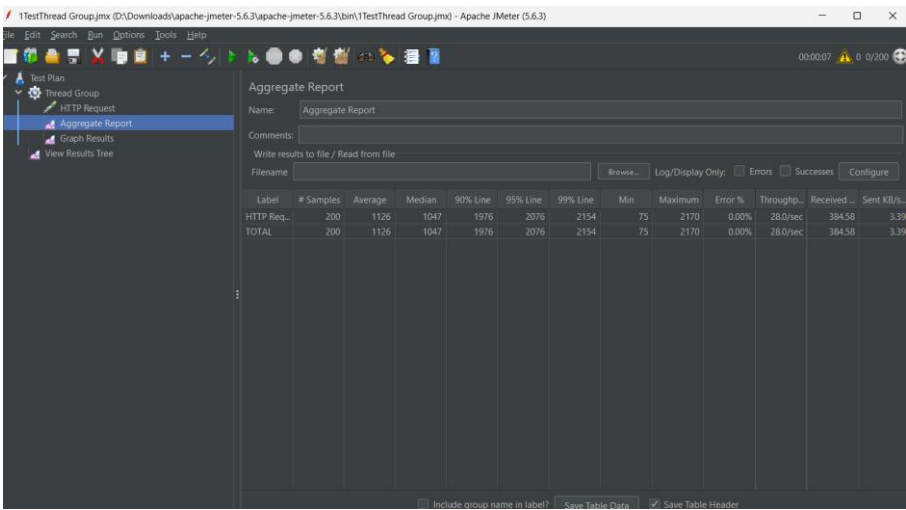


Рисунок 38 – Результаты нагрузочного тестирования при 200 пользователях

Данные на рисунке 38 указывают на то, что при 200 пользователях среднее время ответа возрастает до 1126 мс, а пропускная способность увеличивается незначительно (с 20 до 28 запросов/сек). Для оценки поведения разрабатываемой системы в условиях пиковой перегрузки был проведен финальный тест при 500 пользователях. Результаты которого отражены на рисунке 39.

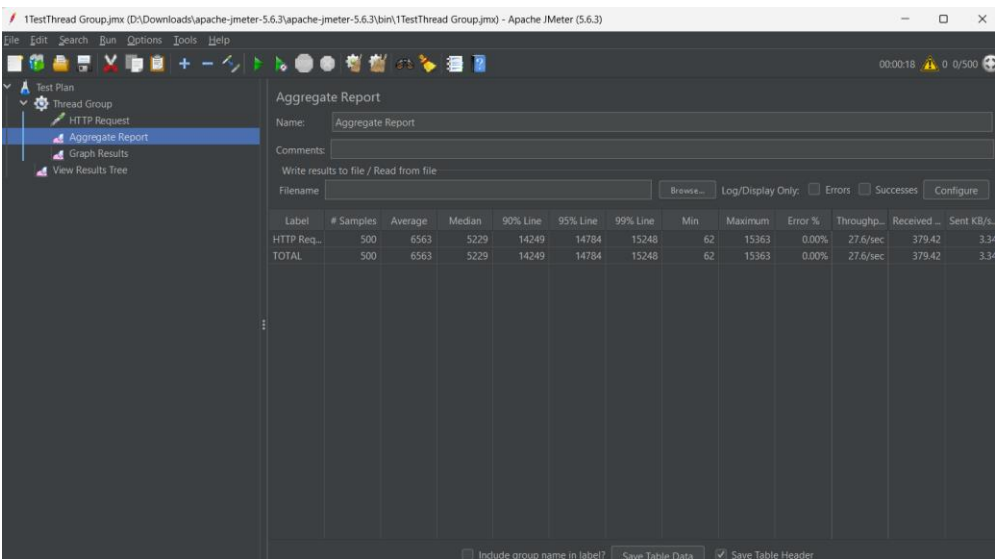


Рисунок 39 – Результаты нагрузочного тестирования при 500 пользователях

При нагрузке 50 и 100 пользователей система показывает высокую производительность: среднее время ответа составляет 48 мс и 51 мс соответственно, ошибки отсутствуют.

При увеличении нагрузки до 200 пользователей среднее время ответа возрастает до 1126 мс, а пропускная способность увеличивается незначительно (с 20 до 28 запросов/сек). Это свидетельствует о том, что узким местом становится обработка параллельных запросов на стороне сервера (вероятнее всего, вызовы HTTP-сервиса 1С без кэширования).

При нагрузке 500 пользователей среднее время ответа достигает 6563 мс, однако ошибки по-прежнему отсутствуют. Система сохраняет стабильность (нет сбоев), но время отклика превышает допустимые для интернет-магазина значения.

Общий вывод: разработанная система устойчива к нагрузке и не выдаёт ошибок даже при 500 одновременных пользователях.

Но в реальной эксплуатации интернет-магазина нужно ускорить работу сервера. Проще всего это сделать, добавив кэширование списка товаров. Тогда сайт не будет каждый раз запрашивать одни и те же данные у 1С. Такая доработка снизит нагрузку на учётную систему и сократит время ожидания при нагрузке выше 100 пользователей.

4.4 Метрики кода

4.4.1 Метрика SLOC

SLOC – это метрика, которая измеряет размер программного кода путём подсчёта количества строк в исходном файле. Она позволяет оценить трудоёмкость разработки, сложность восприятия и поддержки модуля.

Для анализа был выбран основной скрипт `ml_model.py`, который выполняет загрузку данных из файла `sales_data.csv`, строит разреженную матрицу и вычисляет косинусное сходство товаров. Код алгоритма представлен в листинге 8.

Листинг 8 – Программный код ml_model.py

```

import sys
import io
import pandas as pd
from sklearn.metrics.pairwise import cosine_similarity
from scipy import sparse

# Настройка кодировки для вывода в консоль 1C
sys.stdout = io.TextIOWrapper(sys.stdout.buffer, encoding='utf-8')

try:
    # 1. Загрузка данных из 1C
    df = pd.read_csv('D:/temp/sales_data.csv')
    print("Файл sales_data.csv успешно прочитан.")

    # 2. Подготовка разреженной матрицы (CSR)
    # Преобразование названия товаров и заказов в числовые
    категории
    df['order_id'] = df['order_id'].astype('category')
    df['item_name'] = df['item_name'].astype('category')

    # Создание матрицы: строки - заказы, столбцы - товары
    contact_matrix = sparse.csr_matrix((
        [1] * len(df),
        (df['order_id'].cat.codes, df['item_name'].cat.codes)
    ))

    print("Разреженная матрица построена. Начинаю расчет
    косинусного сходства.")

    # 3. Расчет ML-модели
    item_similarity = cosine_similarity(contact_matrix.T)

    # Перевод результат обратно в таблицу Pandas
    item_names = df['item_name'].cat.categories
    item_similarity_df = pd.DataFrame(item_similarity,
    index=item_names, columns=item_names)

    # 4. Сбор рекомендаций (Топ-3 для каждого товара)
    recommendations = []
    for item in item_similarity_df.columns:
        similar_items =
    item_similarity_df[item].sort_values(ascending=False).drop(item)

        for rec_item, score in similar_items.head(3).items():
            if score > 0:
                recommendations.append([item, rec_item,
    round(score, 2)])

    # 5. Сохранение результата для 1C
    result_df = pd.DataFrame(recommendations,
    columns=['antecedents', 'consequents', 'confidence'])

```

```
result_df.to_csv('D:/temp/recommendations.csv', index=False,
encoding='utf-8-sig')
```

```
print("\n--- Успех! ---")
print(f"Обработано уникальных товаров: {len(item_names)}")
print(f"Создано правил рекомендаций: {len(result_df)}")
```

```
except Exception as e:
```

```
print(f"Критическая ошибка в работе ML-модуля: {e}")
```

Результаты тестирования приведены в таблице 24.

Таблица 24 – Количественные метрики кода аналитического модуля

Характеристика	Значение	Комментарий
SLOC (Physical)	53	Полный объем исходного кода Python-скрипта, включая пустые строки и строки с комментариями.
SLOC (Logical)	32	Количество значимых операций (импорты, присваивания, вызовы функций, циклы, условные операторы).
Количество пустых строк	12	Строки, не содержащие кода или комментариев (разделители логических блоков).
Количество строк с комментариями	9	Пояснения к ключевым этапам
Процент комментариев	17,0%	Степень наглядности кода, достаточная для быстрой поддержки модуля другими разработчиками.
Среднее число строк для функций	53	Весь скрипт представляет собой единую процедуру без выделенных функций.
Среднее число строк исходного кода	41	Количество строк, содержащих непосредственно код (Physical SLOC минус пустые строки).

Метрика SLOC показывает размер программного модуля и его документированность. Для скрипта ml_model.py получены следующие значения: 53 физических строк, 32 логических строк, 9 строк с комментариями (17,0% от

общего объёма). Такие показатели означают, что код имеет небольшую сложность. Его структура проста, алгоритм легко прослеживается от начала до конца. Наличие комментариев позволяет другому разработчику быстро разобраться в назначении модуля и при необходимости внести изменения. Компактность кода и наличие пояснений обеспечивают хорошую поддерживаемость. Таким образом, по метрике SLOC скрипт соответствует принципам чистого и хорошо документированного кода.

4.4.2 Метрики Холстеда

Метрика Холстеда позволяет оценить, насколько сложна программа с точки зрения используемых в ней действий и данных. Для этого весь код разбивается на операторы (команды, ключевые слова, знаки операций) и операнды (переменные, константы, строки). На основе их подсчёта вычисляются характеристики: объём программы, сложность и примерные усилия, которые потребуются разработчику для понимания или изменения кода. Чем меньше эти значения, тем проще поддерживать программу.

На основе листинга `ml_model.py` были подсчитаны вхождения операторов и операндов. Перечень операндов и операторов приведены в таблице 25.

Таблица 25 – Операторы и операнды скрипта `ml_model`

Операторы	Количество	Операнды	Количество
<code>import</code>	3	<code>sys</code>	3
<code>from</code>	2	<code>io</code>	2
<code>as</code>	1	<code>pd</code>	2
<code>try</code>	1	<code>sparse</code>	2
<code>except</code>	1	<code>df</code>	5
<code>for</code>	2	<code>contact_matrix</code>	2
<code>in</code>	2	<code>item_similarity</code>	2
<code>if</code>	1	<code>item_names</code>	2
<code>print</code>	4	<code>item_similarity_df</code>	3
<code>exit</code>	1	<code>recommendations</code>	3
<code>sys.stdout</code>	1	<code>item</code>	3
<code>TextIOWrapper</code>	1	<code>similar_items</code>	2
<code>read_csv</code>	1	<code>rec_item</code>	1
<code>astype</code>	2	<code>score</code>	2
<code>csr_matrix</code>	1	<code>result_df</code>	2
<code>len</code>	1	<code>e</code>	2
<code>cat.codes</code>	2	<code>'utf-8'</code>	1

Продолжение таблицы 25

Операторы	Количество	Операнды	Количество
cat.categories	1	'D:/temp/sales_data.csv'	1
.T (транспонирование)	1	'category'	2
cosine_similarity	1	1	2
sort_values	1	3	1
drop	1	0	1
head	1	2	2
items	1	0.05	1
round	1	False	1
DataFrame	2	'D:/temp/recommendations.csv'	1
to_csv	1	'utf-8-sig'	1
= (присваивание)	12	'antecedents'	1
. (доступ к методу)	18	'consequents'	1
() (скобки вызова)	21	'confidence'	1
[] (индексация)	11		
: (начало блока)	4		
, (разделитель)	15		
* (умножение)	1		
> (сравнение)	1		

Сбор базовых показателей.

n1 – Число уникальных операторов: включает ключевые слова Python, знаки препинания, операторы доступа и вызовы методов.

Операторы: import, 2. from, 3. as, 4. try, 5. except, 6. for, 7. in, 8. if, 9. print, 10. exit, 11. sys.stdout, 12. TextIOWrapper, 13. read_csv, 14. astype, 15. csr_matrix, 16. len, 17. cat.codes, 18. cat.categories, 19. .T, 20. cosine_similarity, 21. sort_values, 22. drop, 23. head, 24. items, 25. round, 26. DataFrame, 27. to_csv, 28. =, 29. ., 30. (), 31. [], 32. :, 33. ,, 34. *, 35. >.

Итого операторов n1 = 35.

n2 – Число уникальных операндов: сюда входят названия библиотек, переменных и константы.

Операнды: 1.sys, 2. io, 3. pd, 4. sparse, 5. df, 6. contact_matrix, 7. item_similarity, 8. item_names, 9. item_similarity_df, 10. recommendations, 11. item, 12. similar_items, 13. rec_item, 14. score, 15. result_df, 16. e, 17. 'utf-8', 18. 'D:/temp/sales_data.csv', 19. 'category', 20. 1, 21. 3, 22. 0, 23. 2, 24. 0.05, 25. False, 26. 'D:/temp/recommendations.csv', 27. 'utf-8-sig', 28. 'antecedents', 29. 'consequents', 30. 'confidence'.

Итого операндов $n_2 = 30$.

N_1 - Общее число операторов в программе: все упоминания символов и ключевых слов из списка n_1 .

$N_1 = 120$.

N_2 - Общее число операндов в программе: считаем все упоминания имен переменных и библиотек из списка n_2 .

$N_2 = 61$.

Расчет основных метрик.

Словарь программы (n) вычисляется по формуле:

$$n = n_1 + n_2 = 35 + 30 = 65,$$

где n – словарь программы;

n_1 – число уникальных операторов;

n_2 – число уникальных операндов.

Длина программы (N) вычисляется по формуле:

$$N = N_1 + N_2 = 120 + 61 = 181,$$

где N – длина программы;

N_1 – общее число операторов.

N_2 – общее число операндов.

Теоретическая длина программы (N'):

$$N' = n_1 \times \log_2 n_1 + n_2 \times \log_2 n_2 = 35 \times 5.13 + 30 \times 4.9 = 326.55,$$

где N' – теоретическая длина программы;

n_1 – число уникальных операторов;

n_2 – число уникальных операндов.

Объем программы (V):

$$V = N \times \log_2 n = 181 \times 6.02 = 1089.62,$$

где V – информационный объем программы;

N – длины программы;

n – словарь программы.

Уровень качества программирования (L'):

$$L' = \frac{2 \times n_2}{n_1 \times N_2} = \frac{2 \times 30}{35 \times 61} = 0.0281 ,$$

где L' – оценка уровня программы;

n_1 – число уникальных операторов;

n_2 – число уникальных операндов;

N_2 – общее число операндов.

Трудоемкость кодирования программы (D):

$$D = \frac{1}{L'} = \frac{1}{0.0281} = 35.58,$$

где D – трудоемкость кодирования программы;

L' – оценка уровня качества программы.

Информационное содержание программы (I):

$$I = \frac{V}{D} = \frac{1089.62}{35.58} = 30.62,$$

где I – информационное содержание программы;

V – информационный объем программы;

D – трудоемкость кодирования программы.

Уровень языка выражения (y'):

$$y' = \frac{V}{D^2} = \frac{1089.62}{35.58^2} = 30.62 ,$$

где y' – уровень языка выражения;

V – информационный объем программы;

D – трудоемкость кодирования программы.

Интеллектуальные усилия при разработке (E)

$$E = N' \times \log_2 \frac{n}{L'} = 326.55 \times 11.17 = 3647.56,$$

где E – интеллектуальные усилия при разработке;

N' – теоретическая длина программы;

n – словарь программы;

L' – оценка уровня программы.

Результаты расчета метрик Холстеда систематизированы в таблице 26.

Таблица 26 – Сводные результаты оценки сложности по методике Холстеда

№	Наименование метрики	Обозначение	Значение
1	Словарь программы	n	65
2	Реальная длина программы	N	181
3	Теоретическая длина программы	N'	326,55
4	Объем программы (бит)	V	1089,62
5	Уровень качества программирования	L'	0,0281
6	Трудоемкость кодирования (сложность)	D	35,58
7	Информационное содержание	I	30,62
8	Уровень языка выражения	y'	0,86
9	Интеллектуальные усилия при разработке	E	3647,56

Рассчитанные показатели позволяют оценить качество написанного программного кода и сложность его дальнейшей поддержки. На основе полученных данных можно сделать следующие выводы:

Информационная содержание программы ($I=30.62$). Данный показатель подтверждает высокую содержательность кода. Это означает, что модуль не перегружен лишними командами, нет дублирующих операций. Каждая строка кода выполняет конкретную функцию в процессе анализа данных. Поэтому программа короткая и рабочая.

Трудоемкость кодирования ($D=35.58$). Показатель говорит о том, насколько сложно было писать и проверять код. Это значит, что алгоритм не перегружен деталями, в нем сложно случайно ошибиться и его легко будет поддерживать в будущем.

Эффективность использования языка ($y'=0.86$). Значение уровня языка показывает, что язык Python в данном проекте выступает как связующее звено.

Вся сложная математика (разреженные матрицы, косинусное сходство) отдана Pandas, SciPy и Scikit-learn. Вместо написания сложных функций с нуля, используются проверенные инструменты.

Интеллектуальные усилия ($E=3647.56$). Это довольно низкий показатель. Он говорит о том, что подход выбран верно. Использование готовых библиотек машинного обучения позволило реализовать сложный алгоритм косинусного сходства при минимальном объеме собственного кода. Это значительно сократило время на разработку.

Метрики Холстеда подтверждают, что аналитический модуль спроектирован хорошо и оптимально.

4.4.3 Метрика Чепина

Метрика Чепина показывает, насколько сложен модуль – для этого анализируется, как переменные участвуют в расчётах и влияют на ход выполнения программы.

Классификация переменных кода:

а) Р (вводимые): переменные, которые поступают извне для работы алгоритма:

- 1) df – исходная таблица из файла sales_data.csv;

Итого $P = 1$.

б) М (модифицируемые): переменные, которые создаются или изменяются в процессе выполнения скрипта:

- 1) contact_matrix – сформированная разреженная матрица;
- 2) item_similarity – вычисленная матрица косинусного сходства;
- 3) item_names – извлеченный список уникальных названий товаров;
- 4) item_similarity_df – результирующая таблица с индексами;
- 5) recommendations – список для сбора итоговых пар;
- 6) similar_items – промежуточная выборка для сортировки;
- 7) result_df – финальный объект для экспорта в CSV;

Итого $M = 7$.

с) С (управляющие): переменные, от которых напрямую зависят ветвления и логика циклов:

- 1) `item` – итератор внешнего цикла;
- 2) `rec_item` – итератор внутреннего цикла;
- 3) `score` – коэффициент, участвующий в условии `if score > 0`;
- 4) `e` – переменная для управления логикой в блоке `except`;

Итого $C = 4$.

d) Т (паразитные): неиспользуемые переменные, объявленные, но не задействованные в логике.

В коде таких нет.

Итого $T = 0$.

Классификация переменных представлена в таблице 27.

Таблица 27 – Классификация переменных по методике Чепина

Группа	Характеристика роли переменных	Список переменных в коде	Кол-во
Р	Вводимые переменные: поступают извне для работы алгоритма.	<code>df</code> (данные из CSV)	1
М	Модифицируемые переменные: создаются или изменяются в процессе вычислений.	<code>contact_matrix</code> , <code>item_similarity</code> , <code>item_names</code> , <code>item_similarity_df</code> , <code>recommendations</code> , <code>similar_items</code> , <code>result_df</code>	7
С	Управляющие переменные: определяют логику циклов и условий.	<code>item</code> , <code>rec_item</code> , <code>score</code> , <code>e</code> (блок исключений)	4
Т	Паразитные переменные: не используются в расчетах (отсутствуют).	-	0

Таким образом, проведенный анализ по методике Чепина подтверждает отсутствие в программе лишних (паразитных) переменных. Это говорит о чистоте написанного кода и о том, что каждая объявленная переменная выполняет свою задачу в процессе работы алгоритма.

Расчет метрики Чепина (Q).

Расчет производится по формуле:

$$Q=1\times P+2\times M+3\times C+0,5\times T,$$

где Q – показатель информационной сложности программного модуля;

P – количество вводимых переменных;

M – количество модифицируемых переменных;

C – количество управляющих переменных;

T – количество паразитных (неиспользуемых) переменных.

Итоговый расчет значения по формуле дает следующий результат:

$$Q=1\cdot P+2\cdot M+3\cdot C+0.5\cdot T=1\cdot 1+2\cdot 7+3\cdot 4+0.5\cdot 0=1+14+12=27$$

Значение метрики Q=27 говорит об умеренной сложности разработанного модуля. Основную нагрузку в коде несут модифицируемые переменные (M=7). Это полностью оправдано для аналитической задачи. Это подтверждает, что программа выполняет серьезное преобразование. Превращает обычный список продаж из 1С в систему рекомендаций. При этом модуль сохраняет понятную структуру.

Заключение

В ходе выполнения выпускной квалификационной работы был проведён анализ сферы интернет-торговли и изучены существующие способы создания товарных рекомендаций. Сформулированные на первом этапе требования позволили разработать программную систему, способную автоматически подбирать рекомендованные товары на основе реальной истории продаж.

В рамках этапа проектирования было выполнено моделирование архитектуры системы. Для описания логики и структуры программного комплекса был разработан пакет диаграмм, включающий:

- функциональные модели в нотации IDEF0 (контекстная и детализирующая);
- логическую модель базы данных (ER-диаграмма);
- uml-диаграммы: вариантов использования, деятельности, последовательности и развёртывания.

Это позволило настроить взаимодействие между тремя средами: платформой «1С:Предприятие 8.3», аналитическим модулем на языке Python и веб-интерфейсом на языке Java.

Итогом работы стала готовая информационная система, полностью автоматизирующая цикл формирования рекомендаций. Основные результаты реализации:

- учётная база данных в 1С, которая собирает историю продаж и передает данные на сайт через собственный программный интерфейс ShopAPI (REST API);
- интеллектуальный модуль на Python, реализующий алгоритм машинного обучения (коллаборативная фильтрация);
- интерактивный веб-сайт на базе Spring Boot, где покупатель может пользоваться удобным каталогом, корзиной и видеть блок «С этим товаром покупают».

Для проверки качества работы системы была проведена серия тестов. С помощью методов классов эквивалентности и граничных условий удалось подтвердить, что программа правильно обрабатывает входящие данные и без ошибок выполняет логику подбора товаров.

С помощью инструмента Apache JMeter было выполнено нагрузочное тестирование. Оно помогло определить возможности системы при большом количестве одновременных запросов. Тесты подтвердили, что связка сайта и 1С работает стабильно при нагрузках, характерных для малых и средних магазинов.

Математическая оценка кода по метрикам SLOC, Холстеда и Чепина показала, что аналитический модуль на Python написан эффективно. Использование специализированных библиотек (Pandas, SciPy, Scikit-learn) позволило реализовать сложные алгоритмы машинного обучения в компактном и понятном виде.

В итоге разработанная информационная система полностью решает поставленные задачи. Её внедрение позволит магазину отказаться от ручного подбора рекомендованных товаров и автоматизировать процесс выдачи рекомендаций. Это поможет повысить эффективность продаж.

Список использованных источников

1. Рекомендуемые товары [Электронный ресурс]. URL: https://dev.1c-bitrix.ru/user_help/components/magazin/information_tovars/catalog_recommended_products.php (дата обращения: 17.05.2026).
2. RetailRocket - AI-персонализация для e-commerce | Обзор, цены 2026 [Электронный ресурс]. URL: <https://toolfox.ru/services/s/retailrocket> (дата обращения: 17.05.2026).
3. UML диаграммы: виды, примеры, проектирование и моделирование [Электронный ресурс]. URL: <https://projecto.pro/blog/theory/uml-diagrammy/> (дата обращения: 17.05.2026).
4. Python для анализа данных: почему большинство аналитиков выбирают его [Электронный ресурс]. URL: <https://sky.pro/wiki/python/pochemu-python-populyaren-dlya-analiza-dannyh/> (дата обращения: 17.05.2026).
5. Scikit-learn для Python: что это за библиотека и как с ней работать [Электронный ресурс]. URL: <https://skillbox.ru/media/code/biblioteka-scikitlearn-kak-sozdat-svoy-pervyyu-mlproekt/> (дата обращения: 17.05.2026).
6. Основы Spring Boot: создаём веб-приложение на Java / Skillbox Media [Электронный ресурс]. URL: <https://skillbox.ru/media/code/zapuskajem-pervoe-vebprilozhenie-na-spring-boot/> (дата обращения: 17.05.2026).
7. Использование диаграммы вариантов использования UML при проектировании [Электронный ресурс]. URL: <https://habr.com/ru/articles/566218/> (дата обращения: 20.05.2026).
8. Функциональные и нефункциональные требования: основа IT-проектов [Электронный ресурс]. URL: <https://sky.pro/wiki/sql/funkcionalnye-i-nefunkcionalnye-trebovaniya/> (дата обращения: 20.05.2026).
9. Разница между Agile, Waterfall, Scrum: что выбрать для проектов [Электронный ресурс]. URL: <https://sky.pro/wiki/management/raznitsa-mezhdu-agile-waterfall-i-scrum-chto-vybrat-dlya-proekta/> (дата обращения: 21.05.2026).

10. Методология функционального моделирования процесса [Электронный ресурс]. URL: <https://www.ade-solutions.com/forensic/articles/metodologiya-funktsionalnogo-modelirovaniya-protssessa-notatsiya-protssessa-yazykom-standartov-i-luchsh/> (дата обращения: 21.05.2026).

11. Логическое моделирование баз данных: принципы и лучшие практики [Электронный ресурс]. URL: <https://sky.pro/wiki/sql/logicheskaya-model-bazy-dannyh-chto-eto-i-kak-eyo-sozdat/> (дата обращения: 21.05.2026).

12. Диаграмма Деятельности и Диаграмма Состояния [Электронный ресурс]. URL: <https://habr.com/ru/articles/907808/> (дата обращения: 28.05.2026).

13. Диаграмма последовательности – что это и как построить [Электронный ресурс]. URL: <https://practicum.yandex.ru/blog/sequence-diagram/> (дата обращения: 28.05.2026).

14. Диаграмма развертывания UML. Актуальная инструкция [Электронный ресурс]. URL: https://itonboard.ru/analysis/849-diagramma_razvertyvaniia_deployment_diagram_uml_full_tutorial/ (дата обращения: 29.05.2026).

15. Системы рекомендации: как работают умные алгоритмы персонализации [Электронный ресурс]. URL: <https://sky.pro/wiki/profession/rekomendatelnye-sistemy-v-data-science/> (дата обращения: 02.06.2026).

16. Коллаборативная фильтрация: что это такое, как работает [Электронный ресурс]. URL: <https://www.mango-office.ru/journal/for-marketing/osnovy/kollaborativnaya-filtraciya/> (дата обращения: 05.06.2026).

17. Item-based коллаборативная фильтрация своими руками [Электронный ресурс]. URL: <https://www.pvsm.ru/python/67159> (дата обращения: 05.06.2026).

18. Косинусное сходство в Python: методы расчета и парктическое применение [Электронный ресурс]. URL:

<https://sky.pro/wiki/analytics/kosinusnoe-shodstvo-v-python-metody-rascheta-i-prakticheskoe-primenenie/> (дата обращения: 10.06.2026).

19. Разреженные матрицы (sparse matrix) [Электронный ресурс]. URL: <https://python-school.ru/blog/python/sparse-matrix/> (дата обращения: 06.06.2026).

20. Репозиторий кода программной реализации системы рекомендаций товаров // GITHUB. [Электронный ресурс]. URL: <https://github.com/Diana1495/product-recommender-system/> (дата обращения: 06.06.2026).

21. Как использовать библиотеку Pandas Python: руководство для новичков [Электронный ресурс]. URL: <https://sky.pro/wiki/analytics/kak-ispolzovat-biblioteku-pandas-v-python-rukovodstvo-dlya-novichkov/> (дата обращения: 06.06.2026).

22. SciPy: что это за библиотека Python [Электронный ресурс]. URL: <https://blog.skillfactory.ru/glossary/scipy/> (дата обращения: 06.05.2026).

23. Функциональное тестирование: что это, виды, этапы [Электронный ресурс]. URL: <https://practicum.yandex.ru/blog/funkcionalnoe-testirovanie-po/> (дата обращения: 08.06.2026).

24. Нагрузочное тестирование для начинающих : JMeter [Электронный ресурс]. URL: <https://hirehi.ru/blog/nagruzochnoe-testirovanie-dlia-nachinaishchikh-jmeter-i-gatling-cto-vybrat-i-kak-nachat> (дата обращения: 08.06.2026).

Приложение А

(справочное)

Глоссарий

1С:Предприятие – платформа для автоматизации учёта на предприятиях, позволяющая вести справочники товаров, документы продаж и настраивать обмен данными с внешними системами через HTTP-сервисы.

CSV (Comma-Separated Values) – текстовый формат хранения табличных данных, где поля разделяются запятыми. Используется для обмена историей продаж между 1С и аналитическим модулем.

JSON (JavaScript Object Notation) – текстовый формат обмена данными, основанный на структуре «ключ – значение». Применяется для передачи информации о товарах и рекомендациях от 1С к веб-сайту.

Коллаборативная фильтрация – метод построения рекомендаций, основанный на анализе совместных покупок или схожих предпочтений пользователей. В работе используется объектно-ориентированный подход (item-based).

Косинусное сходство – метрика, измеряющая близость двух векторов через косинус угла между ними. В рекомендательных системах служит для количественной оценки схожести товаров по истории заказов.

Разреженная матрица – матрица, почти все элементы которой равны нулю. В проекте используется формат CSR для экономии памяти при хранении матрицы покупок «заказы – товары».

HTTP-сервис – программный интерфейс, созданный в среде 1С:Предприятие, который обрабатывает входящие HTTP-запросы и возвращает данные в формате JSON.

Spring Boot – фреймворк для разработки веб-приложений на Java, включающий встроенный сервер Tomcat и автоматическую конфигурацию.

Apache JMeter – инструмент для нагрузочного тестирования, позволяющий имитировать большое количество одновременных запросов к веб-серверу.

SLOC (Source Lines of Code) – метрика размера программы, определяемая количеством строк в исходном коде.

Метрика Холстеда – метод оценки сложности программы через подсчёт уникальных и общих операторов и операндов.

Метрика Чепина – метод оценки сложности, основанный на анализе ролей переменных (вводимые, модифицируемые, управляющие, паразитные).

Регламентное задание – автоматически выполняемая процедура в 1С, запускающая выгрузку данных и расчёт рекомендаций по расписанию (например, раз в неделю).

Диаграмма Ганта – вид диаграммы, используемый для визуализации расписания задач проекта: отображает длительность, последовательность и параллельность работ.

Приложение Б

(обязательное)

Листинг подсистемы «1С: Предприятие»

Листинг Б.1 – Программная реализация цикла обработки данных для машинного обучения в общем модуле «МЛ_Интеграция»

Процедура ВыгрузитьДляОбученияНаСервере() Экспорт

```

    ДатаНачала = ДобавитьМесяц(ТекущаяДата(), -6);

    Запрос = Новый Запрос;
    Запрос.Текст =
        "ВЫБРАТЬ
          |      РеализацияТоваровТовары.Ссылка.Номер КАК
НомерЧека,
          |      РеализацияТоваровТовары.Товар.Наименование КАК
ТоварНаименование
          | ИЗ
          |      Документ.РеализацияТоваров.Товары КАК
РеализацияТоваровТовары
          | ГДЕ
          |      РеализацияТоваровТовары.Ссылка.Проведен = ИСТИНА
          |      И РеализацияТоваровТовары.Ссылка.Дата >=
&ДатаНачала
          | УПОРЯДОЧИТЬ ПО
          |      НомерЧека";

    Запрос.УстановитьПараметр("ДатаНачала", ДатаНачала);

    РезультатЗапроса = Запрос.Выполнить();
    Выборка = РезультатЗапроса.Выбрать();

    ТекстФайла = Новый ТекстовыйДокумент;
    ТекстФайла.ДобавитьСтроку("order_id,item_name");

    Пока Выборка.Следующий() Цикл
        ЧистоеИмя = СтрЗаменить(Выборка.ТоварНаименование, ",",
        "");
        СтрокаТаблицы = Строка(Выборка.НомерЧека) + "," +
        ЧистоеИмя;
        ТекстФайла.ДобавитьСтроку(СтрокаТаблицы);
    КонецЦикла;

    ПутьКФайлуCSV = "D:\temp\sales_data.csv";
    ТекстФайла.Записать(ПутьКФайлуCSV, КодировкаТекста.UTF8);

    ПутьПитон = "D:\Programms\Python314\python.exe";
    ПутьСкрипт = "D:\лабы диана\4 курс\1с\диплом\ml_model.py";

    Команда = """" + ПутьПитон + """" """" + ПутьСкрипт + """";

    Попытка

        WshShell = Новый СОМОбъект("WScript.Shell");

```

```

WshShell.Run(Команда, 0, Истина);

Сообщить("Выгрузка и расчет выполнены успешно (в
фоновом режиме)");
Исключение

Сообщить("Ошибка фонового запуска, пробуем
обычный...");
ЗапуститьПриложение(Команда, , Истина);
КонецПопытки;

ФайлРекомендаций = Новый ТекстовыйДокумент;
ПутьРекомендаций = "D:\temp\recommendations.csv";

Попытка
    ФайлРекомендаций.Прочитать(ПутьРекомендаций,
КодировкаТекста.UTF8);
Исключение
    Сообщить("Файл с результатами не найден по пути: " +
ПутьРекомендаций);
    Возврат;
КонецПопытки;
НаборЗаписей =
РегистрыСведений.РекомендацииТоваров.СоздатьНаборЗаписей();
НаборЗаписей.Записать();

КолСтрок = ФайлРекомендаций.КоличествоСтрок();

Для НомерСтроки = 2 По КолСтрок Цикл
    ТекстСтроки =
ФайлРекомендаций.ПолучитьСтроку(НомерСтроки);
    Части = СтрРазделить(ТекстСтроки, ",");

    Если Части.Количество() < 3 Тогда Продолжить;
КонецЕсли;

    ИмяТовара = СокрЛП(Части[0]);
    ИмяРекомендации = СокрЛП(Части[1]);
    ВесСвязи = Число(СтрЗаменить(Части[2], ".", ","));
    СсылкаТовар =
Справочники.Товары.НайтиПоНаименованию(ИмяТовара);
    СсылкаРекомендация =
Справочники.Товары.НайтиПоНаименованию(ИмяРекомендации);

    Если НЕ СсылкаТовар.Пустая() И НЕ
СсылкаРекомендация.Пустая() Тогда
        МенеджерЗаписи =
РегистрыСведений.РекомендацииТоваров.СоздатьМенеджерЗаписи();
        МенеджерЗаписи.Товар = СсылкаТовар;
        МенеджерЗаписи.РекомендованныйТовар =
СсылкаРекомендация;
        МенеджерЗаписи.КоэффициентСвязи = ВесСвязи;
        МенеджерЗаписи.Записать();

```

```

        КонецЕсли;
    КонецЦикла;
    Сообщить ("Расчет окончен. Данные загружены в систему!");
КонецПроцедуры

```

Листинг Б.2 – Программная реализация метода GET HTTP-сервиса для формирования каталога товаров и списка рекомендаций в формате JSON

```

Функция get_dataGetProduct (Запрос)
    ЗапросТекст =
        "ВЫБРАТЬ
        |     Товары.Ссылка КАК Ссылка,
        |     Товары.Наименование КАК Наименование,
        |     ЕСТЬNULL (ЦеныТоваровСрезПоследних.Цена, 0) КАК
Цена
        | ИЗ
        |     Справочник.Товары КАК Товары
        |     ЛЕВОЕ СОЕДИНЕНИЕ
РегистрСведений.ЦеныТоваров.СрезПоследних (&Период, ) КАК
ЦеныТоваровСрезПоследних
        |     ПО Товары.Ссылка =
ЦеныТоваровСрезПоследних.Товар
        | ГДЕ
        |     Товары.Ссылка В
        |         (ВЫБРАТЬ
        |             Остатки.Товар
        |             ИЗ
        |                 РегистрНакопления.ОстаткиТоваров.Остатки
КАК Остатки
        |             ГДЕ
        |                 Остатки.КоличествоОстаток > 0) ";

    ЗапросКБазе = Новый Запрос (ЗапросТекст);
    ЗапросКБазе.УстановитьПараметр ("Период", ТекущаяДата());

    Попытка
        ВыборкаТоваров = ЗапросКБазе.Выполнить().Выбрать();
    Искключение
        Ответ = Новый HTTPСервисОтвет(200);
        Ответ.УстановитьТелоИзСтроки("");
        Возврат Ответ;
    КонецПопытки;

    СписокТоваров = Новый Массив;

    Пока ВыборкаТоваров.Следующий() Цикл

        ЗапросРек = Новый Запрос;
        ЗапросРек.Текст =
            "ВЫБРАТЬ

```

```

|
РекомендацииТоваров.РекомендованныйТовар.Наименование КАК
Название
| ИЗ
| РегистрСведений.РекомендацииТоваров КАК
РекомендацииТоваров
| ГДЕ
| РекомендацииТоваров.Товар = &ТекущийТовар
| УПОРЯДОЧИТЬ ПО
| РекомендацииТоваров.КоэффициентСвязи УБЫВ";

ЗапросРек.УстановитьПараметр ("ТекущийТовар",
ВыборкаТоваров.Ссылка);
ВыборкаРек = ЗапросРек.Выполнить().Выбрать();

МассивРек = Новый Массив;
Пока ВыборкаРек.Следующий() Цикл
    МассивРек.Добавить (ВыборкаРек.Название);
КонецЦикла;

СтруктураТовара = Новый Структура;
СтруктураТовара.Вставить ("id",
Строка (ВыборкаТоваров.Ссылка.УникальныйИдентификатор()));
СтруктураТовара.Вставить ("name",
ВыборкаТоваров.Наименование);
СтруктураТовара.Вставить ("price", ВыборкаТоваров.Цена);
СтруктураТовара.Вставить ("recommendations", МассивРек);

СписокТоваров.Добавить (СтруктураТовара);

КонецЦикла;

ЗаписьJSON = Новый ЗаписьJSON;
ЗаписьJSON.УстановитьСтроку();
ЗаписатьJSON (ЗаписьJSON, СписокТоваров);
СтрокаJSON = ЗаписьJSON.Закрыть();

Ответ = Новый HTTPСервисОтвет (200);
Ответ.УстановитьТелоИзСтроки (СтрокаJSON,
КодировкаТекста.UTF8,
ИспользованиеByteOrderMark.НеИспользовать);
Ответ.Заголовки.Вставить ("Content-Type", "application/json;
charset=utf-8");
Возврат Ответ;
КонецФункции

```

Листинг Б.3 – Программная реализация метода POST HTTP-сервиса для автоматической обработки заказов с сайта

Функция orderCreateOrder (Запрос)

```

    Ответ = Новый HTTPСервисОтвет(200);
    Ответ.Заголовки.Вставить("Content-Type", "application/json;
charset=utf-8");

    ТелоЗапроса = Запрос.ПолучитьТелоКакСтроку();

    Попытка

        Чтение = Новый ЧтениеJSON;
        Чтение.УстановитьСтроку(ТелоЗапроса);
        ДанныеЗаказа = ПрочитатьJSON(Чтение, Истина);
        Чтение.Закрыть();

        ИмяИзЗапроса = ДанныеЗаказа["customerName"];
        ТелефонИзЗапроса = ДанныеЗаказа["customerPhone"];

        КлиентСсылка =
        Справочники.Контрагенты.НайтиПоРеквизиту("Телефон",
        ТелефонИзЗапроса);

        Если КлиентСсылка.Пустая() Тогда
            Попытка
                НовыйКонтрагент =
                Справочники.Контрагенты.СоздатьЭлемент();
                НовыйКонтрагент.Наименование = ИмяИзЗапроса;
                НовыйКонтрагент.Телефон =
                ТелефонИзЗапроса;
                НовыйКонтрагент.ТипКонтрагента =
                Перечисления.ТипыКонтрагентов.ФизическоеЛицо;
                НовыйКонтрагент.Записать();
                КлиентСсылка = НовыйКонтрагент.Ссылка;
            Исключение
                КлиентСсылка =
                Справочники.Контрагенты.НайтиПоНаименованию("Иванов Иван");
            КонецПопытки;
        КонецЕсли;

        НовыйДокумент =
        Документы.РеализацияТоваров.СоздатьДокумент();
        НовыйДокумент.Дата = ТекущаяДата();
        НовыйДокумент.Клиент = КлиентСсылка;
        НовыйДокумент.Менеджер =
        Справочники.Сотрудники.НайтиПоНаименованию("Алексеева О.В.");

        МассивТоваров = ДанныеЗаказа["items"];
        Для Каждого СтрокаТЧ Из МассивТоваров Цикл
            НоваяСтрока = НовыйДокумент.Товары.Добавить();

            Идентификатор = Новый
            УникальныйИдентификатор(СтрокаТЧ["id"]);
            ТоварСсылка =
            Справочники.Товары.ПолучитьСсылку(Идентификатор);

```

```

        Если ТоварСсылка.ПолучитьОбъект() = Неопределено
Тогда
        ТоварСсылка =
Справочники.Товары.НайтиПоНаименованию(СтрокаТЧ["name"]);
        КонечЕсли;

        НоваяСтрока.Товар          = ТоварСсылка;
        НоваяСтрока.Количество    = СтрокаТЧ["qty"];
        НоваяСтрока.Цена          = СтрокаТЧ["price"];
        НоваяСтрока.Сумма         = НоваяСтрока.Количество *
НоваяСтрока.Цена;
        КонечЦикла;

        НовыйДокумент.Записать(РежимЗаписиДокумента.Запись);

        Ответ.УстановитьТелоИзСтроки("{\"status\":\"success\",
        \"number\":\"\" + НовыйДокумент.Номер + \"\"}\",
КодировкаТекста.UTF8);
        Лог = Новый ТекстовыйДокумент;
        Лог.ДобавитьСтроку(ОписаниеОшибки());
        Лог.Записать("D:\temp\lc_error.txt",
КодировкаТекста.UTF8);
        Ответ.КодСостояния = 500;
        Ответ.УстановитьТелоИзСтроки("{\"status\":\"error\"}\",
КодировкаТекста.UTF8);
        КонечПопытки;
        Возврат Ответ;
КонечФункции

```

Приложение В
(обязательное)
Листинг ML-модуля

```

import sys
import io
import pandas as pd
from sklearn.metrics.pairwise import cosine_similarity
from scipy import sparse

sys.stdout = io.TextIOWrapper(sys.stdout.buffer, encoding='utf-8')

try:
    df = pd.read_csv('D:/temp/sales_data.csv')
    print("Файл sales_data.csv успешно прочитан.")

    df['order_id'] = df['order_id'].astype('category')
    df['item_name'] = df['item_name'].astype('category')
    contact_matrix = sparse.csr_matrix((
        [1] * len(df),
        (df['order_id'].cat.codes, df['item_name'].cat.codes)
    ))

    print("Разреженная матрица построена.")
    item_similarity = cosine_similarity(contact_matrix.T)
    item_names = df['item_name'].cat.categories
    item_similarity_df = pd.DataFrame(item_similarity,
    index=item_names, columns=item_names)

    recommendations = []
    for item in item_similarity_df.columns:
        similar_items =
item_similarity_df[item].sort_values(ascending=False).drop(item)

        for rec_item, score in similar_items.head(3).items():
            if score > 0:
                recommendations.append([item, rec_item,
round(score, 2)])

    result_df = pd.DataFrame(recommendations,
columns=['antecedents', 'consequents', 'confidence'])
    result_df.to_csv('D:/temp/recommendations.csv', index=False,
encoding='utf-8-sig')

    print("\n--- Успех! ---")
    print(f"Обработано уникальных товаров: {len(item_names)}")
    print(f"Создано правил рекомендаций: {len(result_df)}")
    print("Результат сохранен в: D:/temp/recommendations.csv")

except Exception as e:
    print(f"Критическая ошибка в работе ML-модуля: {e}")

```

Приложение Г

(обязательное)

Листинг серверной части веб-приложения

Листинг Г.1 – Метод получения каталога товаров и рекомендаций из 1С

```

public List<Product> getAllProducts() {
    RestTemplate restTemplate = new RestTemplate();
    try {
        Product[] response =
restTemplate.getForObject(API_URL, Product[].class);
        return Arrays.asList(response);
    } catch (Exception e) {
        System.err.println("ОШИБКА СВЯЗИ С 1С: " +
e.getMessage());
        return List.of();
    }
}

```

Листинг Г.2 – Метод получения товара и его рекомендаций из 1С

```

public Product getProductById(String id) {
    return getAllProducts().stream()
        .filter(p -> p.getId().equals(id))
        .findFirst()
        .orElse(null);
}

```

Листинг Г.3 – Метод отправки данных о заказе в учетную систему

```

public String sendOrderTo1C(String jsonOrder) {
    RestTemplate restTemplate = new RestTemplate();
    String url1C = "http://localhost:8080/shop/hs/v1/order";
    try {
        return restTemplate.postForObject(url1C, jsonOrder,
String.class);
    } catch (Exception e) {
        System.err.println("Ошибка при отправке заказа в 1С: " +
e.getMessage());
        return "{\"status\":\"error\", \"message\":\"" +
e.getMessage() + "\"}";
    }
}

```