**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«Сибирский государственный университет науки и технологий**

**имени академика М.Ф. Решетнева»**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Институт (факультет) | | | Институт инженерной экономики | |
| Направление | | 09.03.03 «Прикладная информатика» | | |
| Направленность (профиль) | | | | «Цифровые инновации в управлении |
| предприятиями» | | | | |
| Кафедра | Информационных экономических систем | | | |

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА**

Вид ВКР: бакалаврская работа

|  |
| --- |
| **РАЗРАБОТКА РЕКОМЕНДАТЕЛЬНОЙ СИСТЕМЫ** |
| **ПОСТАВОК ПРОДОВОЛЬСТВЕННОЙ ПРОДУКЦИИ** |
| **ДЛЯ ЛОГИСТИЧЕСКОЙ КОМПАНИИ** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Обучающийся |  | / | Л. С. Цветков | / |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Руководитель |  | / | М. А. Масюк | / |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Ответственный за нормоконтроль |  | / | Н. Ю. Юферова | / |

Допускается к защите

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Заведующий кафедрой |  | / | М. А. Масюк | / |

«20» июня 2025 г.

Красноярск 2025

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«Сибирский государственный университет науки и технологий**

**имени академика М.Ф. Решетнева»**

|  |
| --- |
| Институт инженерной экономики |
| институт |
| Кафедра информационных экономических систем |
| кафедра |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| УТВЕРЖДАЮ | | | | |
| Заведующий кафедрой | | | | |
|  |  | | М. А. Масюк |
|  | | «24» декабря 2024 г. | | |

**ЗАДАНИЕ**

**НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ**

**в форме бакалаврской работы**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Обучающийся | | Цветков Лев Сергеевич | | | | | | | | | | | | | | | |
| Группа | БПЦ 21-01 | | | Направление (специальность) | | | | 09.03.03 | | | | | | | | | |
| «Прикладная информатика» | | | | | | | | | | | | | | | | | |
| Тема выпускной квалификационной работы | | | | | | | Разработка рекомендательной | | | | | | | | | | |
| системы поставок продовольственной продукции для логистической | | | | | | | | | | | | | | | | | |
| компании | | | | | | | | | | | | | | | | | |
| Утверждена приказом по университету от | | | | | | 17.03 | | 2025 г. № | | | | | | | 744д | | |
| Руководитель ВКР | | | М. А. Масюк заведующий кафедрой ИЭС, к. т. н., доцент | | | | | | | | | | | | | | |
| Исходные данные для ВКР | | | | | ООО «Сибирская логистическая компания» | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | |
| Перечень разделов ВКР | | | | | | | | | | | | | | | | | |
| 1. Анализ предметной области. | | | | | | | | | | | | | | | | | |
| 1. Проектирование рекомендательной системы. | | | | | | | | | | | | | | | | | |
| 1. Разработка рекомендательной системы. | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | |
| Перечень графического материала (с указанием обязательных чертежей | | | | | | | | | | | | | | | | | |
| при необходимости): | | | | презентация, раздаточный материал | | | | | | | | | | | | | |
| 1. Срок сдачи студентом первого варианта ВКР – | | | | | | | | | | | | «06» июня 2025 г | | | | | |
| 1. Срок сдачи студентом окончательного варианта ВКР – | | | | | | | | | | | | «20» июня 2025 г | | | | | |
| Руководитель ВКР | | | | | | | | | | / | | | М. А. Масюк | | | / | |
| Задание принял к исполнению | | | | | | | | | | | / | | | Л. С. Цветков | | | / |
|  | | | | | | | | | «24 декабря» 2024 г. | | | | | | | | |

СОДЕРЖАНИЕ

[Введение 4](#_Toc200358079)

[1 Анализ предметной области 6](#_Toc200358080)

[1.1 Описание рассматриваемой компании 6](#_Toc200358081)

[1.2 Организационная структура компании 6](#_Toc200358082)

[1.3 IT-инфраструктура компании 8](#_Toc200358083)

[1.4 Анализ бизнес-процессов компании 10](#_Toc200358084)

[1.5 Недостатки автоматизации бизнес-процессов компании 15](#_Toc200358085)

[1.6 Обзор существующих программных продуктов 16](#_Toc200358086)

[1.7 Виды рекомендательных систем и принцип их работы 18](#_Toc200358087)

[1.8 Выводы по главе 20](#_Toc200358088)

[2 Проектирование рекомендательной системы 21](#_Toc200358089)

[2.1 Нефункциональные требования к разработке рекомендательной системы 21](#_Toc200358090)

[2.2 Функциональные требования к разработке рекомендательной системы 23](#_Toc200358091)

[2.3 Диаграмма вариантов использования 24](#_Toc200358092)

[2.4 Аналитическая модель 27](#_Toc200358093)

[2.5 Выбор технологий для проектирования рекомендательной системы 37](#_Toc200358094)

[2.6 Обоснование выбора проектных решений 41](#_Toc200358095)

[2.7 Выводы по главе 42](#_Toc200358096)

[3 Разработка рекомендательной системы 43](#_Toc200358097)

[3.1 Анализ данных для обучения модели 43](#_Toc200358098)

[3.2 Архитектура модели 54](#_Toc200358099)

[3.3 Обучение модели и оценка качества 55](#_Toc200358100)

[3.4 Руководство программиста 57](#_Toc200358101)

[3.5 Руководство пользователя 58](#_Toc200358102)

[3.6 Тестирование системы 63](#_Toc200358103)

[3.7 Выводы по главе 64](#_Toc200358104)

[Заключение 65](#_Toc200358105)

[Список использованных источников 66](#_Toc200358106)

[Приложение 68](#_Toc200358107)

Введение

Актуальность.В условиях высокой конкуренции в сфере поставок продовольственных продукций важно точно прогнозировать, какие товары торговые точки будут закупать с наибольшей вероятностью. Это позволит поставщикам минимизировать финансовые риски и максимизировать свою прибыль. Традиционные методы прогнозирования спроса, основанные на субъективных оценках торговых представителей, зачастую оказываются недостаточно точными, занимают много времени и менее гибкие к изменениям потребительского поведения.

Разрабатываемая рекомендательная система решает эту проблему за счёт применения алгоритмов машинного обучения что позволит:

* точно прогнозировать наиболее вероятные закупки для каждой торговой точки на основе анализа их индивидуального покупательского поведения;
* выявлять скрытые паттерны спроса, которые неочевидны при традиционных методах анализа.

Для разработки рекомендательной системы для Сибирской логистической компании был выбран гибридный метод, который сочетает в себе коллаборативную и контентную фильтрации. Этот метод обеспечит точность и универсальность рекомендаций, т. к. он учитывает характеристики товаров (категория товаров, *ABC*-классификация (контентная фильтрация)) и историю закупок торговых точек (коллаборативная фильтрация). Система написана на языке программирования *Python* с использованием библиотек:

* *Pandas* – для работы с табличными данными;
* *LightFM* – для реализации рекомендательной системы с использованием гибридного метода;
* *Telebot* – для реализации *Telegram*-бота.

Цель и задачи.Целью бакалаврской работы является автоматизация формирования товарных предложений торговым точкам, путём разработки рекомендательной системы, основанной на гибридном методе (коллаборативная и контентная фильтрации).

Для достижения поставленной цели необходимо решить следующие задачи:

* провести обзор существующих программных продуктов;
* провести обзор видов рекомендательных систем и принцип их работы;
* сформулировать требования, необходимые для разработки;
* рассмотреть технологии и алгоритмы, которые будут использоваться для разработки;
* спроектировать разрабатываемую систему;
* определить набор данных для обучения модели машинного обучения;
* составить архитектуру модели машинного обучения;
* обучить и провести оценку качества модели;
* разработать для рекомендательной системы *Telegram*-бота;
* сформулировать руководство пользователя и программиста;
* провести тестирование разработанной системы.

Апробация и публикации. На основе результатов работы была написана и опубликована статья «Рекомендательные системы для логистических компаний» в журнале ВАК «*Journal of Monetary Economics and Management*» [1].

Структура работы. Пояснительная записка к бакалаврской работе состоит из введения, трёх глав, заключения и списка использованных источников из 20 наименований. Пояснительная записка изложена на 74 страницах, содержит 18 рисунков, 21 таблиц.

* в первой главе проведён обзор существующих программных продуктов их достоинства и недостатки. Также проведён обзор рекомендательных систем и принцип их работы;
* во второй главе сформулированы требования необходимые для разработки рекомендательной системы. Также рассмотрены технологии и алгоритмы, которые будут использоваться при разработке. Была спроектирована разрабатываемая система;
* в третьей главе определены данные для обучения модели машинного обучения, выполнена их обработка, составлена архитектура модели, проведено обучение и оценка качества модели, сформулировано руководство пользователя и программиста, а также проведено тестирование разработанной системы.
* в заключении подведены итоги проведённой работы и сформулированы выводы.

1 Анализ предметной области

Логистическая отрасль играет важную роль в современной экономике, обеспечивая перемещение товаров между производителями и конечными потребителями. Стремительное развитие технологий и динамические изменения на рынке создают трудности для логистических компаний. Одной из ключевых трудностей является необходимость оптимизации складских запасов, повышение точности прогнозирования спроса и снижение логистических издержек. На данный момент традиционные методы планирования, основанные на эмпирическом опыте, не позволяют полностью учитывать динамику рыночных изменений и индивидуальные особенности спроса в различных торговых точках.

1.1 Описание рассматриваемой компании

Рассмотрим, компанию ООО «Сибирская логистическая компания» (ООО «СЛК»), которое является крупнейшим оператором складских и транспортных услуг в Красноярском крае. Основными видами деятельности являются хранение и складирование прочих грузов (ОКВЭД 52.10.9). С момента основания в 2006 году компания активно развивает свою инфраструктуру и внедряет современные технологии, что позволяет ей занимать лидирующие позиции в отрасли.

Основным объектом компании является складской терминал класса «А» площадью 26000 м². Терминал включает 23 доковые зоны для приема и отгрузки товаров, более 18000 паллетомест для стеллажного хранения, 6300 ячеек в зоне штучного хранения и крытый железнодорожный тупик с вместимостью до 7 вагонов.

Компания также управляет мультитемпературным складом площадью 3500 м², который поддерживает температурные режимы от 0 до -18 °C, обеспечивая оптимальные условия хранения для продуктов питания и других товаров, требующих соблюдения температурного режима.

Транспортный парк компании насчитывает 62 единицы техники, что дает возможность осуществлять доставку грузов как по Красноярску, так и по всему региону.

Деятельность сибирской логистической компании сертифицирована по международному стандарту *ISO* 9001, а сохранность грузов дополнительно гарантируется страхованием в страховом доме «ВСК».

1.2 Организационная структура компании

Сибирская логистическая компания применяет линейно-штабную структуру, при которой функциональные службы помогают линейным руководителям в стратегическом планировании и анализе, но не обладают самостоятельным правом управленческих решений по нижестоящим звеньям.

1. Руководство:

* генеральный директор – Кожевников Сергей Валентинович;
* учредитель – Ярошенко Егор Сергеевич;

1. Департаменты и отделы:

* планово-экономический отдел. Формирование бюджетов, анализ эффективности маршрутов и проектов;
* отдел маркетинга и продаж. Поиск клиентов, подготовка коммерческих предложений, переговоры;
* отдел развития. Стратегическое планирование, внедрение новых сервисов;
* департамент логистики. Координация всех этапов движения груза, оптимизация цепочек поставок;
* транспортный отдел. Управление автопарком, планирование и мониторинг рейсов;
* складской отдел. Приём, хранение, комплектация и отгрузка грузов (*WMS*-процессы);
* отдел закупок и снабжения. Закупка ГСМ, запчастей, упаковочных и сопутствующих материалов;
* коммерческий отдел. Работа с контрактами, тендерами и документооборотом;
* финансовый отдел. Бюджетирование, учёт затрат, расчёты с контрагентами;
* отдел кадров (*HR*). Подбор персонала, обучение, кадровое администрирование;
* *IT*-служба. Поддержка внутренних информационных систем и внедрение автоматизации.

1. Взаимосвязь и подчинённость:

* все функциональные подразделения подчиняются непосредственно генеральному директору или его заместителям в рамках линейно-штабной модели;
* штабы, например, *IT*-служба, выполняют вспомогательные функции и не принимают самостоятельных решений, предоставляя аналитические и методические материалы линейным руководителям.

Организационная структура компании, построенная по линейно-штабному принципу, обеспечивает чёткое распределение полномочий: линейные руководители принимают управленческие решения, а функциональные подразделения оказывают им аналитическую и методическую поддержку. Все отделы и департаменты подчиняются генеральному директору или его заместителям, что способствует централизованному управлению и оперативному взаимодействию. Данная структура позволяет компании эффективно координировать логистические процессы и развивать стратегические направления.

1.3 IT-инфраструктура компании

Сибирская логистическая компания для автоматизации бизнес-процессов, использует отраслевые решения компании «1С» для управления складами и перевозками, *CRM*-систему для работы с клиентами, а также классические серверные и сетевые решения на базе платформ *Microsoft* и оборудования *Cisco* и *MikroTik*. Основные принципы построения инфраструктуры является отказоустойчивость, масштабируемость и интеграция в единое информационное пространство.

В таблице 1 описываются основные программные продукты, используемые компанией для автоматизации управления логистическими процессами, складскими операциями, перевозками и взаимодействием с клиентами. Применение этих систем позволяет оптимизировать бизнес-процессы, повысить точность планирования и улучшить контроль за выполнением операций.

Таблица 1 – Программное обеспечение компании

|  |  |  |
| --- | --- | --- |
| Категория | Программное обеспечение | Назначение и функции |
| *ERP* и *WMS* | 1С: *WMS* Логистика | Автоматизация складских операций в реальном времени, включая приемку, хранение, комплектацию и отгрузку. Интеграция с *ERP* и *TMS*. |
| 1С: *ERP* Управление предприятием | Централизованное управление финансами, закупками, запасами и логистикой. Обеспечивает единое информационное пространство. |
| *TMS* и учёт перевозок | 1С: *TMS* Логистика | Планирование и контроль перевозок, управление автопарком, мониторинг рейсов и расчет себестоимости перевозок. |
| *CRM* и документооборот | *Bitrix*24.*CRM* | Управление клиентской базой, обработка заявок, контроль продаж и интеграция с другими системами учета. |
| Инструменты аналитики и BI | Встроенные отчёты 1С, *BI*-модули *Bitrix*24 | Анализ ключевых показателей: эффективность маршрутов, загрузка складов, финансовые результаты. |

В таблице 2 представлено оборудование, обеспечивающее работу информационных систем компании. Данная аппаратная база гарантирует стабильность работы критически важных приложений, защиту данных и высокую производительность на всех участках работы.

Таблица 2 – Аппаратное обеспечение компании

|  |  |  |
| --- | --- | --- |
| Категория | Оборудование | Роль в инфраструктуре |
| Серверы и виртуальные платформы | Серверы *Dell* *PowerEdge* | Обеспечивают работу *ERP*, *WMS*, *TMS* и других корпоративных систем. |
| *Microsoft* *Hyper*-V | Виртуализация серверов для повышения отказоустойчивости и эффективности использования ресурсов. |
| Резервный сервер | Гарантирует непрерывность бизнес-процессов при авариях основного оборудования. |
| Рабочие станции и терминалы | ПК и ноутбуки *Dell* | Основные рабочие инструменты офисных сотрудников. |
| Терминалы сбора данных (*Honeywell*, *Zebra*) | Обеспечивают автоматизацию складских процессов через интеграцию с *WMS*. |
| Системы хранения и резервного копирования | *SAN*/*NAS* *QNAP* | Централизованное и безопасное хранение корпоративных данных. |
| *Windows* *Server* *Backup* | Защита данных от потерь за счет регулярного резервного копирования. |

В таблице 3 представлена сетевая инфраструктура компании, которая построена с учетом требований к безопасности, отказоустойчивости и высокой доступности сервисов. Данные решения обеспечивают стабильную связь между офисами, складами и транспортными узлами, а также защиту от внешних угроз.

Таблица 3 – Сетевое обеспечение компании

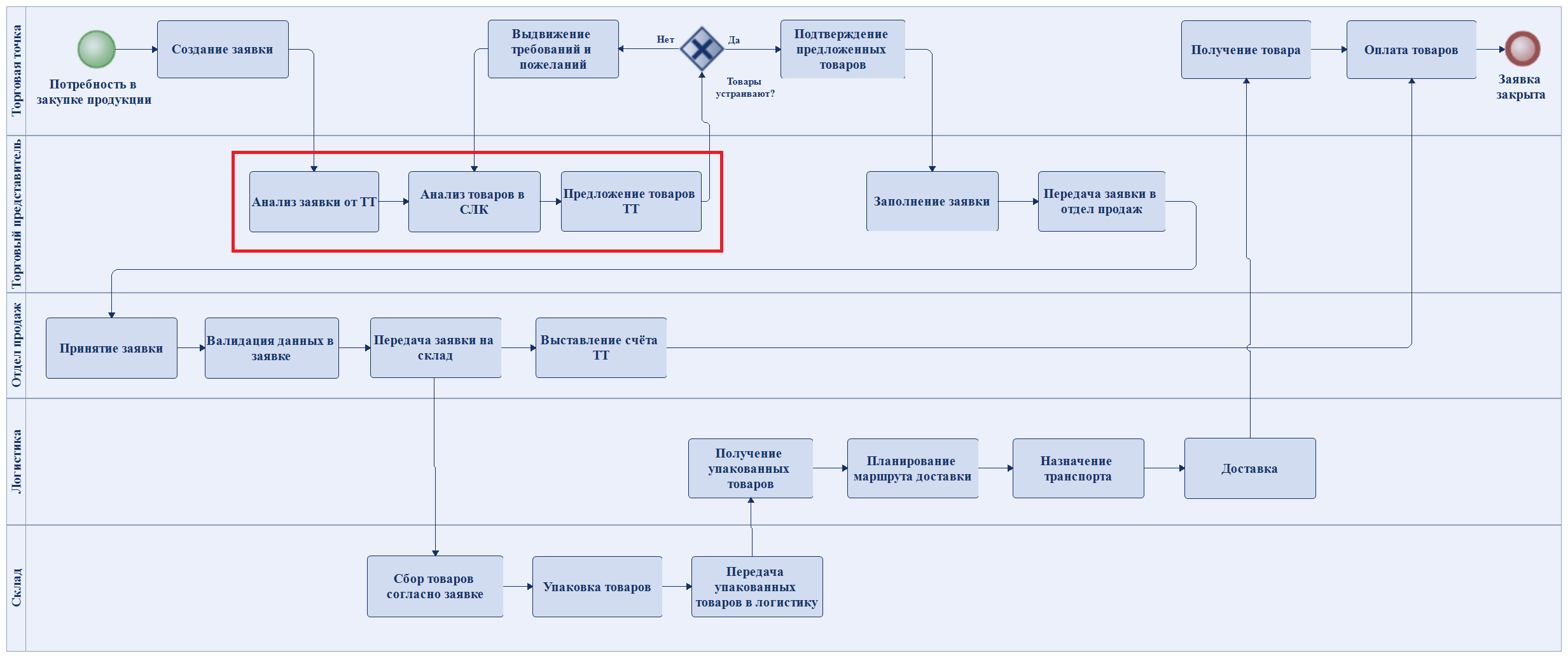
|  |  |  |
| --- | --- | --- |
| Категория | Оборудование/Технологии | Функции и преимущества |
| Маршрутизаторы и межсетевые экраны | *MikroTik* *CCR*1016-12*G* | Высоконадежный маршрутизатор для работы с большими объемами трафика. |
| *Фаерволы* *FortiGate* | Защита периметра сети, фильтрация трафика и организация *VPN*-туннелей. |
| Коммутаторы и *Wi*-*Fi* | *Cisco* *Catalyst* 9200/9500 | Обеспечивают высокоскоростное соединение между узлами сети с поддержкой *QoS*. |
| *MikroTik* | Беспроводное покрытие для сотрудников и *IoT*-устройств. |
| Сети доставки и *VPN* | *MPLS*-*VPN* | Надежные защищенные каналы связи с удаленными объектами. |
| *Active* *Directory* | Централизованное управление учетными записями и правами доступа. |
| Мониторинг и безопасность | *Zabbix*, *PRTG* | Контроль работоспособности сетевых устройств и серверов в реальном времени. |
| *Suricata* (*IDS*/*IPS*) | Обнаружение и предотвращение сетевых атак. |

*IT*-инфраструктура сибирской логистической компании представляет собой комплексное решение, объединяющее специализированное программное обеспечение, надежное аппаратное оснащение и отказоустойчивую сетевую архитектуру. Используемые системы (1С: *ERP*, *WMS*, *TMS*, *Bitrix*24) обеспечивают сквозную автоматизацию бизнес-процессов – от управления складом и перевозками до аналитики и работы с клиентами. Серверная и сетевая инфраструктура построена с учетом требований к производительности и безопасности, что гарантирует бесперебойную работу всех служб. Внедренные технологии позволяют компании эффективно масштабировать операции, минимизировать риски и поддерживать высокий уровень сервиса.

1.4 Анализ бизнес-процессов компании

В разделе представлен детальный анализ ключевых операционных процессов Сибирской логистической компании. В анализе основное внимание уделено сквозному процессу взаимодействия между торговыми точками, отделами продаж, складским комплексом и логистическими подразделениями компании. Для представления бизнес-процессов компании использована нотация *BPMN*, изображённая на рисунке 1, которая позволяет:

* отобразить последовательность операций;
* определить зоны ответственности каждого подразделения;
* обозначить точки принятия решений и условия перехода между этапами.



11

Рисунок 1 – Основной бизнес-процесс «Поставка продовольственной продукции»

Бизнес-процесс взаимодействия между торговой точкой, торговым представителем, отделом продаж, складом и логистическим подразделением реализуется в форме сквозной модели с разграничением зон ответственности и этапов.

* начальной точкой является потребность торговой точки в закупке определённой продукции, на основе которой формируется заявка и передаётся торговому представителю;
* торговый представитель осуществляет предварительный анализ содержимого заявки, проводит ручной подбор товаров на основе ассортимента и наличия, после чего направляет торговой точке предложение. В случае одобрения со стороны торговой точки представитель завершает оформление заявки, в случае отказа – возвращается к уточнению пожеланий торговой точки и повторному подбору ассортимента. Этот процесс позволяет учитывать индивидуальные предпочтения клиента и минимизировать риски возникновения недовольства полученным товаром;
* далее заявка направляется в отдел продаж, где происходит её регистрация, валидация данных, а именно проверка актуальности цен, артикулов, остатков, также формируется счёт на оплату товаров. Параллельно с этим процессом осуществляется складская обработка заявки: на основе утверждённой информации производится отбор продукции, упаковка в соответствии с требованиями безопасности и сохранности, после чего упакованные товары передаются логистическому подразделению;
* после получения упакованных заказов логистическое подразделение формирует оптимальные маршруты доставки, учитывая географическое расположение торговой точки, загруженность дорожной сети и доступные транспортные ресурсы. Для выполнения доставки назначается транспортное средство;
* в заключение, после прибытия груза торговая точка осуществляет приёмку продукции с проверкой количества, целостности упаковки и соответствия содержимого ранее утверждённому заказу. Осуществляется процедура оплаты товара, на основании ранее выставленного счёта. В заключении происходит закрытие заявки.

Таблица бизнес-процессов представлена в таблице 1

Таблица 4 – Описание бизнес-процессов компании

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Этап процесса | Участник | Описание действия | Предшествующий этап | Последующий этап |
| 1 | Потребность в закупке продукции | Торговая точка | Инициирование потребности в пополнении запасов | - | Создание заявки |
| 2 | Создание заявки | Торговая точка | Формирование начальной заявки на поставку | Потребность в закупке продукции | Анализ заявки от ТТ |
| 3 | Анализ заявки от ТТ | Торговый представитель | Оценка корректности и полноты заявки | Создание заявки | Подбор товаров в СЛК |
| 4 | Подбор товаров в СЛК | Торговый представитель | Подбор предложений по запрошенным позициям | Анализ заявки от ТТ | Предложение товаров ТТ |
| 5 | Предложение товаров ТТ | Торговый представитель | Предоставление клиенту предложенных товаров | Подбор товаров в СПК | Уточнение пожеланий или подтверждение |
| 6 | Товары устраивают? (условие) | Торговый представитель | Принятие решения торговой точкой | Предложение товаров ТТ | Подтверждение / Выдвижение требований |
| 7 | Выдвижение требований и пожеланий | Торговая точка | Уточнение характеристик и параметров заказа | Товары не устраивают | Возврат к подбору товаров |
| 8 | Подтверждение предложенных товаров | Торговая точка | Утверждение предложенного ассортимента | Товары устраивают | Заполнение заявки |
| 9 | Заполнение заявки | Торговый представитель | Оформление заявки по согласованным позициям | Подтверждение предложенных товаров | Передача заявки в отдел продаж |
| 10 | Передача заявки в отдел продаж | Торговый представитель | Перемещение оформленной заявки в обработку | Заполнение заявки | Принятие заявки |

Продолжение таблицы 4

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Этап процесса | Участник | Описание действия | Предшествующий этап | Последующий этап |
| 11 | Принятие заявки | Отдел продаж | Регистрация и принятие заявки к исполнению | Передача заявки | Валидация данных |
| 12 | Валидация данных в заявке | Отдел продаж | Проверка наличия, цен, артикулов и правильности | Принятие заявки | Передача заявки на склад |
| 13 | Передача заявки на склад | Отдел продаж | Отправка заявки в складское подразделение | Валидация данных | Сбор товаров согласно заявке |
| 14 | Выставление счёта ТТ | Отдел продаж | Формирование счёта за товары | Передача заявки | Оплата товаров |
| 15 | Сбор товаров согласно заявке | Склад | Отбор запрашиваемой продукции | Передача заявки | Упаковка товаров |
| 16 | Упаковка товаров | Склад | Упаковка отобранной продукции | Сбор товаров | Передача в логистику |
| 17 | Передача упакованных товаров в логистику | Склад | Передача логистическому подразделению | Упаковка товаров | Получение упакованных товаров |
| 18 | Получение упакованных товаров | Логистика | Подтверждение получения заказа | Передача упакованных товаров | Планирование маршрута |
| 19 | Планирование маршрута доставки | Логистика | Построение маршрута до торговой точки | Получение упакованных товаров | Назначение транспорта |
| 20 | Назначение транспорта | Логистика | Назначение транспорта | Планирование маршрута | Доставка |
| 21 | Доставка | Логистика | Фактическая доставка до торговой точки | Назначение транспорта | Получение товаров |
| 22 | Получение товара | Торговая точка | Подтверждение получения товаров | Доставка | Оплата товаров |

Окончание таблицы 4

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Этап процесса | Участник | Описание действия | Предшествующий этап | Последующий этап |
| 23 | Оплата товаров | Торговая точка | Перевод денежных средств на основе счёта | Получение товаров | Заявка закрыта |
| 24 | Заявка закрыта | Торговая точка | Завершение бизнес-процесса | Оплата товаров | - |

1.5 Недостатки автоматизации бизнес-процессов компании

В рамках анализа текущего состояния автоматизации бизнес-процессов компании, можно выявить ряд существенных недостатков, снижающих эффективность деятельности компании, а именно в контексте работы торговых представителей. Один из наиболее проблемных и трудоёмких процессов – это подбор ассортимента товаров для торговых точек, осуществляемая торговыми представителями вручную. Этот процесс представляет слабое звено общей системе управления поставками из-за объёма обрабатываемой информации и отсутствия эффективных автоматизированных инструментов поддержки принятия решений.

Торговый представитель при планировании товарных предложений для торговых точек вынужден самостоятельно анализировать массив данных, включающий более 20000 товарных наименований. Основная цель этого процесса – выявить товары, которые потенциально могут заинтересовать конкретную торговую точку, обеспечить высокую вероятность их последующей закупки, а также максимизировать выгоду для компании. Очевидно, что выполнение этой задачи в ручном режиме сопряжено с колоссальными временными затратами, высокими нагрузками на персонал и снижением точности и полноты анализа. Длительность обработки информации, необходимость поиска релевантных товаров с учётом предпочтений клиентов и логистических ограничений, а также ручной отбор каждого товарного предложения приводит к тому, что торговый представитель физически не в состоянии обработать и оценить весь имеющийся ассортимент качественно и своевременно.

Кроме того, отсутствие автоматизированных средств классификации и прогнозирования спроса ограничивает возможности для системного подхода к формированию товарного предложения. Это означает, что значительная часть решений базируется на интуиции, субъективном опыте сотрудника и часто неформализованных знаниях о предпочтениях конкретных торговых точек. В условиях высокой конкуренции и необходимости оперативного реагирования на изменяющиеся рыночные условия такой подход становится неэффективным и экономически нецелесообразным.

Резюмируя, текущая модель подбора товаров не только увеличивает трудозатраты и снижает продуктивность труда торгового представителя, но и создаёт риски упущенной выгоды как для самой логистической компании, так и для её партнёрских торговых точек. Автоматизация данного бизнес-процесса позволила бы не только сократить время на подготовку предложений, но и значительно повысить их релевантность и коммерческую результативность. Следовательно, необходимость внедрения инструментов персонализированных рекомендаций в данном бизнес-процессе является актуальной и критически важной для повышения эффективности деятельности предприятия.

1.6 Обзор существующих программных продуктов

В настоящее время представлено множество готовых решений для построения рекомендательных систем. Каждое из них обладает определённым набором функциональных возможностей и предназначено для решения типовых задач персонализации. Однако при выборе программного продукта важно учитывать как его технические характеристики, так и соответствие специфике деятельности компании. В данном разделе рассматриваются существующие программные продукты, а также их достоинства и недостатки.

* *Amazon Personalize* – сервис от *AWS*, предназначенный для построения персонализированных рекомендаций в реальном времени. *Amazon Personalize* автоматически обучает модели на пользовательских данных, предлагает готовые решения под различные бизнес-задачи [2];
* *Google Recommendations AI* – сервис входящий в состав *Vertex AI*, предназначенный для генерации персонализированных рекомендаций на основе взаимодействий пользователей и контекста. Сервис использует современные модели глубокого обучения разработанные *Google* [3];
* *Microsoft Azure Personalizer* – сервис реализует концепцию обучения с подкреплением, выбирая наиболее подходящее «действие» на основе накопленного «вознаграждения» за предыдущие действия пользователей [4].

Достоинства и недостатки каждого существующего программного продукта представлены в таблице 5.

Таблица 5 – Достоинства и недостатки каждого существующего программного продукта

|  |  |  |
| --- | --- | --- |
| Существующий программный продукт | Достоинства | Недостатки |
| *Amazon Personalize* | * автоматизация обучения моделей; * высокая масштабируемость; * поддержка нескольких сценариев рекомендаций. | * высокая стоимость при больших объёмах данных; * ограниченная кастомизация алгоритмов. |
| *Google Recommendations AI* | * высокое качество за счёт нейросетевых моделей; * интеграция с аналитическими и тесовыми инструментами *Google.* | * сложность интеграции с внешними системами; * зависимость от экосистем *Google*. |
| *Microsoft Azure Personalizer* | * адаптация к пользовательскому поведению в реальном времени; * простая *REST*-интеграция. | * сервис выведен из активной поддержки; * необходимость настройки *reward*-функций. |

Существующие программные решения для реализации рекомендательных систем обладают широким функционалом, однако их применение в условиях сибирской логистической компании сопряжено с рядом серьёзных ограничений:

* актуальная тема импортозамещения делает использование зарубежных программных продуктов рискованным из-за возможных проблем с доступностью, ростом стоимости и ограничениями в технической поддержке;
* информационная система управления, функционирующая в компании, может оказаться несовместимой с готовыми иностранными решениями, что приведёт к существенным затратам на интеграцию и настройку;  
  Универсальность таких продуктов зачастую сопровождается избыточностью функционала и недостаточной адаптацией под специфические задачи конкретного предприятия;
* создание собственной системы позволяет обеспечить полный контроль над процессами хранения и обработки данных, тем самым снижая риск утечек информации и повышая общую надёжность ИТ-инфраструктуры;
* индивидуальная разработка предоставляет возможность гибкого масштабирования, позволяя оперативно реагировать на внутренние потребности и изменения рыночной ситуации;
* создание собственной системы может быть изначально ориентировано на особенности бизнес-процессов компании, что повышает эффективность и точность функционирования системы.

Таким образом, выбор в пользу создания собственной рекомендательной системы представляется обоснованным и стратегически целесообразным.

1.7 Виды рекомендательных систем и принцип их работы

Рекомендательные системы – это программы, которые осуществляют подбор товаров, услуг или контента и предлагают их пользователям. Такие системы собирают информацию о пользователях, товарах или услугах, например, история покупок, оценки, которые пользователь ставит на товар и т. п. На основе этих данных с помощью алгоритмов машинного обучения система генерирует рекомендации. Для разработки рекомендательных систем используют 4 подхода:

* коллаборативная рекомендательная система. Коллаборативная фильтрация генерирует рекомендации, основанные на предыдущих действий пользователя. Эта модель строится только на основе действий пользователя или группы пользователей с похожими характеристиками. В случае, когда коллаборативная фильтрация использует действие других пользователей, она использует знания для генерации рекомендаций на основе сходства пользователей. Рекомендации основываются на автоматическом сотрудничестве множества пользователей и на выделении (фильтрации) тех пользователей, которые показывают схожие предпочтения или поведение [5];
* контентная рекомендательная система. Контентная фильтрация в отличии от коллаборативных рекомендательных систем, действия пользователя опускаются. Товары или услуги рекомендуются на основе их характеристик: категория, производитель, класс и т. д. Такие рекомендательные системы используют, например, интернет-магазины [5];
* гибридная рекомендательная система. Гибридная фильтрация сочетает в себе коллаборативную и контентную фильтрации. Этот подход значительно повышает эффективность и точность рекомендательной системы. Гибридная фильтрация предоставляет возможность сначала взвесить результаты контентной фильтрации, а затем смещать веса на основе коллаборативной фильтрации [6].

Достоинства и недостатки каждого вида рекомендательной системы представлены в таблице 6.

Таблица 6 – Достоинства и недостатки видов рекомендательных систем

|  |  |  |
| --- | --- | --- |
| Тип рекомендательной системы | Достоинства | Недостатки |
| Коллаборативная фильтрация | * простота реализации; * высокая точность при наличии достаточного объема данных | * проблема «холодного старта» для новых пользователей и объектов; * требует большого объема данных |
| Контентная фильтрация | * хорошо работает при небольшом объеме данных; * позволяет использовать специфические атрибуты объектов | * ограниченность в разнообразии рекомендаций; * рекомендует только похожие объекты |
| Гибридные системы | * совмещение преимуществ разных методов; * повышение точности рекомендаций | * сложность реализации; * высокие вычислительные затраты |

Учитывая поставленную задачу и особенности предметной области, в рамках выпускной квалификационной работы была выбрана гибридная рекомендательная система, включающая в себя коллаборативную и контентную фильтрации. Данный выбор обусловлен по следующим причинам:

* во-первых, в распоряжении имеется история закупок торговых точек, что позволяет использовать методы коллаборативной фильтрации для выявления сходства в покупательском поведении торговых точек. Однако использование только коллаборативного подхода вызывает проблему «холодного старта» для новых товаров и торговых точек, а также не учитывает характеристики товаров;
* во-вторых, контентная фильтрация даёт возможность опираться на характеристики товаров, такие как категория товара и ABC-классификация, что очень важно, т. к. ассортимент товаров может быть очень разнообразным. Однако, контентная фильтрация ограничена в разнообразии рекомендаций и не учитывает поведение торговых точек.

Гибридный подход, позволяет объединить преимущества обоих подходов: учитывать как закупки торговых точек, так и характеристики товаров, тем самым обеспечивая более точные и релевантные рекомендации. Также гибридный подход обеспечивает гибкость масштабирования, адаптивность к изменениям поведения торговых точек.

1.8 Выводы по главе

В первой главе были рассмотрены ключевые особенности логистической отрасли, а также определены актуальные задачи, стоящие перед логистическими компаниями в условиях высокой конкуренции и необходимости точного прогнозирования спроса. Особое внимание было уделено анализу деятельности сибирской логистической компании, для которой разрабатывается рекомендательная система.

Далее в главе были проанализированы существующие программные продукты, предназначенные для построения рекомендательных систем, такие как *Amazon Personalize*, *Google Recommendations* *AI* и *Microsoft Azure Personalizer*. Для каждого из решений были выявлены их достоинства и недостатки, а также обозначены причины, по которым их применение в рассматриваемой компании является ограниченным.

Также в данной главе было раскрыто понятие рекомендательной системы, проведён обзор основных типов фильтрации: коллаборативной, контентной и гибридной. Были рассмотрены их принципы работы, сильные и слабые стороны. В завершение главы обоснован выбор гибридного подхода как наиболее эффективного решения, обеспечивающего точность рекомендаций за счёт сочетания данных о покупательской активности и характеристиках товаров.

2 Проектирование рекомендательной системы

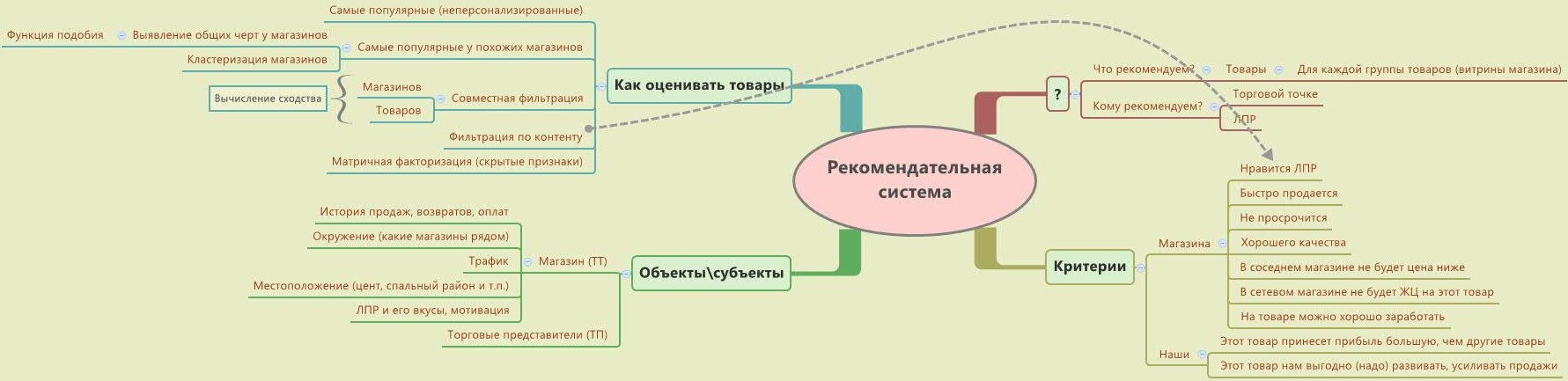
2.1 Нефункциональные требования к разработке рекомендательной системы

Нефункциональные требования – это характеристики системы, описывающие её качество и условия эксплуатации, но не затрагивающие непосредственно выполняемые ею функции. Они фокусируются на таких аспектах, как производительность, надежность, безопасность, масштабируемость, удобство использования и совместимость системы, обеспечивая её соответствие ожиданиям пользователей и бизнес-задачам [7].

Рассмотрим нефункциональные требования для разработки рекомендательной системы:

1. **Объекты и субъекты рекомендации:**
   1. **Объекты:**
      1. Товары, которые подлежат рекомендации.
   2. **Субъекты:**
      1. Торговые точки, для которых генерируются рекомендации.
      2. Лица, принимающие решения в торговых точках, с учётом их предпочтений и мотивов.
2. **Критерии рекомендации для товаров и магазинов:**
   1. **Для товаров:**
      1. Высокий спрос на товар.
   2. **Для магазинов:**
      1. Учитывать прибыльность товара для магазина.
      2. Прогнозируемая выгода от развития ассортимента данного товара.
3. **Методы оценки товаров:**
   1. Использование функции подобия для выявления общих характеристик у магазинов и товаров.
   2. Учет истории продаж.
4. **Технические аспекты:**
   1. Автоматизация сбора и обработки данных о продажах, и предпочтениях клиентов.
   2. Разработка гибкой архитектуры, позволяющей интегрировать различные источники данных.
5. **Цели системы:**
   1. Повышение точности рекомендаций по подбору ассортимента.
   2. Автоматизация процесса выбора товаров и анализа данных.
   3. Увеличение уровня удовлетворённости торговых точек и их конечных клиентов.

Нефункциональные требования к рекомендательной системе представлены на рисунке 2.



22

Рисунок 2 – Нефункциональные требования к рекомендательной системе

2.2 Функциональные требования к разработке рекомендательной системы

Функциональные требования – это спецификации, описывающие, какие функции и задачи должна выполнять система. Они определяют конкретные действия, операции и поведение системы, которые необходимы для выполнения её основной цели. Эти требования включают описание взаимодействий с пользователем, обработки данных и работы системы в заданных сценариях использования [7].

Рассмотрим функциональные требования для разработки рекомендательной системы:

1. Ввод данных пользователем:
   1. Пользователь должен ввести название торговой точки.
   2. Пользователь должен указать категорию товаров, для которой требуется рекомендация.
2. Обработка и анализ данных:
   1. Система должна анализировать данные, сопоставляя введённые параметры с историческими данными о продажах, спросе и прибыльности.
   2. Обработка данных должна учитывать индивидуальные особенности торговой точки, такие как её ассортимент и потребности.
3. Формирование рекомендаций:
   1. Система должна автоматически генерировать список из 3-12 товаров, наиболее подходящих для предложений данной торговой точке по заданной категории.
   2. Рекомендации должны учитывать оптимальный баланс между интересами логистической компании (максимизация прибыли) и торговой точки (высокая удовлетворённость клиентов, минимизация рисков нереализации товара).
4. Отображение результатов:
   1. Система должна выводить список рекомендуемых товаров в удобной для восприятия форме.
5. Учет многофакторной оценки:
   1. Алгоритм формирования рекомендаций должен учитывать:
      1. Прогнозируемую востребованность товара.
      2. Прибыльность для логистической компании.
      3. Снижение рисков избыточных или непроданных запасов для торговой точки.
6. Интерактивность и корректировки:
   1. Пользователь должен иметь возможность изменить параметры запроса без перезапуска системы.

2.3 Диаграмма вариантов использования

При использовании подхода на основе прецедентов функциональные требования задаются в терминах актеров, в роли которых выступают пользователи системы, и прецедентов. Актер участвует в прецеденте. Прецедент устанавливает последовательность взаимодействий между одним или несколькими актерами и системой. На этапе определения требований модель прецедентов описывает систему как черный ящик, а взаимодействие между актерами и системой, то есть действия пользователя и реакция на них системы, указываются в словесной форме. Прецеденты в данной модели выражают внешние требования к системе. Каждый прецедент описывает поведение некоторой части системы, не раскрывая ее внутренней структуры [8].

**Диаграмма прецедентов**

На рисунке 3 показано одно действующее лицо, которое будет работать с рекомендательной системой – торговый представитель. Показаны 5 прецедентов:

* выбор категории товара;
* выбор торговой точки;
* выбор количества рекомендаций;
* генерация рекомендаций;
* заполнение параметров заново.

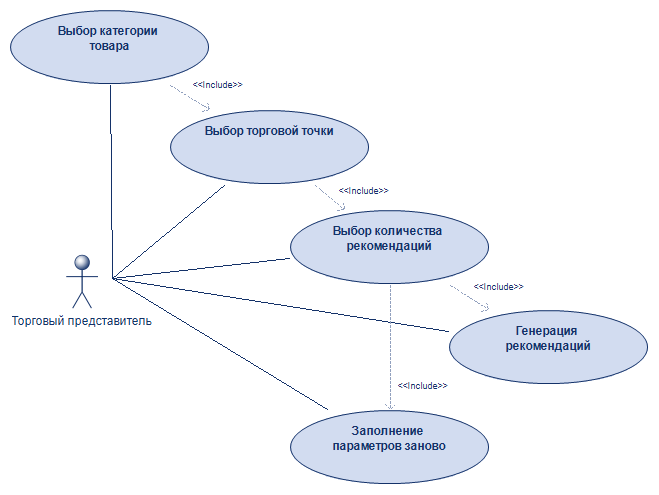


Рисунок 3 – Диаграмма прецедентов

**Спецификация прецедентов**

Спецификация прецедента – основной источник информации для выполнения анализа и проектирования системы [9]. Содержание спецификации представлено в таблицах 7-11 в полной и конструктивной форме.

Таблица 7 описывает процесс выбора категории товара торговым представителем. В ней указаны ключевые аспекты прецедента, такие как имя, сводка, зависимости, актёры, предусловия, описание последовательности действий, альтернативные сценарии и постусловия. Прецедент начинается с нажатия кнопки «Получить рекомендации» и завершается сохранением выбранной категории товара.

Таблица 7 – Прецедент «Выбор категории товара»

|  |  |  |
| --- | --- | --- |
| 1 | Имя прецедента | Выбор категории товара |
| 2 | Сводка | Пользователь из списка категорий товаров выбирает необходимую для получения рекомендаций |
| 3 | Зависимости | Зависимостей нет |
| 4 | Актёры | Торговый представитель |
| 5 | Предусловия | Пользователь должен иметь доступ к системе |
| 6 | Описание | → актёр нажимает на кнопку «Получить рекомендации»;  ← система открывает окно с выбором категорий товаров;  → актёр выбирает одну категорию товара;  ← система переходит к выбору торговой точки. |
| 7 | Альтернативы | Пользователь не имеет доступа к системе, система недоступна. |
| 8 | Постусловия | Категория товара успешно сохранена |

В таблице 8 представлен прецедент выбора торговой точки. Она содержит аналогичные разделы, включая зависимость от предыдущего прецедента «Выбор категории товара».

Таблица 8 – Прецедент «Выбор торговой точки»

|  |  |  |
| --- | --- | --- |
| 1 | Имя прецедента | Выбор торговой точки |
| 2 | Сводка | Пользователь из списка торговых точек выбирает необходимую для получения рекомендаций |
| 3 | Зависимости | Зависит от прецедента «Выбор категории товара» |
| 4 | Актёры | Торговый представитель |
| 5 | Предусловия | Пользователь должен иметь доступ к системе |
| 6 | Описание | ← система после выбора категории товара сразу открывает окно с выбором торговой точки;  → актёр выбирает одну торговую точку;  ← система переходит к выбору количества рекомендаций. |
| 7 | Альтернативы | Пользователь не имеет доступа к системе, система недоступна, пользователь сперва должен выбрать категорию товара. |
| 8 | Постусловия | Торговая точка успешно сохранена |

Таблица 9 описывает процесс выбора количества рекомендаций, которые система должна сгенерировать. Прецедент зависит от выбора торговой точки и включает этапы взаимодействия пользователя с системой, такие как выбор количества рекомендаций (3, 6, 9 или 12) и переход к следующим действиям.

Таблица 9 – Прецедент «Выбор количества рекомендаций»

|  |  |  |
| --- | --- | --- |
| 1 | Имя прецедента | Выбор количества рекомендаций |
| 2 | Сводка | Пользователь из списка количества рекомендаций выбирает необходимую для получения рекомендаций |
| 3 | Зависимости | Зависит от прецедента «Выбор торговой точки» |
| 4 | Актёры | Торговый представитель |
| 5 | Предусловия | Пользователь должен иметь доступ к системе |
| 6 | Описание | ← система после выбора торговой точки сразу открывает окно с выбором количества рекомендаций;  → актёр выбирает количество рекомендаций (3, 6, 9, 12 рекомендаций);  ← система выводит заполненные параметры и выбор дальнейших действий, а именно сгенерировать рекомендации или заполнить параметры заново. |
| 7 | Альтернативы | Пользователь не имеет доступа к системе, система недоступна, пользователь сперва должен выбрать категорию товара и торговую точку. |
| 8 | Постусловия | Количество рекомендаций успешно сохранены |

Таблица 10 детализирует процесс генерации рекомендаций на основе ранее выбранных параметров. В ней указаны действия пользователя – нажатие кнопки «Сгенерировать рекомендации» и реакции системы – передача параметров в модель и вывод результатов.

Таблица 10 – Прецедент «Генерация рекомендаций»

|  |  |  |
| --- | --- | --- |
| 1 | Имя прецедента | Генерация рекомендаций |
| 2 | Сводка | Пользователь после заполнения рекомендация нажимает кнопку сгенерировать рекомендации |
| 3 | Зависимости | Зависит от прецедента «Выбор количества рекомендаций» |
| 4 | Актёры | Торговый представитель |
| 5 | Предусловия | Пользователь должен иметь доступ к системе |
| 6 | Описание | → актёр после выбора количества рекомендаций нажимает кнопку сгенерировать рекомендации;  ← система передаёт ранее заполненные параметры в модель и выводит актёру рекомендации. |
| 7 | Альтернативы | Пользователь не имеет доступа к системе, система недоступна, пользователь сперва должен выбрать количество рекомендаций. |
| 8 | Постусловия | Генерация рекомендаций выполнена и выведена на экран |

В таблице 11 описан прецедент сброса ранее введённых параметров и возврата к начальному этапу выбора категории товара. Пользователь нажимает кнопку «Заполнить заново», после чего система обнуляет параметры и переходит к началу процесса.

Таблица 11 – Прецедент «Заполнение параметров заново»

|  |  |  |
| --- | --- | --- |
| 1 | Имя прецедента | Заполнение параметров заново |
| 2 | Сводка | Пользователь после заполнения рекомендация нажимает кнопку заполнить заново |
| 3 | Зависимости | Зависит от прецедента «Выбор количества рекомендаций» |
| 4 | Актёры | Торговый представитель |
| 5 | Предусловия | Пользователь должен иметь доступ к системе |
| 6 | Описание | → актёр после выбора количества рекомендаций нажимает кнопку заполнить заново;  ← система обнуляет ранее заполненные параметры и переходит к выбору категории товара. |
| 7 | Альтернативы | Пользователь не имеет доступа к системе, система недоступна, пользователь сперва должен выбрать количество рекомендаций. |
| 8 | Постусловия | Параметры обнулены система перешла к выбору категории товара |

2.4 Аналитическая модель

Аналитическая модель используется разработчиком для того, чтобы понять, какое оформление должно быть у системы. На данном этапе описывается, как функциональность системы будет реализована, моделируется структура внутреннего вида. Формируются статическая и динамическая модели [9].

**Статическое моделирование**

Концептуальная статическая модель строится на раннем этапе анализа и используется для лучшего освоения предметной области. Цель состоит в том, чтобы сосредоточить внимание на тех аспектах предметной области, которые наиболее заметны в статической модели, в частности на физических и информационно насыщенных классах, которые называются сущностными.

При построении статической модели предметной области сначала моделируются физические, сущностные и контекста классы. К физическим, классам относятся классы, обладающие физическими характеристиками, то есть описывающие предметы, которые можно увидеть или потрогать. Это физические устройства (нередко являющиеся частью предметной области во встраиваемых приложениях), пользователи, внешние системы и таймеры. Сущностными называются концептуальные информационно насыщенные классы, которые обычно являются устойчивыми, то есть долго живущими. Сущностные классы особенно часто встречаются в информационных системах [8].

**Диаграмма классов контекста**

Диаграмма классов контекста системы формируется посредством статического моделирования внешних классов, взаимодействующих с системой [9]. На рисунке 4 представлена диаграмма классов контекста для рекомендательной системы.

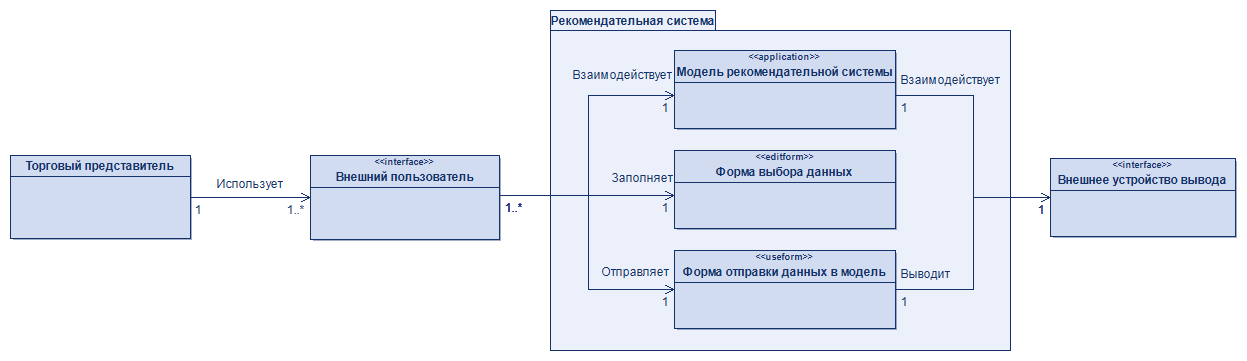


Рисунок 4 – Диаграмма классов контекста

На диаграмме классов контекста представлена архитектура проектируемой рекомендательной системы, включая её ключевые компоненты и взаимодействия с внешними сущностями. Основным пользователем системы является торговый представитель, который взаимодействует с системой через два основных интерфейса:

1. Форма взаимодействия с данными. Эта форма предназначена для выбора параметров, необходимых для генерации рекомендаций. Пользователь выполняет следующие действия:

* выбирает категорию товаров из доступного списка;
* выбирает торговую точку из доступного списка, для которой будут генерироваться рекомендации;
* выбирает фиксированное количество рекомендаций, а именно, 3, 6, 9 или 12 товаров.

1. Форма отправки данных в модель рекомендательной системы. После заполнения параметров пользователь отправляет запрос в модель машинного обучения:

* передача подготовленных параметров в модель для анализа;
* обработка запроса с использованием гибридного алгоритма рекомендательной системы;
* получение и отображение результатов в виде списка рекомендованных товаров.

1. Внешние системы. Диаграмма также отражает интеграцию с внешними системами, такими как базы данных товаров и историями закупок торговых точек. Это позволяет системе учитывать актуальную информацию при формировании рекомендаций.
2. Ядром системы, где происходит обработка данных и генерация рекомендаций, является модель машинного обучения.

**Диаграмма сущностных классов**

Диаграмма сущностных классов в терминологии *UML* представляет собой визуализацию набора сущностей и связей между ними. Она может также содержать атрибуты и операции классов, что позволяет более детально описать их структуру и функциональность [9].

Атрибут класса – это именованное свойство, которое описывает множество значений, присущих экземплярам данного класса. Атрибуты отражают свойства моделируемой сущности, общие для всех её объектов, и являются абстракцией состояния объекта. Каждый атрибут любого объекта должен иметь определённое значение. Класс может обладать произвольным числом атрибутов или не иметь их вовсе [10].

Операция класса – это именованная услуга, которую можно запросить у объекта данного класса. Это абстракция действий, применимых к объекту. Как и в случае с атрибутами, класс может содержать любое количество операций, включая их полное отсутствие. Набор операций класса одинаков для всех объектов этого класса [10].

На рисунке 5 представлена диаграмма сущностных классов.

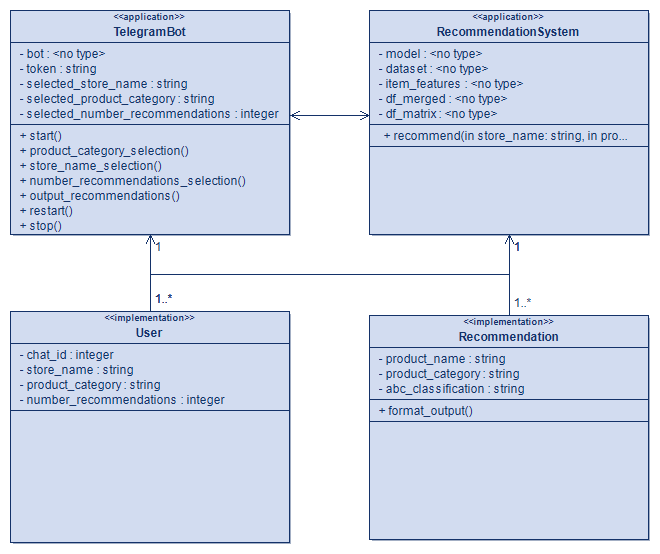


Рисунок 5 – Диаграмма сущностных классов

На диаграмме сущностных классов представлены ключевые компоненты системы, их атрибуты и взаимосвязи. Система состоит из двух основных модулей: *TelegramBot* и *RecommendationSystem*, которые взаимодействуют с сущностями *User* и *Recommendation*.

1. *TelegramBot*. Этот модуль отвечает за взаимодействие с пользователем (торговым представителем) через *Telegram*-интерфейс.

Атрибуты:

* *token*: *string* – уникальный идентификатор бота в *Telegram*;
* *selected*\_*store*\_*name*: *string* – название выбранной торговой точки;
* *selected*\_*product*\_*category*: *string* – выбранная категория товаров;
* *selected*\_*number*\_*recommendations*: *integer* – количество рекомендаций.

Методы:

* *start*() – инициализация бота;
* *product*\_*category*\_*selection*() – выбор категории товаров;
* *store*\_*name*\_*selection*() – выбор торговой точки;
* *number*\_*recommendations*\_*selection*() – задание количества рекомендаций;
* *output*\_*recommendations*() – вывод сформированных рекомендаций;
* *restart*() – сброс параметров для нового запроса;
* *stop*() – завершение работы бота.

1. *RecommendationSystem*. Ядро системы, которое обрабатывает данные и генерирует рекомендации.

Атрибуты:

* *model* – модель машинного обучения;
* *dataset* – набор данных для обучения;
* *item*\_*features* – характеристики товаров;
* *df*\_*merged* – объединённые данные для анализа;
* *df*\_*matrix* – матрица взаимодействий.

Методы:

* *recommend*() – формирование рекомендации на основе введённых параметров.

1. *User*. Сущность, представляющая пользователя системы.

Атрибуты:

* *chat*\_*id*: *integer* – идентификатор чата в *Telegram*;
* *store*\_*name*: *string* – название торговой точки;
* *product*\_*category*: *string* – выбранная категория товаров;
* *number*\_*recommendations*: *integer* – количество запрашиваемых рекомендаций.

1. Recommendation. Сущность, описывающая рекомендуемый товар.

Атрибуты:

* *product*\_*name*: *string* – название товара;
* *product*\_*category*: *string* – категория товара;
* *abc*\_*classification*: *string* – класс товара по *ABC*-анализу.

Методы:

* *format*\_*output*() – форматирует вывод рекомендации для удобного отображения пользователю.

*TelegramBot* взаимодействует с *User*:

* получает от пользователя параметры (*store*\_*name*, *product*\_*category*, *number*\_*recommendations*);
* передаёт их в *RecommendationSystem* для обработки.

*RecommendationSystem* использует данные из *User*, чтобы сгенерировать объекты *Recommendation*

*Recommendation* возвращается в *TelegramBot*, где метод *output*\_*recommendations*() отображает результаты пользователю.

**Динамическое моделирование**

Динамическая модель представляет собой теоретическую конструкцию, которая описывает изменения состояний объекта во времени. Она может включать описание этапов или фаз жизненного цикла системы, диаграммы состояний подсистем, а также использоваться для анализа поведения сложных систем. Часто такая модель имеет математическое выражение, что позволяет формализовать процессы изменения и применять её в различных областях науки.

Первоначально динамические модели широко использовались в общественных науках, таких как социология и экономика, для исследования систем, подверженных постоянным изменениям. Однако современная научная парадигма существенно расширила их применение, и сегодня такие модели востребованы в естественных и технических науках, включая физику, биологию, инженерное дело и информационные технологии [10].

Динамическая модель нередко включает элементы энергетического анализа, где процессы представлены через математические операции суммирования, интегрирования и дифференцирования. В программной инженерии она также может быть дополнена другими типами диаграмм, такими как диаграммы последовательностей, которые отображают порядок взаимодействий между объектами, или диаграммы кооперации, демонстрирующие взаимосвязь объектов в контексте выполнения задач. Диаграммы состояний, как часть динамического моделирования, позволяют наглядно представить переходы между различными состояниями системы, что важно для анализа сложных программных комплексов [10].

**Диаграмма активностей**

Диаграмма активностей – это графический инструмент, предназначенный для моделирования процессов и потоков управления в системе. Она широко используется в системном и программном проектировании, в частности в *UML*, для описания последовательности действий, выполняемых системой или её элементами. Диаграммы активностей позволяют наглядно представить логику бизнес-процессов, алгоритмов и рабочих потоков [9].

Основными элементами диаграммы активностей являются действия (или активности), представляющие собой элементарные операции, и переходы, определяющие порядок их выполнения. Активность может быть как простой, так и составной, включающей вложенные действия. Переходы указывают направление потока управления от одного действия к другому и могут быть условными, то есть выполняться при соблюдении определённых условий [9].

Начальная точка (*initial* *node*) обозначает начало выполнения процесса, а конечные узлы (*final* *nodes*) указывают на завершение выполнения. Также используются управляющие конструкции, такие как ветвления (*decision* *nodes*), параллельное выполнение (*fork* *nodes*) и синхронизация (*join* *nodes*), что позволяет моделировать сложные процессы с альтернативными и параллельными потоками [9].

На рисунке 6 представлена диаграмма состояний.

Процесс начинается с инициализации, где пользователь запускает систему и получает доступ к форме выбора. Далее последовательно выполняются следующие действия:

* выбор категории товара из доступного списка;
* выбор торговой точки, для которой необходимо сгенерировать рекомендации;
* выбор количества рекомендаций;
* генерация рекомендаций, при которой система отправляет введённые параметры в обученную модель и получает результат;
* отображение полученного списка товаров, подходящих по заданным критериям;
* отображена вариативность, где пользователь после вывода всех введённых параметров может приступить к генерации рекомендаций или же заполнить параметры заново.

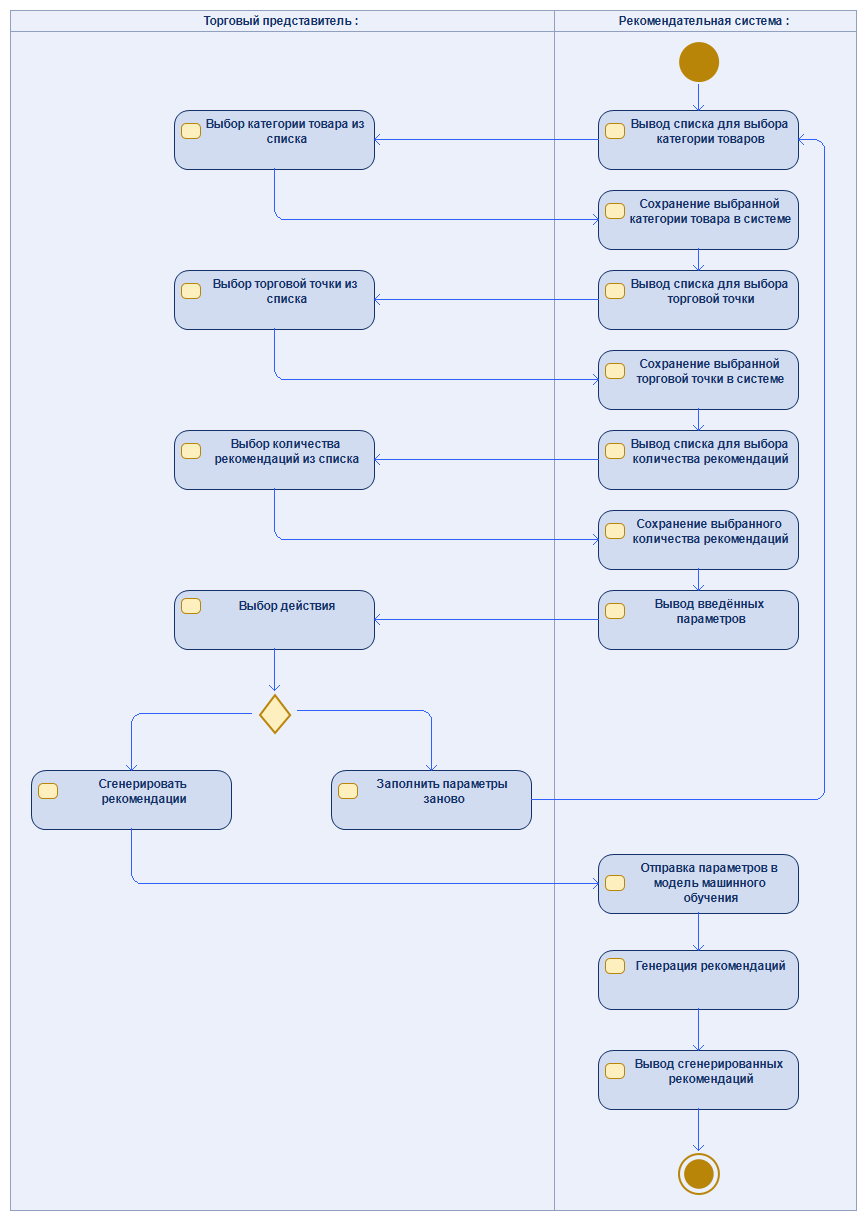


Рисунок 6 – Диаграмма активностей

**Диаграмма кооперации**

Диаграмма кооперации – это один из видов диаграмм, используемых в объектно-ориентированном моделировании для отображения взаимодействий между объектами системы в контексте выполнения конкретного прецедента. На этой диаграмме показаны только те объекты, которые участвуют в описываемом прецеденте. Некоторые объекты могут быть представлены только на одной диаграмме кооперации, в то время как другие могут появляться сразу на нескольких диаграммах [8].

Основная цель диаграммы кооперации – продемонстрировать последовательность взаимодействий между объектами системы, что позволяет лучше понять, как данные объекты взаимодействуют друг с другом в рамках конкретной задачи или функции. Для отображения последовательности этих взаимодействий используются порядковые номера сообщений, которые отражают точный порядок обмена информацией. Порядок сообщений на диаграмме кооперации должен соответствовать той последовательности взаимодействий между актёром и системой, которая описана в прецеденте [9].

На рисунке 7 представлена диаграмма кооперации для прецедента «Выбор категории товара».

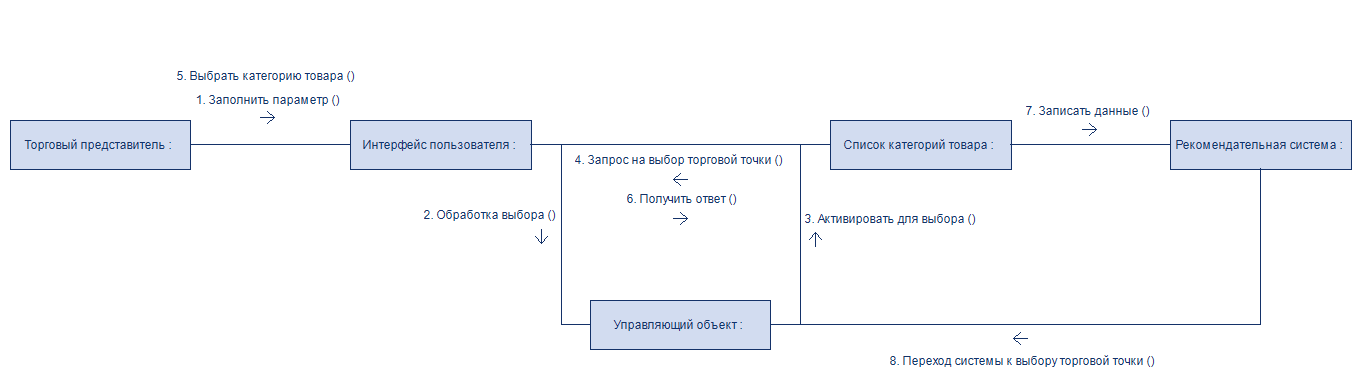


Рисунок 7 – Диаграмма кооперации для прецедента «Выбор категории товара»

На диаграмме представлено взаимодействие между торговым представителем, пользовательским интерфейсом и системой. После инициализации сессии пользователь нажимает кнопку «Получить рекомендации», система выводит список категорий, из которых пользователь выбирает нужную. Система сохраняет введённые параметры и подготавливает переход к следующему этапу – выбору торговой точки.

На рисунке 8 представлена диаграмма кооперации для прецедента «Выбор торговой точки».

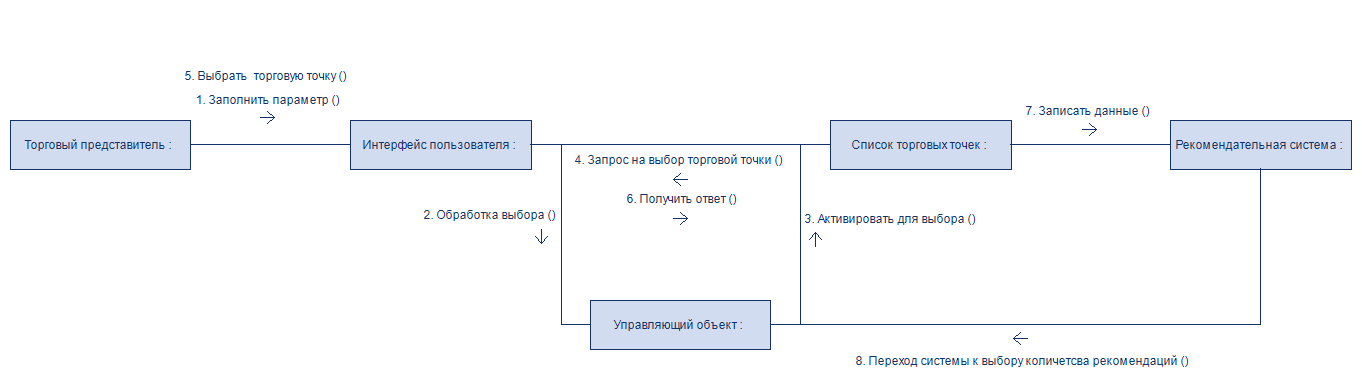


Рисунок 8 – Диаграмма кооперации для прецедента «Выбор торговой точки»

На диаграмме показана последовательность действий при выборе торговой точки. После выбора категории система запрашивает у пользователя торговую точку, принимает введённое значение, сохраняет его в соответствующий атрибут и инициирует переход к этапу выбора количества рекомендаций.

На рисунке 9 представлена диаграмма кооперации для прецедента «Выбор количества рекомендаций».

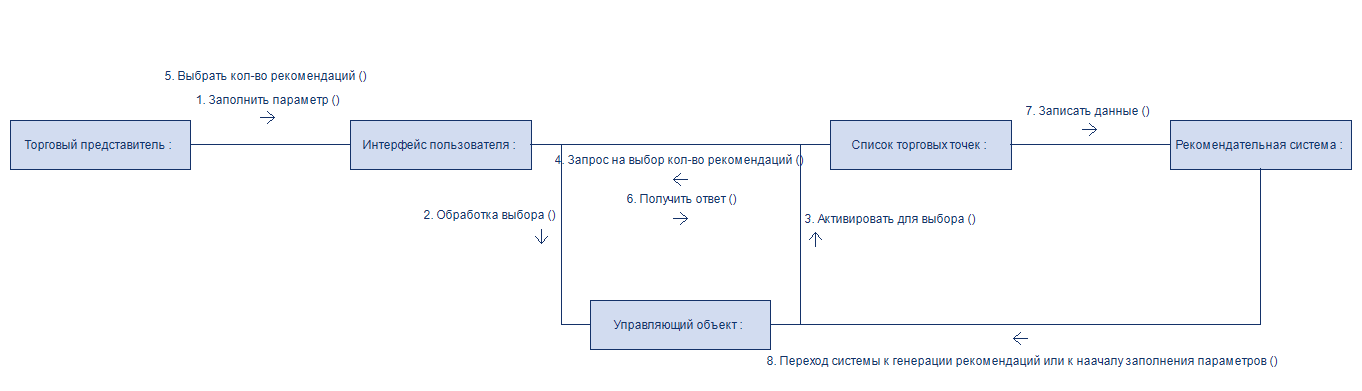


Рисунок 9 – Диаграмма кооперации для прецедента «Выбор количества рекомендаций»

На диаграмме отражён механизм выбора количества рекомендаций. Система получает от пользователя числовой параметр, сохраняет значение и передаёт управление к форме, где пользователь может либо сгенерировать рекомендации, либо заполнить параметры заново.

На рисунке 10 представлена диаграмма кооперации для прецедента «Генерация рекомендаций».

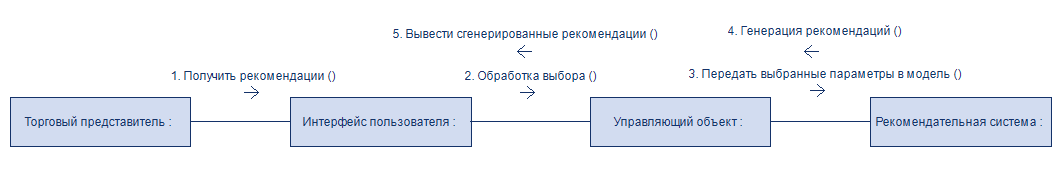


Рисунок 10 – Диаграмма кооперации для прецедента «Генерация рекомендаций»

На диаграмме изображено взаимодействие при генерации рекомендаций. Пользователь инициирует генерацию, система формирует запрос на основе введённых параметров и передаёт его в модель машинного обучения. Система анализирует данные, получает от модели рекомендации и выводит их пользователю.

На рисунке 11 представлена диаграмма кооперации для прецедента «Заполнение параметров заново».

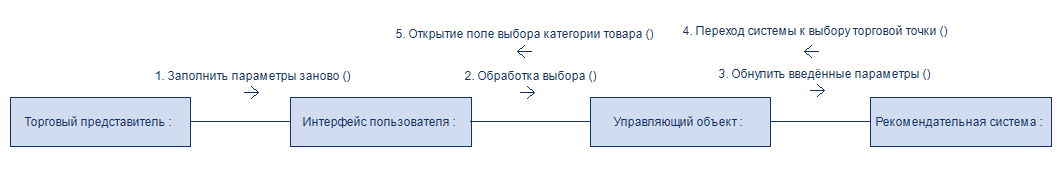


Рисунок 11 – Диаграмма кооперации для прецедента «Заполнение параметров заново»

На диаграмме показан процесс сброса параметров. После выбора параметров пользователь может выбрать опцию «Заполнить заново». Система очищает все ранее введённые данные (категорию, торговую точку и количество рекомендаций), и возвращает пользователя к начальному этапу – выбору категории.

2.5 Выбор технологий для проектирования рекомендательной системы

Выбор технологий для проектирования системы является важным этапом, т. к. от этого зависит эффективность, производительность и масштабируемость системы. Будут рассмотрены ключевые технологии, которые будут использоваться для реализации системы, а именно язык программирования, библиотеки для машинного обучения, обработки данных, создания *Telegram*-бота и сериализации модели.

**Выбор языка программирования**

Выбор языка программирования определяет не только скорость разработки, но и возможности интеграции с существующей инфраструктурой, а также доступность инструментов для машинного обучения. Ниже сравниваются популярные языки программирования, такие как *Python*, *R* и *Java*, с точки зрения их применимости в контексте рекомендательных систем.

* *Python* – это высокоуровневый язык программирования общего назначения, который используется для разработки веб-приложений, анализа данных, искусственного интеллекта, автоматизации задач и т. п. *Python* стал одним из самых популярных языков программирования благодаря своей простоте, читаемости, универсальности и широкому спектру применения. Он используется в различных областях, начиная от веб-разработки и заканчивая научными исследованиями [11];
* *R* – это функциональный язык программирования с элементами объектно-ориентированного подхода. Технические особенности *R* во многом определяют его позиционирование как специализированного инструмента для анализа данных [12];
* *Java* – это высокоуровневый объектно-ориентированный язык программирования, который помогает разработчикам создавать программное обеспечение для мобильных устройств, *POS*-терминалов, банкоматов, *IoT* и веб-страниц [13].

Сравнение языков программирования представлена в таблице 12.

Таблица 12 – Сравнение языков программирования

|  |  |  |  |
| --- | --- | --- | --- |
| Критерий | *Python* | *R* | *Java* |
| Скорость разработки | Высокая | Средняя | Низкая |
| Поддержка *ML* | Да, например, *LightFM* | Да, например, *Сaret* | Требуется интеграция сторонних решений |
| Интеграция | Легко интегрируется, например, с *API* | Сложно интегрируется с веб-сервисами | Высокая с последующей настройкой |
| Читаемость кода | Простой и понятный синтаксис | Специфичный синтаксис | Сложный синтаксис |

Для проектирования был выбран язык программирования *Python* по следующим причинам:

* быстрое построение модели машинного обучения, в нашем случае *LightFM*;
* готовая экосистема для обработки данных, например, *Pandas*;
* простота интеграции с *API*, в нашем случае, Telegram *API*.

**Выбор библиотеки для машинного обучения**

Библиотеки для машинного обучения играют ключевую роль в создании рекомендательной системы, так как от их функциональности зависит точность и качество рекомендаций. Ниже анализируются такие библиотеки, как *LightFM*, *Surprise* и *TensorFlow* на их преимущества и ограничения.

* *LightFM* – это библиотека для *Python*, которая поддерживает гибридные рекомендательные системы. Она позволяет комбинировать коллаборативную и контентную фильтрацию, что делает её мощным инструментом для создания рекомендаций. *LightFM* предоставляет различные алгоритмы и методы для обучения моделей и оценки их качества [14];
* *Surprise* – это библиотека на *Python*, предназначенная для построения и анализа рекомендательных систем. Библиотека поддерживает различные алгоритмы коллаборативной фильтрации и позволяет легко экспериментировать с ними. *Surprise* предоставляет удобные инструменты для работы с данными, обучения моделей и оценки их качества [14];
* *TensorFlow* – это мощная и гибкая библиотека машинного обучения, разработанная компанией *Google*. Библиотека широко используется для создания и обучения нейронных сетей, а также для выполнения сложных вычислений. *TensorFlow* поддерживает как обучение на *CPU*, так и на *GPU*, что делает её идеальным инструментом для разработки моделей машинного обучения различной сложности. Библиотека поддерживает много языков программирования такие как, *Python*, *C*++ и *JavaScript*, *TensorFlow* [15].

Сравнение библиотек для машинного обучения представлены в таблице 13.

Таблица 13 – Сравнение библиотек для машинного обучения

|  |  |  |  |
| --- | --- | --- | --- |
| Критерий | *LightFM* | *Surprise* | *TensorFlow* |
| Гибкость | Поддержка гибридных моделей | Поддержка только коллаборативных моделей | Требует кастомной реализации |
| Производительность | Средняя | Средняя | Высокая |
| Интерпретируемость | Встроенные метрики | Сложность интерпретации | Ограниченные метрики |

Для проектирования была выбрана библиотека *LightFM* по следующим причинам:

* возможность использования гибридного подхода;
* эффективная работа с разрежёнными матрицами;
* встроенные метрики для оценки качества модели.

**Выбор библиотеки для обработки данных**

Обработка данных – один из ключевых этапов в работе рекомендательной системы. Ниже рассматриваются библиотеки *Pandas* и *Dask*, которые позволяют эффективно управлять данными, обеспечивая их подготовку для дальнейшего анализа и обучения модели.

*Pandas* – это библиотека являющаяся одной из самых популярных библиотек для работы с данными в *Python*. Она предоставляет мощные инструменты для манипуляции и анализа данных [16];

*Dask* – это библиотека для параллельной и распределенной обработки данных, которая позволяет работать с данными, превышающими объем оперативной памяти [16].

Сравнение библиотек для обработки данных представлены в таблице 14.

Таблица 14 – Сравнение библиотек для обработки данных

|  |  |  |
| --- | --- | --- |
| Критерий | *Pandas* | *Dask* |
| Удобство анализа | Высокое, имеется интерактивный режим | Высокое, но сложнее настройки |
| Производительность | Быстрый для средних данных | Оптимизирован для больших данных |
| Интеграция с ML | Прямая совместимость | Совместим, но требует адаптации |

Для проектирования была выбрана библиотека *Pandas* по следующим причинам:

* интерактивная обработка данных;
* совместимость с библиотекой *LightFM.*

**Выбор библиотеки для разработки *Telegram*-бота**

*Telegram*-бот является удобным интерфейсом для взаимодействия пользователей с рекомендательной системой. Ниже сравниваются библиотеки *Telegram* *Bot* *API* и *Aiogram*, чтобы определить оптимальное решение для реализации интуитивного и функционального бота.

* *Telegram* *Bot* *API* – это набор методов для управления *Telegram*-ботами. С его помощью можно обрабатывать сообщения пользователей, автоматизировать модерацию групповых чатов и создавать полезные утилиты на базе мессенджера;
* *Aiogram* – это мощная и удобная библиотека для создания *Telegram*-ботов на языке программирования *Python*. Она позволяет легко и быстро разрабатывать ботов, используя асинхронные функции и удобный интерфейс [17].

Сравнение библиотек для написания *Telegram*-бота представлены в таблице 15.

Таблица 15 – Сравнение библиотек для написания *Telegram*-бота

|  |  |  |
| --- | --- | --- |
| Критерий | *Telegram* *Bot* *API* | *Aiogram* |
| Скорость разработки | Минимальный код для старта | Асинхронность делает код сложнее |
| Функциональность | Базовая, например, кнопки, меню и т. п. | Поддержка *FSM* |
| Поддержка | Средняя | Высокая |

Для проектирования была выбрана библиотека *Telegram* *Bot* *API* по следующим причинам:

* быстрое создание интуитивного интерфейса;
* минимальные затраты на развёртывание;
* необязательное использование сервера.

**Выбор библиотеки для сериализации модели**

Сериализация модели необходима для сохранения и повторного использования обученной модели без необходимости её переобучения. Ниже рассматриваются библиотеки *Pickle* и Joblib, которые обеспечивают эффективное хранение и загрузку моделей машинного обучения.

* *Pickle* – это модуль *Python*, который предоставляет возможность сериализовать и десериализовать объекты Python. Сериализация – это процесс преобразования объекта в поток байтов, который затем может быть сохранен в файл или передан через сеть. Десериализация – это обратный процесс, при котором поток байтов преобразуется обратно в объект [18];
* *Joblib* – это *Python*‑библиотека, которая предоставляет инструменты для параллельных вычислений, кэширования и эффективной обработки данных. Она используется для ускорения выполнения операций, таких как многократные вычисления, обработка больших массивов данных и параллельная обработка однотипных задач [19].

Сравнение библиотек для сериализации модели представлены в таблице 16.

Таблица 16 – Сравнение библиотек для сериализации модели

|  |  |  |
| --- | --- | --- |
| Критерий | *Pickle* | *Joblib* |
| Универсальность | Подходит для любых объектов | В основном предназначен для *NumPy* |
| Скорость | Средняя | Высокая, но для массивов |
| Безопасность | Средняя | Высокая |

Для проектирования была выбрана библиотека *Pickle* по следующим причинам:

* простота использования;
* совместима с другими библиотеками, используемыми при проектировании.

2.6 Обоснование выбора проектных решений

При разработке рекомендательной системы для сибирской логистической компании были приняты проектные решения, направленные на обеспечение эффективности, точности и удобства использования системы.

Для достижения точного прогнозирования товаров, которые с большей долей вероятности закупят торговые точки было принято решение о разработке гибридной рекомендательной системы, сочетающей коллаборативную и контентную фильтрации.

Коллаборативная фильтрация, основанная на истории закупок торговых точек, позволяет выявлять скрытые паттерны спроса и учитывать индивидуальное покупательское поведение.

Контентная фильтрация, в свою очередь, учитывает характеристики товаров, такие как категория и *ABC*-классификация, что обеспечивает универсальность рекомендаций.

*Python* был выбран в качестве основного языка программирования благодаря своей гибкости, широкому набору библиотек для анализа данных и машинного обучения.

Для реализации рекомендательной системы была выбрана библиотека *LightFM*. *LightFM* обеспечивает эффективную работу с разрежёнными матрицами данных, что особенно важно при анализе больших объёмов данных о закупках, а также предоставляет встроенные метрики для оценки качества модели. *LightFM* даёт возможность реализации гибридного подхода к рекомендациям, что соответствует выбранной архитектуре системы.

Для обработки и манипулирования табличными данными была выбрана библиотека *Pandas*. *Pandas* предоставляет удобные структуры данных и инструменты для анализа данных, а также совместима с библиотекой *LightFM*, что упрощает интеграцию компонентов системы.

*Telegram*-бот был выбран в качестве интерфейса взаимодействия пользователя с рекомендательной системой ввиду его популярности, удобства использования и широкой доступности. Для разработки *Telegram*-бота была использована библиотека *Telegram* *Bot* *API*, предоставляющая удобные инструменты для создания и управления ботами.

Для сериализации модели машинного обучения и обеспечения возможности её сохранения и загрузки был использован модуль *Pickle*. *Pickle* позволяет эффективно сохранять и восстанавливать объекты *Python*, что необходимо для интеграции модели с *Telegram*-ботом и обеспечения переносимости системы.

Выбранные проектные решения обеспечивают создание эффективной, точной и удобной в использовании рекомендательной системы, отвечающей потребностям компании.

2.7 Выводы по главе

В данной главе были определены требования к разрабатываемой рекомендательной системе, как функциональные, так и нефункциональные. Внимание было уделено описанию пользовательских сценариев взаимодействия с системой, что позволило формализовать основные действия, которые должен будет выполнять пользователь при работе с программным продуктом. Составлены диаграммы прецедентов и спецификации вариантов использования, что обеспечило полное описание бизнес-логики системы.

Также в главе были разработаны статические и динамические модели, включающие диаграммы классов, активностей и кооперации. Эти модели позволили детально отразить внутреннюю архитектуру будущей системы, связи между её элементами и логику их взаимодействия. Кроме того, произведён обоснованный выбор технологий: языка программирования *Python*, библиотеки *LightFM* для реализации рекомендательной модели, *Pandas* для обработки данных, а также библиотеки *Telebot* для создания пользовательского интерфейса через *Telegram*.

3 Разработка рекомендательной системы

3.1 Анализ данных для обучения модели

Для обучения модели рекомендательной системы будут использоваться данные о продажах товаров в торговых точках, а также характеристиках товаров. Этот набор данных поможет выявить закономерности в предпочтениях торговых точках и формировать персонализированные рекомендации.

Данные для обучения делятся на две категории:

1. Данные о характеристиках товаров, которые позволяют определить популярность товаров и их значимость для компании. Эти данные имеют следующие параметры:
   1. Наименование товара.
   2. Категория товара.
   3. Сумма продаж.
   4. Процент от общей суммы продаж.
   5. Накопительная сумма.

Часть данных о характеристиках товаров представлена в таблице 17.

Таблица 17 – Часть данных о характеристиках товаров

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *product* *name* | *product* *category* | *total* *realization* | *share* *realization* | *accumulation* *realization* |
| Аленка (ОК) 90г | Шоколад | 71201224,45 | 0,01156 | 0,01156 |
| РОЛЛТОН ВЕРМИШЕЛЬ б/п витамин. куриная на домашнем бульоне, 60гр | Продукты быстрого приготовления | 64360762,22 | 0,01045 | 0,02201 |
| МАКФА МУКА пшеничная в/с, 2кг.Челябинск | Мука | 49262315,82 | 0,00800 | 0,03001 |
| МАКФА Спираль макар.изд., в/с, 223-3А 400гр. | Макаронные изделия | 47573899,91 | 0,00773 | 0,03774 |
| В шок. Москвичка БК | Карамель | 32893871,3 | 0,00534 | 0,04308 |
| Халва подсолнечная Тимоша | Халва | 32521128,58 | 0,00528 | 0,04836 |
| Батон Бабаевский с помадно-сливочной начинкой (ОК) 50г | Шоколад | 32311342,78 | 0,00525 | 0,05361 |
| Батончик Рот-Фронт в/з 250г | Конфеты фасованные | 32181866,51 | 0,00523 | 0,05883 |
| Нури (пак) 100\*2г (0201) Высок ДАБЛ | Чай | 32143053,67 | 0,00522 | 0,06405 |
| Мореход бк2 | Конфеты | 30926215,88 | 0,00502 | 0,06907 |
| Люкс (ОК) 90г | Шоколад | 29928414,59 | 0,00486 | 0,07393 |

Окончание таблицы 17

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *product* *name* | *product* *category* | *total* *realization* | *share* *realization* | *accumulation* *realization* |
| Бушидо Оригинал крист.100г | Кофе | 27574027,28 | 0,00448 | 0,07841 |
| РОЛЛТОН ВЕРМИШЕЛЬ б/п витамин. в говяжьем бульоне, 60гр | Продукты быстрого приготовления | 26534105,73 | 0,00431 | 0,08272 |
| Канди (Лист) 250г (0052) Медиум | Чай | 25121995,81 | 0,00408 | 0,08680 |
| Аленка (ОК) 200г | Шоколад | 24821582,32 | 0,00403 | 0,09083 |
| ЗЛАТО подсолнечное рафин.дезод. масло, 1 сорт, 1л | Масла растительные | 24642335,53 | 0,00400 | 0,09483 |
| Батончик Рот-Фронт в/з | Конфеты | 23153206,32 | 0,00376 | 0,09859 |
| Аленка (ОК) 15г | Шоколад | 22696294,75 | 0,00369 | 0,10227 |
| Левушка с карамельной нач. | Конфеты | 22451966,77 | 0,00365 | 0,10592 |
| ЗОЛОТАЯ СЕМЕЧКА подсолнечное рафин.дезор.масло 1 сорт,1л | Масла растительные | 22112057,93 | 0,00359 | 0,10951 |
| Кара-кум в/н | Конфеты | 21851349,47 | 0,00355 | 0,11306 |
| Кола 2л ПЭТ | Соки, напитки | 21411738,6 | 0,00348 | 0,11654 |
| Халва Рот-Фронт арахисовая глазир. в/з | Халва | 20565164,13 | 0,00334 | 0,11987 |
| МАКФА МУКА пшеничная в/с, 1кг. Челябинск | Мука | 20556377,21 | 0,00334 | 0,12321 |

1. Данные о закупках товаров торговыми точками, которые позволяют определить какие товары в какой торговой точке востребованы, а также найти скрытые факторы. Эти данные имеют следующие параметры:
   1. Название торговой точки.
   2. Наименование товара.
   3. Количество продаж.

Часть данных о закупках товаров торговыми точками представлена в таблице 18

Таблица 18 – Часть данных о закупках товаров торговыми точками

|  |  |
| --- | --- |
| *sales* *outlet*/*goods* | *quantity* |
| 1 Мая пер, 3а, Магазин "Колосок" |  |
| Армавир, п/к, МГС 1,8-2,2кг, Вербицкие ООО | 2,16 |
| Артек плюс ПКФ | 4,50 |
| Баранка яичная 400гр | 5,00 |
| Голень копчено-вареная Классическая МГС 1.5-1.9 кг Вербицкие ООО | 1,04 |
| Грудинка по-домашнему охл в/у, ТК Браво | 0,79 |
| Десерт с творожным вкусом (С) | 1,20 |
| Заливное Курица (куренок ободок) 250гр | 1,00 |
| Заливное Чахохбили (куренок) 250гр | 1,00 |
| Зефир Ассорти 270г | 2,00 |
| Зефир сгущ.молоко глазир.декор | 2,50 |
| Коктейль молочный со вкусом голубики м.д.ж 3,2 % ТМ Село Зеленое 0,2 мл | 1,00 |
| Коктейль молочный со вкусом груши м.д.ж 3,2 % ТМ Село Зеленое 0,2 мл | 1,00 |
| Коктейль молочный со вкусом малины м.д.ж 3,2 % ТМ Село Зеленое 0,2 мл | 1,00 |
| Колбаса вареная Болонская с чесноком ТМ Delikaiser 400 гр | 1,00 |
| Колбаса вареная Докторская высокий стандарт Атяшево 400 гр | 1,00 |
| Колбаса вареная молочная Высокий стандарт Атяшево 400 гр | 1,00 |
| Колбаса ливерная Печеночная 0,25кг | 2,00 |
| Колбаски Печеночные 0,25кг | 2,00 |
| Оригинальная 0,5кг ТМ Доступно | 1,00 |
| Особая 0,4 кг ТМ Дюжий Двор | 2,00 |
| Особая со шпиком 0,4 кг ТМ Дюжий Двор | 2,00 |
| Пельмени Любимая порция с бульоном 800гр | 2,00 |
| Пряник Мятный 400гр | 3,00 |

Для данных о характеристиках товаров был проведён анализ, и обработка данных в процессе которых было сделано:

1. Текстовые поля приведены к нижнему регистру.
2. Объединение некоторых категорий товаров.
3. Удалены товары с отрицательной реализацией.
4. Уделаны редкие категории товаров с малым количеством товаров.
5. На основе значения накопительной суммы была произведена *ABC*-классификация, где:

* товары с долей реализации до 80% были отнесены к классу *A*;
* товары с долей реализации от 80% до 95% – к классу *B*;
* товары с долей реализации выше 95% – к классу *C.*

Результат анализа и обработки данных о характеристиках товаров представлен в таблице 19.

Таблица 19 – Результат анализа и обработки данных о характеристиках товаров

|  |  |  |
| --- | --- | --- |
| *product*\_*name* | *product*\_*category* | *abc*\_*classification* |
| аленка (ок) 90г | шоколад | A |
| роллтон вермишель б/п витамин. куриная на домашнем бульоне, 60гр | продукты быстрого приготовления | A |
| макфа мука пшеничная в/с, 2кг.челябинск | мука | A |
| макфа спираль макар.изд., в/с, 223-3а 400гр. | макаронные изделия | A |
| в шок. москвичка бк | карамель | A |
| халва подсолнечная тимоша | к чаю | A |
| батон бабаевский с помадно-сливочной начинкой (ок) 50г | шоколад | A |
| батончик рот-фронт в/з 250г | конфеты | A |
| нури (пак) 100\*2г (0201) высок дабл | чай | A |
| мореход бк2 | конфеты | A |
| люкс (ок) 90г | шоколад | A |
| бушидо оригинал крист.100г | кофе | A |
| роллтон вермишель б/п витамин. в говяжьем бульоне, 60гр | продукты быстрого приготовления | A |
| канди (лист) 250г (0052) медиум | чай | A |
| аленка (ок) 200г | шоколад | A |
| злато подсолнечное рафин.дезод. масло, 1 сорт, 1л | растительные масла | A |
| батончик рот-фронт в/з | конфеты | A |
| аленка (ок) 15г | шоколад | A |
| левушка с карамельной нач. | конфеты | A |
| золотая семечка подсолнечное рафин.дезор.масло 1 сорт,1л | растительные масла | A |
| кара-кум в/н | конфеты | A |
| кола 2л пэт | напитки | A |
| халва рот-фронт арахисовая глазир. в/з | к чаю | A |
| макфа мука пшеничная в/с, 1кг. челябинск | мука | A |

Код обработки и анализа данных о характеристиках товаров представлен в листинге 1.

|  |
| --- |
| Листинг 1 – Код обработки и анализа данных о характеристиках товаров |
| # Функция обработки датасета с характеристиками товаров  def preparation\_goods\_characteristics\_data(df):  # Привидение текстовых переменных к нижнему регистру  df[['product\_name', 'product\_category']] = df[['product\_name', 'product\_category']].apply(lambda x: x.str.lower())  # Словарь для объединения категорий  category\_mapping = {  'конфеты': ['конфеты', 'конфеты фасованные', 'конфеты в коробках'],  'шоколад': ['шоколад', 'шоколад весовой', 'шоколодные и ореховые пасты'],  'печенье': ['печенье', 'печенье фасованное', 'крекер'],  'вафли': ['вафли', 'вафли фасованные'],  'карамель': ['карамель', 'карамель фасованная'],  'чай': ['чай', 'хорека-чай'],  'кофе': ['кофе', 'хорека-кофе'],  'консервы': ['консервы мясные', 'консервы овощные', 'консервы рыбные и морепродукты', 'консервы фруктово-ягодные'],  'молоко сгущенное': ['молоко сгущенное и концентрированное'],  'мясные продукты': ['деликатесы мясные', 'мясо', 'мясо охлажденное', 'холодец'],  'полуфабрикаты': ['полуфабрикаты', 'готовые блюда', 'паназиатская кухня'],  'хлебобулочные изделия': ['хлеб, хлебобулочные изделия'],  'к чаю': ['баранки, соломка, сушка, сухари', 'кексы', 'рулеты', 'зефир', 'мармелад', 'пастила', 'восточные сладости', 'козинаки', 'халва'],  'десерты': ['десерты', 'торты ,бисквиты,пирожные', 'хорека-коржи, торты'],  'напитки': ['соки, напитки'],  'крупы и зерновые': ['крупы', 'каши'],  'растительные масла': ['масла растительные'],  'соусы и приправы': ['кетчуп и соусы', 'томатная паста'],  'приправы': ['специи, приправы', 'смеси для гарнира'],  'продукты для выпечки': ['сода, уксус, дрожжи'],  'сухие завтраки': ['сухие завтраки', 'хлопья'],  'замороженные продукты': ['замороженные овощи, ягоды, грибы'],  'овощи/фрукты': ['овощи,фрукты грибы,ягоды'],  'детские товары': ['детское питание', 'детские товары'],  'полезное питание': ['полезное питание', 'полезное питание , продукция без сахара'],  'снэки': ['снэки', 'жевательная резинка'],  'новогодние товары': ['новогодние подарки нк', 'новогодние подарки', 'новогодняя упаковка/сайт', 'новогодняя упаковка'],  'прочее': ['прочие', 'прочие нк(швейный цех)', 'нпу 2024', 'янеиспользуемый', 'остатки сладки', 'элитная продукция', 'подарки'],  'товары для дома': ['хозяйственные товары', 'бытовая химия'],  'личная гигиена': ['личная гигиена', 'косметика, гигиена'],  'табак': ['табак, спички', 'табак собственные магазины'],  'рыба': ['рыба', 'рыбные пресервы'],  'яйца': ['яйцо'],  'орехи/сухофрукты': ['орехи, сухофрукты'],  'животные': ['товары для животных'],  'собственные магазины': ['собственные магазины']  }  # Функция для объединения категорий товаров  def replace\_categories(df, column\_name):  # Обратный словарь для замены, где ключ - подкатегория, значение - основная категория  reverse\_mapping = {}  # Заполнение обратного словаря  # Проход по категориям товаров  for categories, subcategories in category\_mapping.items():  # Проход по подкатегориям категории товара  for subcategory in subcategories:  # Заполнения обратного словаря категорями товаров на основе подкатегории товаров  reverse\_mapping[subcategory] = categories  # Заполнение стобца датасета категориями из обратного словаря  df.loc[:, column\_name] = df[column\_name].map(reverse\_mapping).fillna(df[column\_name])  return df  # Приминение функции замены категорий товаров  df = replace\_categories(df, 'product\_category')  # Удаление товаров с отрицательной суммой реализации  df = df.loc[df['total\_realization'] > 0].copy()  # Удаление категорий товаров, у которых количество товаров меньше 10  # Подсчёт количества товаров по категориям  category\_counts = df['product\_category'].value\_counts()  # Выбор категорий с менее чем 10 товарами  rare\_categories = category\_counts[category\_counts < 10].index  # Удаление строк с редкими категориями из датасета  df = df.loc[~df['product\_category'].isin(rare\_categories)].copy()  # Преобразования накопительной суммы реализации в проценты  df['accumulation\_realization'] = df['accumulation\_realization'].apply(lambda x: round((x \* 100), 2))  # Функция ABC-классификации товаров  def abc\_classification(accumulation\_realization):  if accumulation\_realization < 80:  return 'A'  elif 80 <= accumulation\_realization <= 95:  return 'B'  else:  return 'C'  # Создание нового столбца с ABC-классификацией товаров  df['abc\_classification'] = df['accumulation\_realization'].apply(abc\_classification)  # Удаление столбцов датасета  df = df.drop(['share\_realization', 'accumulation\_realization'], axis = 1)  # Возврат обработанного датасета с характеристиками товаров  return df |

Для данных о закупках товаров торговыми точками был проведён анализ, и обработка данных в процессе которых было сделано:

1. Был реализован алгоритм, позволяющий отделить строки с наименованиями магазинов от товарных записей с присвоением каждому товару соответствующую торговую точку.
2. Преобразование текстовых и числовых значения в корректный формат.
3. Удалены строки с некорректными или слишком короткими названиями торговых точек.
4. Выбраны только те торговые точки, в которых ассортимент насчитывает от 150 до 1000 уникальных товаров.
5. Из отфильтрованных торговых точек случайным образом были отобраны 150.
6. Данные были преобразованы в матрицу, где:

* строки – наименование товара;
* столбцы – название торговой точки;
* ячейки – количество закупленного товара торговой точкой.

Результат обработки и анализа данных о закупках товаров торговыми точками представлен в таблице 20.

Таблица 20 – Результат обработки и анализа данных о закупках товаров торговыми точками

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *product*\_*name* | хороший | черемушки | шодруз | успех | экспресс |
| fantola молочный с начинкой вкус голубая малина и печеньем 66г | 0 | 0 | 0 | 0 | 0 |
| fantola молочный с начинкой со вкусом avocado-fest и печеньем 66г | 0 | 0 | 0 | 0 | 0 |
| fantola молочный с начинкой со вкусом bubble gum и печеньем 66г | 0 | 0 | 0 | 0 | 0 |
| fantola молочный с начинкой со вкусом choco vibe c начинкой и печеньем 66г | 0 | 0 | 0 | 0 | 0 |
| шедевр хрустящие со слив.начинкой, целым фундуком темный шок. 145г | 0 | 0 | 0 | 0 | 0 |
| #макфа хлопья 4 злака, 400гр.,(02.02.24) | 0 | 0 | 0 | 0 | 0 |
| #мармелад арбузный (29.12.2023) | 0 | 0 | 0 | 0 | 0 |
| #ясно солнышко каша овсяная с абрикосом 45 гр(11.11.23) | 0 | 0 | 0 | 0 | 0 |
| #ясно солнышко хлопья овсяные №2 500 гр(01.12.23) | 0 | 0 | 0 | 0 | 0 |
| \*аленка 250г | 4840 | 0 | 0 | 0 | 20 |
| \*ассорти сибирская белочка 200г | 0 | 0 | 0 | 0 | 0 |
| \*бабаевская белочка 200г | 12795 | 0 | 0 | 0 | 0 |
| \*барбарис 200г | 0 | 0 | 0 | 0 | 0 |
| \*батончики ореховая роща 250г | 8580 | 0 | 0 | 0 | 0 |
| \*батончики ореховая роща 500г | 0 | 0 | 0 | 0 | 0 |
| \*веселые леденцы микс 300г | 4297 | 0 | 0 | 0 | 18 |
| \*взлет ул/э 200г | 4640 | 0 | 0 | 0 | 0 |
| \*джек 344 гр | 0 | 0 | 0 | 0 | 0 |
| \*живинка арбуз 250 гр | 0 | 0 | 0 | 0 | 0 |
| \*живинка ассорти 180 гр | 0 | 0 | 0 | 0 | 0 |
| \*карамель зеленый кузнечик вкф 200г | 0 | 0 | 0 | 0 | 0 |
| \*карамель коровка 250г | 3665 | 0 | 0 | 0 | 0 |

Код обработки и анализа данных о закупках товаров торговыми точками представлен в листинге 2.

|  |
| --- |
| Листинг 2 – Код обработки и анализа данных о закупках товаров торговыми точками |
| # Функция создания матрицы количества проданных товаров по торговым точкам  def creating\_googs\_sales\_matrix(df):  # Переименовывание колонок  df.columns = ['raw\_text', 'quantity']  # Список для хранения названий торговых точек  store\_names = []  # Переменная для названия торговой точки  current\_store = None  # Перебор всех строк в датасете  for i, row in df.iterrows():  # Если в строке 'quantity' — NaN, значит это строка с названием торговой точки  if pd.isna(row['quantity']):  # Записывание в переменную название торговой точки, убираем лишние символы  current\_store = str(row['raw\_text'])  # Добавление переменной с торговой точкой в список  store\_names.append(current\_store)  # Добавление названий торговых точек в столбец датасета  df['store\_name'] = store\_names  # Фильтрация только строк с товарами  df = df[pd.notna(df['quantity'])].copy()  # Преобразование 'quantity' в тип данных float  df['quantity'] = df['quantity'].apply(lambda x: float(str(x).replace(",", ".")))  # Переименовывание колонки с названием товара  df = df.rename(columns = {'raw\_text': 'product\_name'})  # Сброс индекса после фильтрации  df = df.reset\_index(drop = True)  # Привидение текстовых переменных к нижнему регистру  df[['product\_name', 'store\_name']] = df[['product\_name', 'store\_name']].apply(lambda x: x.str.lower())  # Функция для извлечения названия торговой точки из столбца 'store\_name'  def extract\_store\_name(store\_name):  # Поиск любого содержания в кавычках  match = re.search(r'"([^"]\*)"', store\_name)  # Возвращаем найденный текст  return match.group(1) if match else None  # Применение изменений в столбце 'store\_name'  df['store\_name'] = df['store\_name'].apply(extract\_store\_name)  # Функция для фильтрации по длине слов в названии торговой точки из столбца 'store\_name'  def deleting\_rows\_by\_store(store\_name):  # Проверка на пропущенные значения  if pd.isna(store\_name):  return False  # Разбиение слов  words = str(store\_name).split()  # Условие удаления строк  return all(5 <= len(word) <= 15 for word in words)  # Применение изменений в столбце 'store\_name'  df = df[df['store\_name'].apply(deleting\_rows\_by\_store)]  # Группировка по торговым точкам и товарам, суммируя 'quantity'  df = df.groupby(['store\_name', 'product\_name'])['quantity'].sum().reset\_index()  # Подсчёт количества уникальных товаров для каждой торговой точки  number\_unique\_products = df.groupby('store\_name')['product\_name'].nunique().reset\_index()  number\_unique\_products.columns = ['store\_name', 'unique\_products']  # Фильтрация торговых точек по количеству уникальных товаров  filtering\_stores\_by\_number\_unique\_products = number\_unique\_products[(number\_unique\_products['unique\_products'] >= 150) & (number\_unique\_products['unique\_products'] <= 1000)]  # Выбор 100 случайных торговых точек  filtere\_stores = filtering\_stores\_by\_number\_unique\_products.sample(n = min(150, len(filtering\_stores\_by\_number\_unique\_products)), random\_state = 42) if not filtering\_stores\_by\_number\_unique\_products.empty else filtering\_stores\_by\_number\_unique\_products  # Фильтрация датасета по выбранным торговым точкам  df = df[df['store\_name'].isin(filtere\_stores['store\_name'])]  # Создание матрицы количества проданных товаров по торговым точкам  df = df.pivot\_table(index = 'product\_name', columns = 'store\_name', values = 'quantity', fill\_value = 0)  # Возврат матрицы количества проданных товаров по торговым точкам  return df |

После подготовки обеих таблиц они были объединены с использованием левого соединения по наименованию товара. В итоговую таблицу вошли следующие признаки:

* матрица с количеством реализованных единиц товара по каждой торговой точке;
* наименование товара;
* категория товара;
* *ABC*-класс.

Все пропущенные значения были заменены нулями, т. к. отсутствие продаж интерпретируется как отсутствие интереса к товару.

Результат объединения таблиц представлен в таблице 21.

Таблица 21 – Результат объединения таблиц

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| яблоко | яблоня | яранга | product name | product category | abc classification |
| 45 | 15 | 60 | аленка (ок) 90г | шоколад | A |
| 84 | 84 | 42 | роллтон вермишель б/п витамин. куриная на домашнем бульоне, 60гр | продукты быстрого приготовления | A |
| 0 | 6 | 0 | макфа мука пшеничная в/с, 2кг.челябинск | мука | A |
| 3 | 0 | 5 | макфа спираль макар.изд., в/с, 223-3а 400гр. | макаронные изделия | A |
| 2 | 0 | 3 | в шок. москвичка бк | карамель | A |
| 5 | 0 | 5 | халва подсолнечная тимоша | к чаю | A |
| 0 | 20 | 20 | батон бабаевский с помадно-сливочной начинкой (ок) 50г | шоколад | A |
| 0 | 0 | 0 | батончик рот-фронт в/з 250г | конфеты | A |
| 0 | 0 | 3 | нури (пак) 100\*2г (0201) высок дабл | чай | A |
| 0 | 0 | 0 | мореход бк2 | конфеты | A |
| 0 | 0 | 0 | люкс (ок) 90г | шоколад | A |
| 0 | 0 | 0 | бушидо оригинал крист.100г | кофе | A |
| 0 | 84 | 42 | роллтон вермишель б/п витамин. в говяжьем бульоне, 60гр | продукты быстрого приготовления | A |
| 10 | 0 | 0 | канди (лист) 250г (0052) медиум | чай | A |
| 0 | 0 | 3 | аленка (ок) 200г | шоколад | A |
| 15 | 15 | 0 | злато подсолнечное рафин.дезод. масло, 1 сорт, 1л | растительные масла | A |
| 0 | 0 | 2 | батончик рот-фронт в/з | конфеты | A |
| 0 | 84 | 62 | аленка (ок) 15г | шоколад | A |
| 0 | 0 | 0 | левушка с карамельной нач. | конфеты | A |
| 0 | 15 | 0 | золотая семечка подсолнечное рафин.дезор.масло 1 сорт,1л | растительные масла | A |
| 0 | 0 | 1 | кара-кум в/н | конфеты | A |
| 0 | 0 | 6 | кола 2л пэт | напитки | A |
| 0 | 0 | 1 | халва рот-фронт арахисовая глазир. в/з | к чаю | A |
| 0 | 10 | 0 | макфа мука пшеничная в/с, 1кг. челябинск | мука | A |

Код объединения таблиц представлен в листинге 3.

|  |
| --- |
| Листинг 3 – Код объединения таблиц |
| # Объединение датасетов по столбцу 'product\_name'  df\_merged = df\_matrix.merge(df\_info[['product\_name', 'product\_category', 'abc\_classification', 'total\_realization']], left\_index = True, right\_on = 'product\_name', how = 'right')\  # Заполнение пропуенных значений в матрице нулями  df\_merged = df\_merged.fillna(0) |

На основе объединённой таблицы была создана матрица взаимодействия, где строки соответствуют товарам, столбцы – торговым точкам, а значения – количеству проданных единиц. Эта матрица использовалась в качестве входных данных для модели *LightFM*, которая применяет гибридный подход, учитывая как историю взаимодействий, так и контентные признаки (категория товара, *ABC*-класс) при генерации рекомендаций.

3.2 Архитектура модели

Рекомендательная система, разработанная для дипломной работы, использует гибридный подход, объединяющий коллаборативную и контентную фильтрации. Для реализации модели выбран алгоритм *LightFM* – библиотека на языке *Python*, ориентированная на построение рекомендательных систем, которые учитывают как взаимодействие пользователей и объектов, так и признаки самих объектов.

Архитектура модели включает следующие ключевые компоненты:

* пользователи – торговые точки, которым необходимо рекомендовать товары;
* объекты – товары, классифицированные по категориям и *ABC*-классу;
* контентные признаки объектов – объединённая строка из категории товара и его *ABC*-классификации, которая используется как атрибут для формирования фич;
* матрица взаимодействий – разреженная матрица, в которой отражено количество реализованных единиц каждого товара в каждой торговой точке;
* модель *LightFM* – обучается на матрице взаимодействий и контентных признаках, что позволяет учитывать как прошлое поведение торговой точки, так и характеристики товаров.

Для построения модели использовалась объектно-ориентированная структура:

* все компоненты, включая датасеты, признаки и модель, сохраняются в сериализованном виде, что обеспечивает возможность повторного использования и интеграции в *Telegram*-бота;
* функция генерации рекомендаций принимает параметры торговой точки, категории товара и количества рекомендаций и возвращает ранжированный список наименований товаров на основе предсказанных оценок интереса.

3.3 Обучение модели и оценка качества

Обучение рекомендательной модели происходило на основе заранее подготовленной разреженной матрицы взаимодействий, где строки представляют собой товары, столбцы – торговые точки, а значения соответствуют количеству реализованных единиц продукции. Это позволяет зафиксировать поведение торговых точек в терминах покупок и использовать его в дальнейшем для генерации персонализированных рекомендаций.

В качестве инструмента была использована библиотека *LightFM*, специально предназначенная для реализации рекомендательных систем, сочетающих коллаборативную и контентную фильтрации. Обучение проводилось с использованием следующих параметров модели:

* *loss*=*'warp'* – алгоритм обучения с функцией потерь *Weighted* *Approximate*-*Rank* *Pairwise*. Он предназначен для оптимизации качества ранжирования товаров в списке рекомендаций. Основная идея *WARP* заключается в том, чтобы максимизировать вероятность правильной сортировки товаров, т. е. чтобы релевантные товары находились выше в списке;
* *no*\_*components*=30 – размерность латентного пространства факторов. Эта величина определяет, сколько скрытых признаков (*features*) будет использоваться для представления как товаров, так и торговых точек. Чем выше значение, тем больше модель может уловить сложных паттернов, но и возрастает риск переобучения;
* *epochs*=20 – количество эпох обучения. Одна эпоха – это полный проход по обучающему набору данных. Данное значение выбрано как сбалансированное, позволяющее достичь сходимости модели без излишней перегрузки;
* *learning*\_*rate*=0.05 – скорость обучения. Этот параметр регулирует, насколько сильно обновляются параметры модели на каждом шаге градиентного спуска. Слишком низкое значение замедлит обучение, а слишком высокое может привести к нестабильности модели. Значение 0.05 позволяет модели достаточно быстро обучаться, сохраняя при этом устойчивость.

Код регистрации пользователей и объектов и обучение модели представлены в листинге 4

|  |
| --- |
| Листинг 4 – Код регистрации пользователей и объектов и обучение модели |
| # Создание объекта Dataset для хранения данных  df\_lightfm = Dataset()  # Регистрация торговых точек и товаров  df\_lightfm.fit(users = df\_matrix.columns, items = df\_merged['product\_name'])  # Формирования списка уникальных признаков товара (категория + ABC-классификация)  goods\_feature\_list = (df\_merged['product\_category'] + ' ' + df\_merged['abc\_classification']).unique().tolist()  # Регистрация признаков товаров  df\_lightfm.fit\_partial(items = df\_merged['product\_name'], item\_features = goods\_feature\_list)  # Построение матрицы взаимодействий между торговыми точками и товарами  interactions, \_ = df\_lightfm.build\_interactions(      # Создания кортежа (торговая точка, товар, объём закупки)      ((store, goods, quantity)      # Перебор каждый товар      for goods in df\_matrix.index      # Для каждого товара перебор закупок по точкам      for store, quantity in df\_matrix.loc[goods].items()      # Учитывание только положительных закупок и товаров, которые зарегистрированы       if quantity > 0 and goods in df\_lightfm.mapping()[2])  )  # Формирование разреженной матрицы признаков товаров  item\_features = df\_lightfm.build\_item\_features(((row['product\_name'], [str(row['product\_category']) + ' ' + str(row['abc\_classification'])]) for \_, row in df\_merged.iterrows()))  # Инициализация модели с функцией потерь Warp  lightfm\_model = LightFM(item\_alpha = 1e-05, learning\_rate = 0.01, loss = 'bpr', no\_components = 16, user\_alpha = 0.0)  # Обучение модели на всей выборке  lightfm\_model.fit(interactions, item\_features = item\_features, epochs = 10, num\_threads = 2) |

После завершения обучения была проведена оценка модели на всём доступном датасете с использованием метрики *precision*@*k*. Эта метрика измеряет долю релевантных товаров в первых *k* рекомендациях и позволяет оценить, насколько точно система попадает в интересы торговой точки.

Для оценки была выбрана наиболее практическая и часто используемая метрика – *precision*@5, означающая точность в топ-5 рекомендациях. Результаты показали, что *precision*@5 – 0.627, что означает: примерно 63% товаров, попавших в первую пятёрку рекомендованных, действительно представляют интерес для соответствующей торговой точки на основе её предыдущих покупок.

3.4 Руководство программиста

Данное руководство предназначено для описания настройки и использования рекомендательной системы поставок продовольственной продукции для логистических компаний, разрабатываемой в выпускной квалификационной работе.

В руководстве описаны назначение и условия применения системы, характеристики системы, описаны процедуры вызова системы, описана организация используемой входной и выходной информации.

Функциональный возможности системы:

* возможность выбора категории товара;
* возможность выбора торговой точки;
* возможность выбора количества рекомендаций;
* возможность генерации рекомендаций на основе введённых параметров;
* возможность сброса введённых параметров и из заполнения заново.

Система реализована с использованием языка программирования *Python* 3.11.11 на операционной системе *Windows* 11.

Файловая структура системы:

* *FPLS*\_*personalized*\_*recommendations*.*ipynb* – *Juputer* *Notebook* отвечающий за считывание, обработку, сохранений данных, а также обучение модели;
* *characteristics*\_*and*\_*sales*\_*of*\_*goods*.*xlsx* – необработанные данные содержащие в себе характеристики товаров;
* *sales*\_*of*\_*goods*\_*by*\_*store*.*xlsx* – необработанные данные содержащие в себе продажи товаров по торговым точкам;
* *characteristics*\_*and*\_*sales*\_*of*\_*goods*\_*prepared*.*xlsx* – обработанные данные содержащие в себе характеристики товаров;
* *matrix*\_*number*\_*goods*\_*sold*\_*in*\_*stores*.*xlsx* – обработанные данные содержащие в себе продажи товаров по торговым точкам;
* *FPLS*\_*lightfm*\_*model*.*pickle* – сериализованная модель машинного обучения;
* *telegram*\_*bot*.*py* – реализация логики *Telegram*-бота.

Для начала использования системы нужно проделать следующие шаги:

* создайте виртуальное окружение с версией *Python* 3.11.11;
* установить все необходимые библиотеки, которые прописаны в исполняющих файлах;
* запустить систему из исполняющего файла *telegram*\_*bot*.*py*.

Системные требования к составу и параметрам технических средств:

* процессор *Intel* *Core* i3/ *AMD* *Ryzen* 3 или лучше;
* оперативная память не меньше 8 Гб;
* не менее 20 Гб свободного места на жестком диске;
* операционная система *Windows* 10/11.

3.5 Руководство пользователя

Разработанная рекомендательная система представляет собой программное обеспечение, предназначенное для оптимизации процесса предложения товаров торговыми представителями. Система реализована в форме *Telegram*-бота, что обеспечивает удобный и интуитивно понятный интерфейс взаимодействия. Код Telegram-бота представлен в приложении «А» Ниже приводится детальное описание этапов работы с системой.

На рисунке 12 изображена инициация взаимодействия с системой осуществляется посредством запуска *Telegram*-бота. Данная процедура предполагает открытие приложения *Telegram*, нахождение бота по имени или с использованием предоставленной гиперссылки и последующую активацию посредством нажатия кнопки «Старт». В ответ на данное действие система генерирует приветственное сообщение, сигнализирующее о готовности к функционированию и содержащее дальнейшие инструкции по эксплуатации.

На рисунке 13 пользователю предоставляется возможность выбора категории товарной продукции, для которой необходимо сгенерировать рекомендации. Система отображает перечень доступных категорий, из которого пользователь осуществляет выбор категории. Данная функциональность обеспечивает фильтрацию данных и формирование более релевантных рекомендаций, соответствующих потребностям конкретной торговой точки.

|  |  |
| --- | --- |
| Рисунок 12 – Стартовое окно бота | Рисунок 13 – Выбор категории товара |

После определения категории товара, система запрашивает у пользователя информацию о конкретной торговой точке, для которой требуется формирование рекомендаций. На рисунке 14 пользователю предоставляется список доступных торговых точек, из которого он выбирает целевую точку. Выбор торговой точки является критически важным этапом, поскольку система использует исторические данные о закупках и предпочтениях данной точки для генерации персонализированных рекомендаций.

На рисунке 15 система предоставляет пользователю возможность определения объема выходных данных, а именно количества генерируемых рекомендаций. Пользователю предлагаются фиксированные варианты на выбор, 3, 6, 9 или 12 рекомендаций. Данная функция позволяет пользователю контролировать информационную нагрузку.

|  |  |
| --- | --- |
| Рисунок 14 – Выбор торговой точки | Рисунок 15 – Выбор количества рекомендаций |

После ввода пользователем параметров система выводит эти параметры списком для проверки. Также перед пользователем встаёт выбор, изображённый на рисунке 16 подтвердить и сгенерировать рекомендации или же заполнить параметры заново.

На основе введенных пользователем параметров система осуществляет генерацию персонализированных рекомендаций с применением алгоритмов машинного обучения. Результатом данного процесса является отображение списка товаров, которые с высокой вероятностью будут востребованы выбранной торговой точкой. Выводимая изображённая на рисунке 17 информация включает наименование товара, категорию, *ABC*-классификацию

|  |  |
| --- | --- |
| Рисунок 16 – Вывод введённых параметров | Рисунок 17 – Генерация рекомендаций |

Система предусматривает функциональность повторной инициации процесса заполнения параметров. На рисунке 18 пользователь имеет возможность активировать функцию «Заполнить заново», что приводит к обнулению ранее введенных данных и возврату к этапу выбора категории товара. Данная функция обеспечивает гибкость взаимодействия и позволяет оперативно корректировать запросы.

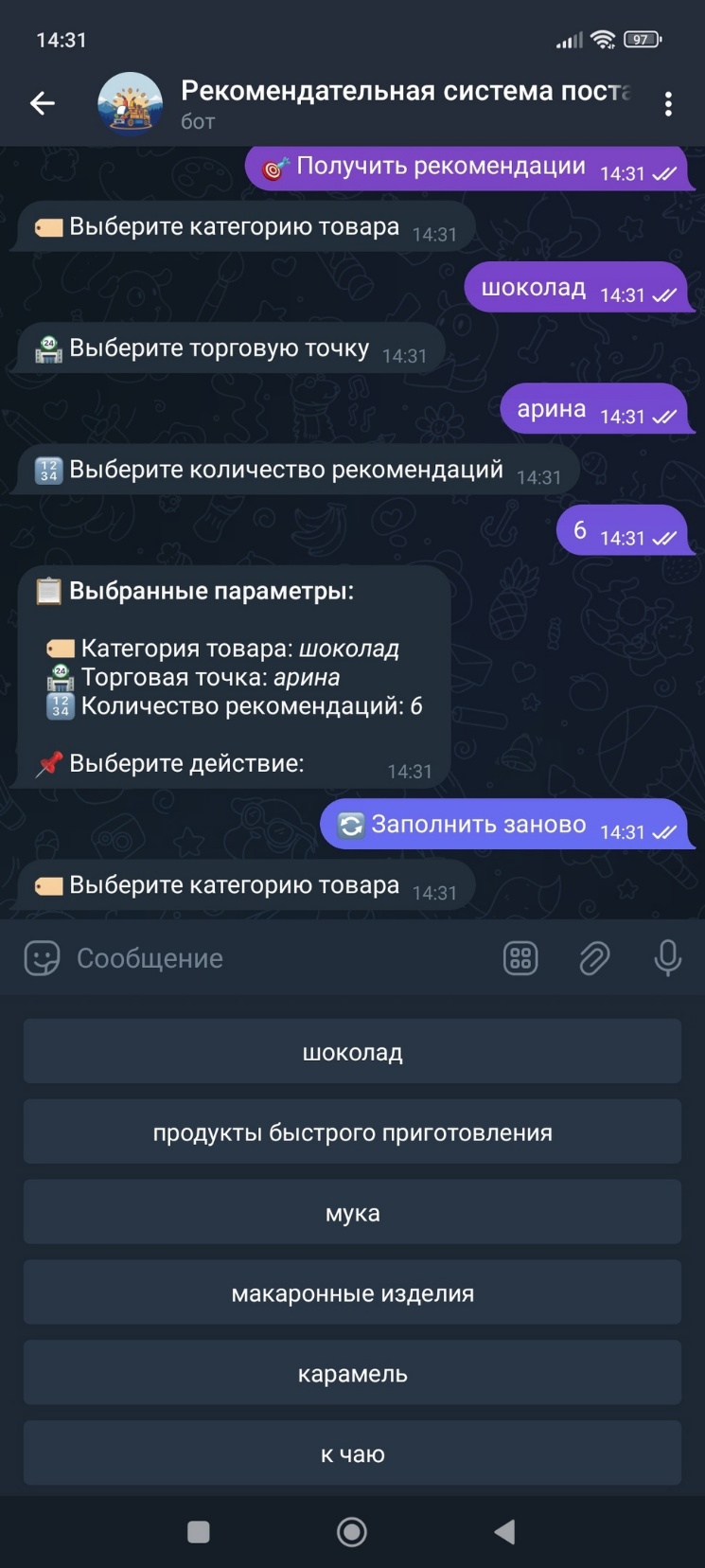


Рисунок 19 – Заполнение параметров заново

3.6 Тестирование системы

Тестирование рекомендательной системы проводилось с целью проверки корректности работы разработанного программного обеспечения, а также оценки качества и релевантности формируемых рекомендаций. В ходе тестирования особое внимание уделялось функциональной стабильности всех компонентов системы, корректной интеграции модели машинного обучения с пользовательским интерфейсом на базе *Telegram*-бота, а также соответствию рекомендаций заявленным требованиям.

Функциональная проверка охватывала полный цикл взаимодействия пользователя с системой: от ввода исходных параметров до получения сформированных предложений. Была протестирована работа всех основных функций *Telegram*-бота: выбор категории товара, выбор торговой точки, задание количества рекомендаций, генерация результатов и возможность повторного ввода параметров. *Telegram*-бот устойчиво функционировал при множественных сеансах взаимодействия, корректно обрабатывал пользовательский ввод и обеспечивал передачу данных в модель *LightFM*. При генерации рекомендаций бот отображал список товаров с указанием наименования, категории и соответствующего *ABC*-класса. Оформление вывода реализовано с использованием *HTML*-разметки, что повышает удобство восприятия информации.

Особое внимание в процессе тестирования было уделено корректности семантической логики рекомендаций. Для этого была проведена ручная валидация предлагаемых товаров, основанная на экспертной оценке соответствия рекомендованного ассортимента ранее зафиксированным предпочтениям конкретных торговых точек. Были выбраны различные сценарии тестирования: для точек с высоким оборотом, для точек с ограниченным ассортиментом, а также для новых или неактивных точек. Анализ результатов показал, что система стабильно предлагает наиболее релевантные товары, преимущественно относящиеся к классам *A* и *B* по *ABC*-анализу.

В целом рекомендации характеризовались высокой степенью соответствия как выбранной категории товаров, так и особенностям торговой точки. Товары с низким уровнем реализации и нехарактерные позиции в сформированных списках не присутствовали, что подтверждает эффективность применённой фильтрации и предобработки данных.

3.7 Выводы по главе

В третьей главе была описана разработка рекомендательной системы в соответствии с ранее сформулированными требованиями и архитектурными решениями. Проведён детальный анализ исходных данных: выполнена предобработка характеристик товаров, произведена их агрегация по категориям, а также реализована *ABC*-классификация на основе накопительной суммы. Для истории закупок торговых точек была построена разреженная матрица взаимодействий, отражающая фактическое поведение клиентов.

На основе объединённых данных была обучена рекомендательная модель с использованием библиотеки *LightFM*. Подробно описана архитектура модели: учтены признаки товаров, реализован гибридный подход, сочетающий коллаборативную и контентную фильтрации. Обучение модели происходило с использованием функции потерь *WARP* и дало точность *precision*@5 – 0.627, что указывает на высокую релевантность генерируемых рекомендаций.

Проведено всестороннее тестирование системы, включая функциональную проверку корректности взаимодействия всех компонентов, а также семантическую валидацию рекомендаций. Установлено, что *Telegram*-бот стабильно работает, корректно принимает параметры от пользователя и отображает рекомендации в понятной форме. Проверка результатов вручную показала, что предложенные товары соответствуют бизнес-логике и интересам торговых точек.

Дополнительно были разработаны руководства пользователя и программиста, содержащие инструкции по запуску, настройке и эксплуатации системы. Эти документы повышают удобство внедрения и сопровождения программного продукта на предприятии.

Заключение

В результате выполнения выпускной квалификационной работы была разработана интеллектуальная рекомендательная система для автоматизации процесса формирования товарных предложений торговым точкам логистической компании. Разработанная система позволяет с высокой точностью подбирать товары, основываясь на истории закупок конкретной торговой точки, характеристиках продукции и применении гибридного метода фильтрации.

Для достижения поставленной цели были решены следующие задачи:

* проведён анализ предметной области, дана характеристика логистической компании, её организационной структуры, бизнес-процессов и текущего уровня автоматизации;
* рассмотрены современные подходы к построению рекомендательных систем, включая коллаборативную, контентную и гибридную фильтрации; обоснован выбор гибридного метода как наиболее эффективного для поставленной задачи;
* сформулированы функциональные и нефункциональные требования к системе, разработаны диаграммы прецедентов и аналитические модели, описывающие внутреннюю структуру и поведение системы;
* произведена предобработка данных: сгруппированы и очищены характеристики товаров, реализована *ABC*-классификация, построена матрица взаимодействий на основе истории продаж;
* обучена рекомендательная модель с использованием библиотеки *LightFM* и подобранными параметрами обучения. Достигнута точность *precision*@5 на уровне 62.7%, что свидетельствует о высоком качестве выдаваемых рекомендаций;
* реализован *Telegram*-бот, обеспечивающий интуитивно понятный интерфейс взаимодействия с системой; пользователь может выбрать категорию, торговую точку и количество рекомендаций, после чего получает адаптированные товарные предложения;
* проведено комплексное тестирование программного продукта, подтвердившее его корректную работу, стабильность и соответствие требованиям; вручную проверена релевантность рекомендаций, которые соответствовали поведенческой модели закупок и бизнес-логике;
* разработаны и представлены руководства пользователя и программиста, обеспечивающие внедрение и поддержку системы в условиях эксплуатации.

Таким образом, в ходе выполнения работы была создана законченная программная система, способная существенно повысить эффективность процесса подбора товаров, снизить нагрузку на торговых представителей и улучшить коммерческие результаты логистической компании. Полученное решение готово к дальнейшему внедрению и масштабированию в рамках ИТ-инфраструктуры компании.

Список использованных источников

1. Рекомендательные системы для логистических компаний / [Электронный ресурс] // jomeam : [сайт]. — URL: https://jomeam.ru/ru/nauka/article/91740/view (дата обращения: 04.05.2025).
2. What is Amazon Personalize? / [Электронный ресурс] // amazon : [сайт]. — URL: https://docs.aws.amazon.com/personalize/latest/dg/what-is-personalize.html (дата обращения: 04.05.2025).
3. Recommendations AI modeling / [Электронный ресурс] // google : [сайт]. — URL: https://cloud.google.com/blog/topics/developers-practitioners/recommendations-ai-modeling (дата обращения: 04.05.2025).
4. What is Personalizer? / [Электронный ресурс] // microsoft : [сайт]. — URL: https://learn.microsoft.com/en-us/azure/ai-services/personalizer/what-is-personalizer (дата обращения: 04.05.2025).
5. Головчинер, М. Н. Введение в системы знаний. Курс лекций – Томск. 2011. – 168 с. – Текст: электронный / М. Н. Головчинер // Файловый архив для студентов. *StudFiles* [сайт] – URL: http://tic.tsu.ru/apache22/data/news/files/Системы%20знаний.pdf (дата обращения: 13.05.2025).
6. Коробова, И. Л. Методы представления знаний – Тамбов: Издательство ТГТУ, 2003. – 521 с. – Текст: электронный / И. Л. Коробова // Z-Library – крупнейшая в мире электронная библиотека. Свободный доступ к знаниям и культуре [сайт] – URL: https://reader.z-lib.fm/read/41221d00cf11caebebe0f86c2cdd82c6ff53a7e2ed77d4cb856fdb40734c51bc/801691/9eac1c/Методы-представления-знаний.html?client\_key=1fFLi67gBrNRP1j1iPy1&extension=pdf&signature=6a2400f9c2aec39029eaf65651cce8f3dc33d6047412647985dc15fe48f39393 (дата обращения: 13.05.2025).
7. Вигерс Карл. Разработка требований к программному обеспечению. 3-е изд., дополненное / Пер. с англ. — М. : Издательство «Русская редакция» ; СПб. : БХВ-Петербург, 2014. — 736 с. Текст: электронный / Вигерс Карл, Битти Джой // Кафедра ИМО [сайт] – URL: https://cs.petrsu.ru/~ybgv/Progproject/ucheb/Vigers-treb.pdf (дата обращения: 13.05.2025).
8. Жданова, С. А. Информационные системы [Текст] / С. А. Жданова, М. Л. Соболевой — 2015 — 304 c. (Дата обращения: 14.05.2025).
9. Григорьев М., Григорьева И. Проектирование информационных систем. Учебное пособие для вузов. – Litres, 2021. – 297 с. Текст: непосредственный (Дата обращения: 14.05.2025).
10. Никонова Е. З. Информационное моделирование в проектировании информационных систем // Наука и мир. – 2017. – Т. 1. – №. 2. – С. 39-40. Текст: непосредственный (Дата обращения: 14.05.2025).
11. Python: Введение и история создания / [Электронный ресурс] // sky.pro : [сайт]. — URL: https://sky.pro/wiki/python/python-vvedenie-i-istoriya-sozdaniya (Дата обращения: 15.05.2025).
12. Язык программирования R: возможности, особенности и преимущества / [Электронный ресурс] // sky.pro : [сайт]. — URL: https://sky.pro/wiki/analytics/yazyk-programmirovaniya-r-vozmozhnosti-osobennosti-i-preimushchestva/ (Дата обращения: 15.05.2025).
13. Java: что это такое, где используется этот язык программирования, свойства, плюсы и минусы, как начать изучать с нуля / [Электронный ресурс] // sky.pro : [сайт]. — URL: https://sky.pro/media/kak-nachat-izuchat-java/ (Дата обращения: 15.05.2025).
14. Рекомендательные системы: основы и примеры / [Электронный ресурс] // sky.pro : [сайт]. — URL: https://sky.pro/wiki/python/rekomendatelnye-sistemy-osnovy-i-primery/ (Дата обращения: 15.05.2025).
15. Как установить и использовать TensorFlow / [Электронный ресурс] // sky.pro : [сайт]. — URL: https://sky.pro/wiki/python/kak-ustanovit-i-ispolzovat-tensorflow/ (Дата обращения: 15.05.2025).
16. Python для обработки больших данных / [Электронный ресурс] // sky.pro : [сайт]. — URL: https://sky.pro/wiki/python/python-dlya-obrabotki-bolshih-dannyh/ (Дата обращения: 15.05.2025).
17. Создание ботов на Python с использованием Aiogram / [Электронный ресурс] // sky.pro : [сайт]. — URL: https://sky.pro/wiki/python/sozdanie-botov-na-python-s-ispolzovaniem-aiogram/ (Дата обращения: 15.05.2025).
18. Что такое Pickle и как его использовать в Python – Skypro / [Электронный ресурс] // sky.pro : [сайт]. — URL: https://sky.pro/media/chto-takoe-pickle-i-kak-ego-ispolzovat-v-python/ (Дата обращения: 15.05.2025).
19. Joblib: максимум из параллельных вычислений в Python / [Электронный ресурс] // Ha : [сайт]. — URL: https://habr.com/ru/companies/otus/articles/874810/ (Дата обращения: 15.05.2025).
20. Баланов, А. Н. Машинное обучение и искусственный интеллект: учебное пособие для вузов – Санкт-Петербург: Лань, – 2024. – 172 с. – Текст: электронный / А. Н. Баланов // Лань: электронно-библиотечная система. – URL: https://e.lanbook.com/book/414920 (дата обращения: 15.05.2025).

Приложение А

import os

import pickle

import telebot

import numpy as np

import pandas as pd

from lightfm import LightFM

from dotenv import load\_dotenv

from telebot.types import ReplyKeyboardMarkup, KeyboardButton

# Загрузка модели рекомендательной системы

with open('model/FPLS\_lightfm\_model.pickle', 'rb') as f:

model\_data = pickle.load(f)

# Извлечение компонентов модели рекомендательной системы

lightfm\_model = model\_data['model']

df\_lightfm = model\_data['dataset']

item\_features = model\_data['item\_features']

df\_merged = model\_data['df\_merged']

df\_matrix = model\_data['df\_matrix']

# Функция для получения рекомендаций

def recommendations\_goods\_for\_stores(store\_name, product\_category, number\_recommendations):

# Проверка на наличие торговой точки в датасете

if store\_name not in df\_matrix.columns:

return (f'Торговая точка - {store\_name} не найдена')

# Извлечение всех товаров из выбранной категории

candidate\_products = df\_merged[(df\_merged['product\_category'] == product\_category)]

# Проверка на наличие товаров по выбранной категории и торговой точке

if candidate\_products.empty:

return (f'Нет данных о закупках в категории - {product\_category} для торговой точки - {store\_name}')

# Лист со списком наименований товаров, которые будут рекомендоваться

product\_names = candidate\_products['product\_name'].values

try:

# Определение идентификатора магазина для модели

store\_id = df\_lightfm.mapping()[0][store\_name]

# Преобразование наименований товаров в идентификаторы

product\_id = [df\_lightfm.mapping()[2][p] for p in product\_names if p in df\_lightfm.mapping()[2]]

# Генерация рекомендаций

generating\_recommendations = lightfm\_model.predict(store\_id, product\_id, item\_features = item\_features)

# Сортировка товаров по убыванию предсказанных оценок и выбор топ-n товаров

ratings\_recommendations = np.argsort(-generating\_recommendations)[:number\_recommendations]

top\_product\_names = [product\_names[i] for i in ratings\_recommendations]

# Возвращение сгенерированных рекомендаций товаров для торговой точки

return df\_merged[df\_merged['product\_name'].isin(top\_product\_names)][['product\_name', 'product\_category', 'abc\_classification', 'total\_realization']]

except KeyError as e:

return (f'Ошибка: {str(e)}')

# Список доступных торговых точек

store\_names = list(df\_matrix.columns)

# Список доступных категорий товаров

product\_categories = df\_merged['product\_category'].unique()

# Считывание токена из файла для инициализации бота

try:

with open('telegram\_bot\_key.txt', 'r') as file:

tg\_token = file.read().strip()

if not tg\_token:

raise ValueError('Файл пустой')

tg\_bot = telebot.TeleBot(tg\_token)

except FileNotFoundError:

print('Файл не найдей')

except ValueError as ve:

print(f'Ошибка: {ve}')

except Exception as e:

print(f'Неизвестная ошибка: {e}')

# Переменная для выбранной категории товара

selected\_product\_category = None

# Переменная для выбранной торговой точки

selected\_store\_name = None

# Переменная для выбранного количетсва рекомендаций

selected\_number\_recommendations = 0

# Функция вывода главного меню бота

def show\_main\_menu(message):

markup = ReplyKeyboardMarkup(resize\_keyboard = True)

markup.add(KeyboardButton('🎯 Получить рекомендации'))

tg\_bot.send\_message(message.chat.id, '🚀 Нажмите на кнопку, чтобы начать заполнение данных для получения рекомендаций', reply\_markup = markup)

# Функция для начала работы бота

@tg\_bot.message\_handler(commands = ['start'])

def start(message):

markup = ReplyKeyboardMarkup(resize\_keyboard = True)

# Добавление кнопки 'Получить рекомендации'

markup.add(KeyboardButton('🎯 Получить рекомендации'))

# Вывод текста перед кнопкой

tg\_bot.send\_message(message.chat.id,'🚀 Нажмите на кнопку, чтобы начать заполнение данных для получения рекомендаций', reply\_markup = markup)

# Функция для выбора категории товара

@tg\_bot.message\_handler(func = lambda message: message.text == '🎯 Получить рекомендации')

def product\_category\_selection(message):

# Создание клавиатуры с кнопками, где одна кнопка - одна категория товара

markup = ReplyKeyboardMarkup(resize\_keyboard = True, one\_time\_keyboard = True)

# Добавление кнопки для каждой категории товара

for product\_category in product\_categories:

markup.add(KeyboardButton(product\_category))

# Вывод текста перед кнопками

tg\_bot.send\_message(message.chat.id, '🏷 Выберите категорию товара', reply\_markup = markup)

# Функция для сохранения выбранной категории товара

@tg\_bot.message\_handler(func = lambda message: message.text in product\_categories)

def saved\_product\_category(message):

# Использование глобальной переменной для хранения выбранной категории товара

global selected\_product\_category

# Получение выбранной категории товара

selected\_product\_category = message.text

# Вызов функции для выбора торговой точки

store\_name\_selection(message)

# Функция для выбора торговой точки

def store\_name\_selection(message):

# Создание клавиатуры с кнопками, где одна кнопка - одна торговая точка

markup = ReplyKeyboardMarkup(resize\_keyboard = True, one\_time\_keyboard = True)

# Добавление кнопки для каждой торговой точки

for store\_name in store\_names:

markup.add(KeyboardButton(store\_name))

# Вывод текста перед кнопками

tg\_bot.send\_message(message.chat.id, '🏪 Выберите торговую точку', reply\_markup = markup)

# Функция для сохранения выбранной торговой точки

@tg\_bot.message\_handler(func=lambda message: message.text in store\_names)

def saved\_store\_name(message):

# Использование глобальной переменной для хранения выбранной торговой точки

global selected\_store\_name

# Получение выбранной торговой точки

selected\_store\_name = message.text

# Вызов функции для выбора количетсва рекомендаций

number\_recommendations\_selection(message)

# Функция для выбора количества рекомендаций

def number\_recommendations\_selection(message):

# Создание клавиатуры с кнопками, где одна кнопка - колиество рекомендаций

markup = ReplyKeyboardMarkup(resize\_keyboard = True, one\_time\_keyboard = True)

# Создание первого ряда с количеством рекомендаций

markup.row(KeyboardButton('3'), KeyboardButton('6'))

# Создание второго ряда с количеством рекомендаций

markup.row(KeyboardButton('9'), KeyboardButton('12'))

# Вывод текста перед кнопками

tg\_bot.send\_message(message.chat.id, '🔢 Выберите количество рекомендаций', reply\_markup = markup)

# Функция для сохранения выбранного количетсва рекомендаций

@tg\_bot.message\_handler(func = lambda message: message.text in ['3', '6', '9', '12'])

def saved\_number\_recommendations(message):

# Использование глобальной переменной для хранения выбранного количетсва рекомендаций

global selected\_number\_recommendations, selected\_product\_category, selected\_store\_name

# Получение выбранного количетсва рекомендаций

selected\_number\_recommendations = int(message.text)

# Создание клавиатуры с кнопками, где одна кнопка - вызов функции генерации рекомендаций, вторая - начать заполнение сначала

markup = ReplyKeyboardMarkup(resize\_keyboard = True, row\_width = 2)

# Создание кнопок, где одна кнопка - вызов функции генерации рекомендаций, вторая - начать заполнение сначала

markup.add(KeyboardButton('✅ Сгенерировать рекомендации'), KeyboardButton('🔄 Заполнить заново'))

# Сообщение с информацией о введённых параметрах

info = (

"📋 <b>Выбранные параметры:</b>\n\n"

f" 🏷 Категория товара: <i>{selected\_product\_category}</i>\n"

f" 🏪 Торговая точка: <i>{selected\_store\_name}</i>\n"

f" 🔢 Количество рекомендаций: <i>{selected\_number\_recommendations}</i>\n\n"

"📌 Выберите действие:"

)

# Отправка информации о введённых параметрах

tg\_bot.send\_message(message.chat.id, info, reply\_markup = markup, parse\_mode = 'HTML')

# Функция перезапуска выбора параметров для генерации рекомендаций

@tg\_bot.message\_handler(func = lambda message: message.text == '🔄 Заполнить заново')

def restart\_process(message):

product\_category\_selection(message)

# Функция для отправки рекомендаций

@tg\_bot.message\_handler(func = lambda message: message.text == '✅ Сгенерировать рекомендации')

def output\_recommendations(message):

try:

# Получение рекомендаций из модели

recommendations = recommendations\_goods\_for\_stores(store\_name = selected\_store\_name, product\_category = selected\_product\_category, number\_recommendations = selected\_number\_recommendations)

# Если на выходе сообщение об ошибке, то выводим её

if isinstance(recommendations, str):

tg\_bot.send\_message(message.chat.id, recommendations)

# Иначе, если датасет, то выводим его

else:

# Вызов функции отвечающей за оформления вывода рекомендаций

result = recommendation\_output\_format(recommendations)

# Отправка оформленных рекомендаций

tg\_bot.send\_message(message.chat.id, result, parse\_mode = 'HTML')

except Exception as e:

# Вывод сообщения об исключении

error\_message = (f'Ошибка при генерации рекомендаций:\n{str(e)}')

# Вывод сообщения об исключении

tg\_bot.send\_message(message.chat.id, error\_message)

finally:

show\_main\_menu(message)

#Функция оформления вывода рекомендаций

def recommendation\_output\_format(recommendations):

count = 1

# Заголовок сообщения с выводом рекомендаций

message = "<b>🎯 Рекомендации по выбранным параметрам:</b>\n\n"

# Вывод всех рекмоендованных товаров

for i, row in recommendations.iterrows():

message += (

f"<b>№ {count}</b> 🍎 Наименование товара - <i>{row['product\_name']}</i>\n"

f" 🏷 Категория товара - <i>{row['product\_category']}</i>\n"

f" ⚖️ ABC-классификация - <i>{row['abc\_classification']}</i>\n\n"

)

count += 1

return message

# Функция для остановки бота

@tg\_bot.message\_handler(commands = ['stop'])

def stop\_bot(message):

tg\_bot.send\_message(message.chat.id, 'Бот остановлен')

tg\_bot.stop\_polling()

exit(0)

# Запуск бота

tg\_bot.polling(none\_stop = True)

Бакалаврская работа выполнена мною самостоятельно. Используемые в работе материалы и концепции из опубликованной научной литературы и других источников имеют ссылки на них. Библиографический список содержит 20 наименований.

Один печатный экземпляр и один электронный экземпляр сданы на кафедру ИЭС.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
| (дата) |  | (подпись) |  | (ФИО) |