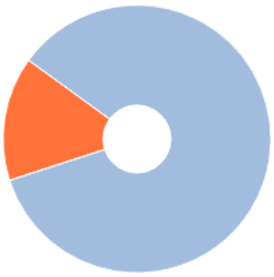
АНТИПЛАГИАТ
ОБНАРУЖЕНИЕ ЗАИМСТВОВАНИЙ

Отчет о проверке

РЕЗУЛЬТАТЫ ПРОВЕРКИ



Совпадения:
14,67%

Оригинальность:
85,33%

Цитирования:
0%

Самоцитирования:
0%

«Совпадения», «Цитирования», «Самоцитирования», «Оригинальность» являются отдельными показателями, отображаются в процентах и в сумме дают 100%, что соответствует проверенному тексту документа.

Есть подозрения на следующие группы маскировки заимствований: Сгенерированный текст на страницах: 11, 30, 35... еще на 2 стр.

Проверено: 94,59% текста документа, исключено из проверки: 5,41% текста документа. Разделы, отключенные пользователем: Библиография

- **Совпадения** — фрагменты проверяемого текста, полностью или частично сходные с найденными источниками, за исключением фрагментов, которые система отнесла к цитированию или самоцитированию. Показатель «Совпадения» – это доля фрагментов проверяемого текста, отнесенных к совпадениям, в общем объеме текста.
- **Самоцитирования** — фрагменты проверяемого текста, совпадающие или почти совпадающие с фрагментом текста источника, автором или соавтором которого является автор проверяемого документа. Показатель «Самоцитирования» – это доля фрагментов текста, отнесенных к самоцитированию, в общем объеме текста.
- **Цитирования** — фрагменты проверяемого текста, которые не являются авторскими, но которые система отнесла к корректно оформленным. К цитированиям относятся также шаблонные фразы; библиография; фрагменты текста, найденные модулем поиска «СПС Гарант: нормативно-правовая документация». Показатель «Цитирования» – это доля фрагментов проверяемого текста, отнесенных к цитированию, в общем объеме текста.
- **Текстовое пересечение** — фрагмент текста проверяемого документа, совпадающий или почти совпадающий с фрагментом текста источника.
- **Источник** — документ, проиндексированный в системе и содержащийся в модуле поиска, по которому проводится проверка.
- **Оригинальный текст** — фрагменты проверяемого текста, не обнаруженные ни в одном источнике и не отмеченные ни одним из модулей поиска. Показатель «Оригинальность» – это доля фрагментов проверяемого текста, отнесенных к оригинальному тексту, в общем объеме текста.

Обращаем Ваше внимание, что система находит текстовые совпадения проверяемого документа с проиндексированными в системе источниками. При этом система является вспомогательным инструментом, определение корректности и правомерности совпадений или цитирований, а также авторства текстовых фрагментов проверяемого документа остается в компетенции проверяющего.

ПАРАМЕТРЫ ПРОВЕРКИ

Выполнена проверка с учетом редактирования: Да

Исключение элементов документа из проверки: Нет

Выполнено распознавание текста (OCR): Нет

Выполнена проверка с учетом структуры: Да

Модули поиска: IEEE, Сводная коллекция ЭБС, Переводные заимствования по коллекции Интернет в английском сегменте, Перефразированные заимствования по коллекции Интернет в английском сегменте, Патенты СССР, РФ, СНГ, Кольцо вузов*, ИПС Адилет, Медицина, Коллекция НБУ, СМИ России и СНГ, Публикации РГБ (переводы и перефразирования), Переводные заимствования IEEE, Перефразированные заимствования по коллекции Интернет в русском сегменте, СПС ГАРАНТ: аналитика, Публикации РГБ, Шаблонные фразы, Издательство Springer Nature, Кольцо вузов (переводы и перефразирования), Переводные заимствования по коллекции Гарант: аналитика, Переводные заимствования по коллекции Интернет в русском сегменте, Цитирование, Диссертации НББ, Публикации eLIBRARY, Перефразирования по коллекции IEEE, Публикации eLIBRARY (переводы и перефразирования), Перефразирования по СПС ГАРАНТ: аналитика, СПС ГАРАНТ: нормативно-правовая документация, Интернет Плюс, Собственная коллекция компании

ИСТОЧНИКИ

№	Доля в тексте	Доля в отчете	Источник	Актуален на	Модуль поиска	Комментарий
[01]	2,75%	0,46%	ПРОЕКТИРОВАНИЕ ИС ВЕДЕНИЯ ...	29 Янв 2025	Кольцо вузов*	
[02]	2,44%	0,51%	ПРОЕКТИРОВАНИЕ ИС ВЕДЕНИЯ ...	29 Янв 2025	Кольцо вузов (переводы и перефразирования)	
[03]	2,25%	0,19%	6447-10095-1-SM.docx	15 Июл 2022	Кольцо вузов*	
[04]	2,25%	0%	6447-10095-1-SM.docx	15 Июл 2022	Кольцо вузов*	
[05]	2,24%	0,87%	150304_20I0826_Sedov_I_O_Tekhn...	06 Июн 2024	Кольцо вузов*	
[06]	2,14%	0,89%	https://istina.msu.ru/download/51... https://istina.msu.ru	06 Фев 2025	Перефразированные заимствования по коллекции Интернет в русском сегменте	
[07]	2,02%	1,46%	6447-10095-1-SM.docx	15 Июл 2022	Кольцо вузов (переводы и перефразирования)	
[08]	1,85%	0,54%	150304_20I0826_Sedov_I_O_Tekhn...	06 Июн 2024	Кольцо вузов (переводы и перефразирования)	
[09]	1,84%	0,69%	Разработка информационной си...	20 Июн 2024	Кольцо вузов (переводы и перефразирования)	
[10]	1,76%	0%	6447-10095-1-SM.docx	15 Июл 2022	Кольцо вузов (переводы и перефразирования)	
[11]	1,66%	1,66%	3101-4930-1-SM.pdf	05 Авг 2021	Кольцо вузов (переводы и перефразирования)	
[12]	1,45%	0,18%	Разработка информационной си...	20 Июн 2024	Кольцо вузов*	
[13]	1,31%	1,31%	Аналитика больших данных для ...	20 Мар 2025	Перефразирования по СПС ГАРАНТ: аналитика	
[14]	1,3%	0%	https://nauchkor.ru/uploads/docu... https://nauchkor.ru	10 Апр 2024	Интернет Плюс	
[15]	1,27%	1,03%	Креативно-когнитивные модули ... https://book.ru	01 Янв 2022	Сводная коллекция ЭБС	
[16]	1,27%	0%	Креативно-когнитивные модули ... https://book.ru	01 Янв 2024	Сводная коллекция ЭБС	
[17]	1,17%	1,17%	ПРОГРАММНЫЙ ПРОДУКТ "СТО: ... http://elibrary.ru	01 Янв 2017	Публикации eLIBRARY (переводы и перефразирования)	
[18]	1,11%	0,07%	ВКР Ремизова 2.06	02 Июн 2022	Кольцо вузов*	
[19]	1,05%	0%	Образец экономической части В... https://topuch.ru	06 Июл 2022	Интернет Плюс	

[20]	1,05%	0%	Образец экономической части В... https://topuch.ru	26 Окт 2021	Интернет Плюс	
[21]	1,03%	1,03%	Джобава попытка 1	08 Июн 2024	Кольцо вузов*	
[22]	0,95%	0%	не указано	26 Дек 2024	Шаблонные фразы	Источник исключен. Причина: Маленький процент пересечения.
[23]	0,69%	0,69%	Яндекс Картинки https://yandex.ru	11 Ноя 2024	Перефразированные заимствования по коллекции Интернет в английском сегменте	
[24]	0,69%	0,69%	m_th_e.v.starkov_2024.pdf https://elar.urfu.ru	17 Мая 2025	Перефразированные заимствования по коллекции Интернет в русском сегменте	
[25]	0,64%	0,33%	ПЗ_Асланян_ДГ_БПИ238.pdf	10 Апр 2025	Кольцо вузов (переводы и перефразирования)	
[26]	0,58%	0%	Новости для бухгалтера, экономи...	25 Мар 2025	СПС ГАРАНТ: аналитика	
[27]	0,58%	0,22%	VKR_HEBO-01-19_Gurakova.V.S.pdf https://istina.fnkcrr.ru	17 Сен 2024	Перефразированные заимствования по коллекции Интернет в русском сегменте	
[28]	0,55%	0%	Хаутиев, Адам Магомет-Баширов... http://dlib.rsl.ru	01 Янв 2015	Публикации РГБ	
[29]	0,53%	0%	Актуальные вопросы медико-био... http://elibrary.ru	01 Янв 2020	Публикации eLIBRARY	Источник исключен. Причина: Маленький процент пересечения.
[30]	0,49%	0%	Развитие системы оказания пом... http://elibrary.ru	01 Янв 2020	Публикации eLIBRARY	Источник исключен. Причина: Маленький процент пересечения.
[31]	0,43%	0%	2015_Аитов АИ_080801.65_Газетд...	25 Авг 2015	Кольцо вузов*	
[32]	0,43%	0%	Обзоры изменений по охране тр... http://ivo.garant.ru	05 Фев 2022	СПС ГАРАНТ: аналитика	Источник исключен. Причина: Маленький процент пересечения.
[33]	0,42%	0%	Школьная медицина https://book.ru	01 Янв 2024	Сводная коллекция ЭБС	Источник исключен. Причина: Маленький процент пересечения.
[34]	0,4%	0%	Исследование характеристик нап... http://fan-5.ru	07 Дек 2017	Интернет Плюс	Источник исключен. Причина: Маленький процент пересечения.
[35]	0,37%	0%	Ч. 2 http://dlib.rsl.ru	01 Янв 2014	Публикации РГБ	Источник исключен. Причина: Маленький процент пересечения.
[36]	0,37%	0%	Экономика предприятия	19 Дек 2016	Медицина	Источник исключен. Причина: Маленький процент пересечения.
[37]	0,36%	0%	Обзоры изменений по охране тр... http://ivo.garant.ru	09 Ноя 2019	СПС ГАРАНТ: аналитика	Источник исключен. Причина: Маленький процент пересечения.
[38]	0,34%	0%	Общепрофессиональные аспект...	20 Янв 2020	Медицина	Источник исключен. Причина: Маленький процент пересечения.
[39]	0,34%	0%	Общепрофессиональные аспект...	20 Дек 2016	Медицина	Источник исключен. Причина: Маленький процент пересечения.
[40]	0,34%	0%	Общепрофессиональные аспект...	20 Янв 2020	Медицина	Источник исключен. Причина: Маленький процент пересечения.
[41]	0,34%	0%	ПЕРВАЯ ПОМОЩЬ ПРИ УГРОЖА... http://elibrary.ru	01 Янв 2020	Публикации eLIBRARY	Источник исключен. Причина: Маленький процент пересечения.
[42]	0,34%	0,34%	Лучшие Сайты сведений об орга... https://soware.ru	25 Мар 2025	Переводные заимствования по коллекции Интернет в русском сегменте	
[43]	0,32%	0,32%	Формирование аппаратного и пр...	07 Ноя 2024	Переводные заимствования по коллекции Гарант: аналитика	
[44]	0,31%	0%	Планирование на предприятии https://e.lanbook.com	22 Янв 2020	Сводная коллекция ЭБС	Источник исключен. Причина: Маленький процент пересечения.
[45]	0,31%	0%	2019ВКР530335КОМАРОВ	21 Июн 2019	Кольцо вузов (переводы и перефразирования)	
[46]	0,3%	0%	Проектирование веб-приложени...	19 Мая 2025	Кольцо вузов (переводы и перефразирования)	Источник исключен. Причина: Маленький процент пересечения.
[47]	0,27%	0%	Шкатулла В.И., Краснов Ю.К., Суе... http://ivo.garant.ru	22 Фев 2020	СПС ГАРАНТ: аналитика	Источник исключен. Причина: Маленький процент пересечения.
[48]	0,27%	0%	Димова, Алла Львовна Теоретич... http://dlib.rsl.ru	01 Янв 2022	Публикации РГБ	Источник исключен. Причина: Маленький процент пересечения.
[49]	0,27%	0%	Бойко, Ирина Геннадьевна Сове... http://dlib.rsl.ru	01 Янв 2016	Публикации РГБ	Источник исключен. Причина: Маленький процент пересечения.

[50]	0,26%	0%	U3BghtgjAEeD.pdf https://elib.pnzgu.ru	23 Мая 2025	Переводные заимствования по коллекции Интернет в русском сегменте	Источник исключен. Причина: Маленький процент пересечения.
[51]	0,26%	0%	Экономика автотранспортной от... http://elibrary.ru	01 Янв 2022	Публикации eLIBRARY	Источник исключен. Причина: Маленький процент пересечения.
[52]	0,25%	0%	Прогнозирование спроса для сер... https://na-journal.ru	08 Апр 2025	Переводные заимствования по коллекции Интернет в русском сегменте	Источник исключен. Причина: Маленький процент пересечения.
[53]	0,23%	0%	Бюджетное право	20 Янв 2020	Сводная коллекция ЭБС	Источник исключен. Причина: Маленький процент пересечения.
[54]	0,23%	0%	Органы охраны правопорядка: у...	19 Дек 2016	Медицина	Источник исключен. Причина: Маленький процент пересечения.
[55]	0,23%	0%	О ВНЕСЕНИИ ИЗМЕНЕНИЙ В ЧАС... http://lipeck.bezformata.com	22 Ноя 2018	СМИ России и СНГ	Источник исключен. Причина: Маленький процент пересечения.
[56]	0,23%	0%	Федеральный закон от 27 ноября... https://rg.ru	30 Ноя 2017	СМИ России и СНГ	Источник исключен. Причина: Маленький процент пересечения.
[57]	0,23%	0%	Практическая налоговая энцикло... http://ivo.garant.ru	01 Дек 2013	СПС ГАРАНТ: аналитика	Источник исключен. Причина: Маленький процент пересечения.
[58]	0,23%	0%	под ред. А. В. Касьянова Постате... http://dlib.rsl.ru	01 Янв 2013	Публикации РГБ	Источник исключен. Причина: Маленький процент пересечения.
[59]	0,23%	0%	http://www.bek-ufa.ru/upload/Prog... http://bek-ufa.ru	29 Дек 2021	Интернет Плюс	Источник исключен. Причина: Маленький процент пересечения.
[60]	0,21%	0%	Распоряжение Комитета эконом... http://ivo.garant.ru	20 Авг 2005	СПС ГАРАНТ: нормативно-правовая документация	Источник исключен. Причина: Маленький процент пересечения.
[61]	0,2%	0%	Бухгалтерский учет. Интенсивны...	27 Ноя 2017	Сводная коллекция ЭБС	Источник исключен. Причина: Маленький процент пересечения.
[62]	0,2%	0%	Бухгалтерский учет. Интенсивны...	19 Дек 2016	Медицина	Источник исключен. Причина: Маленький процент пересечения.
[63]	0,19%	0%	Методологические основы разра... https://book.ru	01 Янв 2023	Сводная коллекция ЭБС	Источник исключен. Причина: Маленький процент пересечения.
[64]	0,19%	0%	Постников, Никита Александрови... http://dlib.rsl.ru	20 Сен 2024	Публикации РГБ	Источник исключен. Причина: Маленький процент пересечения.
[65]	0,17%	0%	О ратификации Договора о Евраз... http://adilet.zan.kz	04 Окт 2017	ИПС Адилет	Источник исключен. Причина: Маленький процент пересечения.
[66]	0,17%	0%	О внесении на рассмотрение Пр... http://adilet.zan.kz	04 Окт 2017	ИПС Адилет	Источник исключен. Причина: Маленький процент пересечения.
[67]	0,17%	0%	О ратификации Договора о Евраз... http://adilet.zan.kz	21 Янв 2016	ИПС Адилет	Источник исключен. Причина: Маленький процент пересечения.
[68]	0,16%	0%	Способ формирования сигнала в... http://findpatent.ru	25 Июн 2015	Патенты СССР, РФ, СНГ	Источник исключен. Причина: Маленький процент пересечения.
[69]	0,16%	0%	Способ контроля электромагнит... http://findpatent.ru	24 Июн 2015	Патенты СССР, РФ, СНГ	Источник исключен. Причина: Маленький процент пересечения.
[70]	0,16%	0%	Шкатулла В.И., Краснов Ю.К., Суе...	27 Мар 2025	СПС ГАРАНТ: аналитика	Источник исключен. Причина: Маленький процент пересечения.
[71]	0,16%	0%	Методические рекомендации пр... http://elibrary.ru	01 Янв 2023	Публикации eLIBRARY	Источник исключен. Причина: Маленький процент пересечения.
[72]	0,15%	0%	Проверка списания организация... http://ivo.garant.ru	23 Авг 2003	СПС ГАРАНТ: аналитика	Источник исключен. Причина: Маленький процент пересечения.
[73]	0,15%	0%	Основные положения по планир... http://ivo.garant.ru	раньше 2011	СПС ГАРАНТ: нормативно-правовая документация	Источник исключен. Причина: Маленький процент пересечения.
[74]	0,15%	0%	Приказ Минздравмедпрома ССС... http://ivo.garant.ru	14 Апр 2001	СПС ГАРАНТ: нормативно-правовая документация	Источник исключен. Причина: Маленький процент пересечения.
[75]	0,11%	0%	Доклинические исследования ле...	26 Янв 2018	Медицина	Источник исключен. Причина: Маленький процент пересечения.

1

2

3

УДК 004.4

Руководитель ВКР: к.т.н., доцент Т.Б. Аждер

Консультант по экономическому разделу: к.э.н., доцент И.В. Чижанькова

Мухаметшин А.Р., 3 Выпускная квалификационная работа по образовательной программе «Разработка программных продуктов и проектирование информационных систем» направления подготовки «Программная инженерия» на тему «Интеллектуальная система оценки спроса на продукт»: М. 2025 г., МИРЭА – Российский технологический университет (РТУ МИРЭА), Институт информационных технологий (ИТ), кафедра инструментального и прикладного программного обеспечения (ИиППО) – 65 стр. 7 , 23 илл., 10 табл., 8 листинг., 4 формул, 30 ист. лит (в т.ч. 14 на английском яз.).

Ключевые слова 15 : прогнозирование спроса, машинное обучение, микросервисная архитектура, Python, Golang, анализ данных, REST API, Docker, база данных.

Целью работы является разработка интеллектуальной системы для автоматизированной оценки спроса на продукт, основанной на анализе данных из разнородных источников с применением алгоритмов машинного обучения.

Mukhametshin A.R., Final qualification work on the educational program "Software Product Development and Information Systems design" in the field of Software Engineering on the topic 15 “Intelligent system of product demand estimation”: M. 2025, 25 MIREA - Russian Technological University (RTU MIREA), Institute of Information Technologies (IT), Department of Instrumental and Applied Software (IiPPO) - 65 p., 23 illustrations, 10 tables, 8 listings, 4 formulas, 30 references (including 14 in English 15).

Keywords: demand forecasting, machine learning, microservice architecture, Python, Golang, data analysis, REST API, Docker, database.

The goal of the work is to develop an intelligent system for automated product demand estimation based on analyzing data from heterogeneous sources using machine learning algorithms.

РТУ МИРЭА: 119454, Москва, пр-т Вернадского, д. 78

Кафедра инструментального и прикладного программного обеспечения (ИиППО)

Тираж: 1 экз. (на правах рукописи)

Файл 15 : «ЛЗ_090304_21И1589_ИКБО-20-21_МухаметшиАР.pdf», исполнитель Мухаметшин А.Р.

4

СОДЕРЖАНИЕ

6

ТЕРМИНЫ И ОПРЕДЕЛЕНИЯ

В настоящем отчёте применяются следующие термины и определения.

Анализ данных – процесс обработки и интерпретации данных для выявления закономерностей и поддержки принятия решений.

Контейнеризация – Технология, позволяющая упаковывать приложения и их зависимости в изолированные контейнеры для упрощения развертывания и масштабирования (например, Docker).

Машинное обучение – раздел искусственного интеллекта, использующий алгоритмы для обучения моделей на основе данных.

Микросервисная архитектура – подход к разработке, при котором приложение разбивается на независимые, автономные сервисы, каждый из которых отвечает за конкретную функцию системы.

Прогнозирование спроса – метод оценки будущих потребностей рынка на основе анализа исторических данных и текущих трендов.

Масштабируемость – возможность увеличивать ресурсы или добавлять дополнительные экземпляры сервисов для поддержания производительности системы при росте нагрузки.

REST API – архитектурный стиль для создания программных интерфейсов, использующий HTTP-запросы для

обмена данными между клиентом и сервером.

Веб-скрапинг – технология автоматизированного извлечения данных с веб-сайтов для последующей обработки и анализа.

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ

В настоящем отчёте применяются следующие сокращения и обозначения.

API – Application Programming Interface (Программный интерфейс приложения)

CSS – Cascading Style Sheets (каскадные таблицы стилей)

CSV – Comma-Separated Values (формат файла для хранения табличных данных)

JWT – JSON Web Token

ML – Machine Learning (Машинное обучение)

SPA – Single Page Application

ИС – Интеллектуальная система

ИТ – Информационные технологии

ПО – Программное обеспечение

ВВЕДЕНИЕ

В условиях стремительного развития цифровых технологий и роста конкуренции компании сталкиваются с необходимостью оперативного реагирования на изменения спроса. Разработка интеллектуальных систем для автоматизированного анализа данных и прогнозирования рыночных потребностей становится ключевым инструментом повышения эффективности бизнеса.

Целью выпускной квалификационной работы является разработка интеллектуальной системы для автоматизированной оценки спроса на продукт, использующей машинное обучение для прогнозирования на основе данных из разнородных источников. Система направлена на повышение конкурентоспособности компаний за счет точных прогнозов и оптимизации бизнес-процессов.

Актуальность разработки обусловлена изменениями рыночных условий, где традиционные методы оценки спроса на основе экспертных суждений или ручного анализа данных теряют эффективность. Исследования показывают, что внедрение интеллектуальных систем анализа данных сокращает издержки на 15–20% за счет минимизации избыточных запасов и упущенных возможностей [1]. Использование технологий машинного обучения повышает точность прогнозирования спроса по сравнению с традиционными подходами [2,23].

Новизна работы заключается в создании универсальной системы 11, интегрирующей данные из внешних API, веб-скрапинга и подготовленных наборов данных, их обработку с помощью ML-моделей и предоставление результатов через веб-интерфейс и API. Система отличается гибкостью настройки источников данных и адаптивностью к различным бизнес-сценариям.

В задачи текущей работы входит анализ предметной области и решений, проектирование архитектуры системы, выбор инструментов и методов разработки, разработка и тестирование системы, расчет стоимости работ.

Объектом исследования является интеллектуальная система оценки спроса на продукт, включающая модули сбора, предобработки и анализа данных с использованием технологий машинного обучения для прогнозирования рыночных потребностей.

Предметом исследования являются процессы автоматизированного сбора, предобработки и анализа данных из разнородных источников с использованием технологий машинного обучения для прогнозирования спроса на продукт в интеллектуальной системе.

На защиту выносится интеллектуальная система, обеспечивающая автоматизированный сбор данных из различных источников, их анали 13 з с помощью ML-моделей и предоставление прогнозов спроса через API и веб-интерфейс.

Стандарты, используемые в работе: СМКО МИРЭА 7.5.1/03.П.30-19 [3], ГОСТ 7.32-2017 [4], ГОСТ Р 7.0.100-2018 [5], СМКО МИРЭА 7.5.1/03.П.67-19 [6], ФГОС ВО 3++ по направлению подготовки 09.03.04 Программная инженерия 15 [7].

В процессе написания выпускной квалификационной работы автор (дипломник) руководствовался следующими нормативными актами:

1. «О защите населения и территории от чрезвычайных ситуаций природного и техногенного характера» от 21.12.1994 No 68-ФЗ.
2. «Об основах охраны здоровья граждан в Российской Федерации» от 21.11.2011 No 323-ФЗ.
3. «О гражданской обороне» от 12.02.1998 No 28-ФЗ.
4. Приказ Минздравсоцразвития РФ от 04.05.2012 No 477н «Об утверждении перечня состояний, при которых оказывается первая помощь, и перечня мероприятий по оказанию первой помощи».
5. Трудовой кодекс Российской Федерации от 30.12.2001 No 197-ФЗ.
6. СанПиН 2.2.2/2.4.1340-03 «Гигиенические требования к персональным электронно-вычислительным машинам и организации работы». 21

1 ИССЛЕДОВАТЕЛЬСКИЙ РАЗДЕЛ 21

1.1 Анализ существующих решений

В условиях цифровизации бизнеса и роста объемов данных системы автоматической оценки спроса становятся важным инструментом для компаний, стремящихся оптимизировать управление запасами и повысить эффективность маркетинговых стратегий [22]. Для определения требований к разрабатываемой интеллектуальной системе проведен сравнительный анализ существующих решений.

1.1.1 Ozon Seller

Ozon Seller [8] – это инструмент аналитики, разработанный для продавцов на платформе Ozon, одной из крупнейших российских торговых онлайн-площадок. Платформа предоставляет пользователям доступ к статистике продаж, прогнозам спроса и аналитике конкурентной среды. Веб-интерфейс приложения позволяет визуализировать ключевые показатели, такие как объемы продаж по категориям товаров и динамика спроса в определенные периоды. Пример веб-интерфейса представлен на рисунке 1.1.

Рисунок 1.1 – Веб-интерфейс платформы Ozon Seller

1.1.2 Moneyplace

Moneyplace [9] – это российская аналитическая платформа, предназначенная для продавцов маркетплейсов, включая Ozon, Wildberries и Яндекс.Маркет. Сервис предоставляет инструменты для анализа спроса, конкурентной среды и управления ассортиментом. Веб-интерфейс платформы ориентирован на визуализацию данных о продажах и прогнозах 13, что помогает пользователям принимать обоснованные решения о закупках и ценообразовании. Пример интерфейса показан на рисунке 1.2.

Рисунок 1.2 – Веб-интерфейс платформы Moneyplace

1.1.3 MPStats

MPStats [10] – это аналитическая платформа, предназначенная для продавцов на маркетплейсах Wildberries, Ozon и Яндекс.Маркет. Сервис предоставляет инструменты для анализа продаж, конкурентов, оптимизации рекламных кампаний и исследования товаров. Веб-интерфейс платформы позволяет пользователям получать доступ к детальным отчетам о продажах, выручке, ценах, рейтингах и других ключевых показателях. Кроме того, MPStats предлагает расширение для браузера Chrome, которое упрощает доступ к аналитическим данным непосредственно на страницах маркетплейсов. Пример интерфейса показан на рисунке 1.3.

Рисунок 1.3 – Веб-интерфейс платформы MPStats

1.1.4 Результаты сравнительного анализа

По итогу сравнительного анализа существующих аналогов разрабатываемой ИС составлена таблица 1.1.

Таблица 1.1 – Сравнительный анализ аналогов ИС

Критерий	Ozon Seller	Moneyplace	MPStats
Функциональность	Отчеты о продажах, анализ конкурентов, рекомендации по закупкам	Анализ спроса, сравнение цен, прогноз остатков	Аналитика продаж, анализ конкурентов, оптимизация рекламы, исследование товаров
Интеграция	REST API для доступа к данным	API для получения аналитических данных	API для получения аналитических данных
Гибкость	Ограничена платформой Ozon		
Поддержка	нескольких		

маркетплейсов
Поддержка
Wildberries, Ozon,
Яндекс.Маркет
Аналитика и
статистика
Статистика продаж и
конкурентов
Детализированная
аналитика по товарам
Подробные отчеты
по продажам,
выручке, ценам,
рейтингам
Дополнительные
возможности
Простота интерфейса Анализ данных с
разных платформ
Расширение для
браузеров
13

Проведенный анализ показывает, что существующие решения ориентированы преимущественно на анализ данных внутри конкретных маркетплейсов и не обладают достаточной гибкостьюдля интеграции внешних источников или применения продвинутых методов прогнозирования. Разрабатываемая система должна преодолеть эти ограничения и соответствовать следующим требованиям:

– гибкость настройки: Возможность подключения различных источников данных (готовые наборы данных, API, веб-скрапинг) и адаптации под нужды конкретного бизнеса 42 ;

– интеграция: Предоставление REST API для легкой интеграции с существующими системами компаний;

– аналитика: Использование ML-моделей для глубокого анализа и точного прогнозирования спроса [16], а также визуализация результатов через веб-интерфейс.

Таким образом, разрабатываемая система должна сочетать универсальность, высокую точность прогнозов и удобство использования, что обеспечит ее конкурентоспособность на рынке аналитических решений.

1.2 Определение требований к решению

1.2.1 Формулирование требований к программной части системы

Функциональные требования

Система должна автоматизировать сбор данных об использовании продуктов и прогнозировать их спрос. Целевой аудиторией являются владельцы бизнесов.

Необходимо обеспечить:

– сбор данных из подготовленных наборов данных, внешних API и веб-скрапинг;

– обработка и нормализация данных для ML;

– прогнозирование спроса с использованием ML-моделей;

– предоставление API для интеграции с внешними системами;

– визуализация прогнозов и аналитики через веб-интерфейс 11 .

14

Нефункциональные требования

Основными метриками, которые необходимо обеспечить является:

– время отклика API – не более 1 секунды при нагрузке до 100 пользователей;

– время генерации прогноза – не более 5 минут;

– безопасность: авторизация через JWT [14], шифрование данных.

Система должна запускаться в Docker’е для обеспечения кроссплатформенности.

Требования пользователей

Для удобства использования система должна иметь:

– удобный интерфейс для просмотра прогнозов;

– понятную и простую структуру страниц веб-интерфейса.

1.2.2 Формулирование требований к аппаратной части системы

Требования к серверной части системы

Для серверной части выдвигаются следующие минимальные требования к аппаратному обеспечению:

– процессор: 64-битный, с минимальной частотой 2.0 ГГц 43 ;

– оперативная память: 8 ГБ для обеспечения одновременной работы нескольких микросервисов и брокера сообщений;

– пропускная способность сетевого соединения: 100 Мбит/с для быстрой передач 25 и данных между сервисами и внешними источниками;

– хранилище: 100 ГБ SSD.

Требования к оперативной памяти обусловлены необходимостью параллельной обработки запросов от множества пользователей и асинхронного взаимодействия между микросервисами через брокер сообщений. Высокая пропускная способность сети необходима для оперативного получения данных из внешних API и передачи их в систему для последующей обработки.

15

1.3 Выбор инструментов и методов создания системы

Для серверной части системы выбраны следующие технологии:

– Go [19] (Gin[11]): Высокопроизводительный язык программирования с фреймворком Gin для создания микросервисов. Go обеспечивает быструю обработку запросов, низкое потребление ресурсов и простоту работы с конкурентными задачами благодаря встроенным горутинам;

– RabbitMQ: Брокер сообщений для асинхронной коммуникации между микросервисами. RabbitMQ позволяет декомпозировать процессы сбора, обработки и анализа данных, обеспечивая устойчивость системы при высоких нагрузках.

Преимущества данного подхода включают высокую скорость выполнения запросов, легкость масштабирования и надежность передачи данных, что соответствует требованиям микросервисной архитектуры [28].

Клиентская часть системы реализована с использованием React [13] – библиотекой JavaScript для построения динамических и адаптивных пользовательских интерфейсов с применением компонентного подхода, что позволяет создать удобный, быстрый и адаптивный веб-интерфейс, обеспечивающий интерактивное взаимодействие с системой и визуализацию результатов анализа.

Для управления данными выбрана система PostgreSQL: Основная реляционная база данных для хранения структурированных данных (пользователи, обработанные данные, прогнозы). PostgreSQL обеспечивает высокую производительность, надежность и поддержку сложных запросов, что делает её подходящей для аналитических задач. Так же некоторые данные, такие как обработанные данные или готовые модели будут храниться в виде файлов в локальном хранилище на сервере.

Модуль машинного обучения реализован с использованием:

– Python (scikit-learn, LightGBM): Python выбран как основной язык для разработки ML-моделей благодаря богатому набору библиотек. Scikit-learn используется для обработки данных, а LightGBM – для построения

высокоэффективных градиентных бустинговых моделей [15], которые отлично подходят для прогнозирования временных рядов спроса и цен;

– Go с интеграцией Python: для эффективного использования моделей применяется подход с вызовом Python-скриптов из Go-сервиса, что обеспечивает гибкость при обучении и использовании моделей при сохранении высокой производительности API.

– Pickle: Формат для сохранения обученных моделей. Модели сохраняются в формате pickle, что позволяет легко передавать их между средами выполнения Python.

Такой подход обеспечивает гибкость в обучении моделей и их эффективное использование в реальном времени.

Для удобного развертывания и масштабирования системы используется Docker[24] и Docker Compose. Эти инструменты позволяют запускать весь стек приложений в изолированных контейнерах, не требуя сложных настроек окружения, что значительно упрощает развертывание и масштабирование. Каждый микросервис работает в своём контейнере, что обеспечивает изоляцию и упрощает управление зависимостями.

Для разработки системы выбраны следующие инструменты:

– GoLand: Интегрированная среда разработки для Go, предоставляющая удобный интерфейс, автодополнение кода и встроенные плагины для работы с микросервисами и gRPC. Это ускоряет процесс написания серверной части;

– PgAdmin: Графический интерфейс для администрирования PostgreSQL[25], упрощающий управление базой данных и выполнение запросов.

Эти инструменты выбраны за их популярность, функциональность и совместимость с технологическим стеком проекта. GoLand поддерживает интеграцию с Docker, что упрощает тестирование и отладку микросервисов в контейнерах. PgAdmin обеспечивает удобный доступ к базе данных, позволяя визуализировать структуру таблиц и оптимизировать SQL-запросы для повышения производительности системы.

1.4 Постановка задачи к проектированию и разработке системы

Целью выпускной квалификационной работы является создание интеллектуальной системы, позволяющей пользователям анализировать рынок и эффективно прогнозировать спрос на товары.

Для реализации поставленной цели выделены следующие задачи к проектированию и разработке:

– спроектировать интеллектуальную систему (архитектуру и

взаимодействие элементов системы);
– определить и обосновать информационные, технические, программные средства для разработки системы;
– разработать серверную и клиентскую части системы, предоставляющей пользователям заявленный функционал;
– произвести тестирование разработанной системы;
– Рассчитать стоимость проведенных работ.

1.5 Вывод к разделу 1

В данном разделе проведен детальный сравнительный анализ существующих решений для оценки спроса, что позволило выявить ключевые требования и конкурентные преимущества разрабатываемой системы. На основе анализа определены функциональные и нефункциональные требования, а также выбраны оптимальные инструменты и методы разработки. Были поставлены задачи к проектированию и разработке системы.

2 ПРОЕКТНЫЙ РАЗДЕЛ

2.1 Проектирование функциональной схемы

Для разработки системы необходимо было определить и формализовать ключевые бизнес-процессы, обеспечивающие её функционирование. С этой целью была спроектирована функциональная схема в методологии IDEF0 [12], которая позволяет структурировать процессы создания системы, отражая их взаимосвязи, входные данные, управляющие факторы и выходные результаты.

Контекстная диаграмма, представленная на рисунке А.1, демонстрирует взаимодействие системы с внешней средой: источниками данных, пользователями и инфраструктурой. Входными данными выступают необработанные данные о спросе и требования пользователей, управляющими факторами – стандарты разработки и конфигурации источников, а выходными результатами – прогнозы спроса и аналитические отчёты.

На декомпозиции контекстной диаграммы, изображённой на рисунке А.2, раскрываются основные функции системы: сбор данных, их обработку, обучение моделей машинного обучения, генерацию прогнозов и предоставление результатов через веб-интерфейс и API.

На рисунках А.3-А.4 показаны декомпозиции основных блоков системы «Обработка данных» и «Прогнозирование спроса». На них более подробно раскрывается функционал и работа системы.

2.2 Проектирование архитектуры системы

Для разработки ИС была выбрана микросервисная архитектура [18]. Это обусловлено необходимостью разделения функциональных компонентов системы – сбора данных, их обработки, прогнозирования спроса, аутентификации и предоставления API – на независимые модули, что обеспечивает гибкость, масштабируемость, отказоустойчивость и эффективность работы.

Так же было решено использовать паттерн API Gateway, реализованный в виде отдельного сервиса, который принимает входящие запросы от клиентского сервиса, выполняет их аутентификацию и маршрутизацию к

соответствующим микросервисам. Этот подход упрощает взаимодействие между клиентской частью и серверными компонентами, а также повышает безопасность системы за счёт централизованной обработки запросов.

Коммуникация между микросервисами осуществляется как через

прямые HTTP-запросы, так и асинхронно через брокер сообщений RabbitMQ. Асинхронное взаимодействие позволяет эффективно обрабатывать поток данных между сервисами сбора и обработки данных, обеспечивая устойчивость системы под нагрузкой.

Для хранения данных используется PostgreSQL как основная реляционная база данных, обеспечивающая надежность и поддержку сложных запросов для всех компонентов системы – от хранения учетных записей пользователей до сохранения обработанны 24 х данных и результатов прогнозирования. Каждый микросервис имеет доступ к необходимым таблицам в общей базе данных, при этом соблюдается принцип изоляции данных для обеспечения независимости компонентов системы.

2.3 Проектирование клиентской части системы

Клиентский уровень реализован через веб-приложение на React с TypeScript, предоставляющее удобный интерфейс для управления данными, просмотра прогнозов и анализа спроса на товары. Взаимодействие с сервером осуществляется через REST API, предоставляемый API Gateway.

Архитектура клиентской части построена на принципах компонентного подхода, что обеспечивает переиспользование кода и упрощает поддержку системы. Это особенно важно для отображения различных аналитических данных и интерактивной работы с прогнозами спроса и цен. Использование TypeScript повышает надежность кода за счет строгой типизации, что снижает вероятность ошибок при разработке. Кроме того, модульная структура React-компонентов позволяет легко масштабировать интерфейс, добавляя новые функции, такие как интерактивные графики и фильтры для анализа данных.

Схема связи страниц клиентской стороны ИС представлена на рисунке 2.1.

20

Рисунок 2.1 – Схема связи страниц клиентской части ИС

2.4 Проектирование серверной части системы

Серверный уровень реализован как набор взаимодействующих микросервисов, каждый из которых решает специфическую задачу в рамках системы прогнозирования спроса. Взаимодействие между сервисами происходит через асинхронный обмен сообщениями (с использованием RabbitMQ) и прямые HTTP-запросы. Такой подход обеспечивает устойчивость системы: если один из сервисов временно недоступен, остальные продолжают функционировать, обрабатывая данные из очереди сообщений.

Основные микросервисы системы включают:

– Data Collector Service: отвечает за сбор и первичную обработку данных о товарах маркетплейса. Сервис периодически извлекает информацию из подключенных источников данных, обрабатывает её и отправляет в очередь RabbitMQ для дальнейшей обработки.

– Data Processor Service: получает данные из RabbitMQ, проводит их очистку, нормализацию и агрегацию. Устраняет дубликаты, проводит валидацию, вычисляет дополнительные признаки (например, скользящие средние, лаги) и сохраняет подготовленные данные в PostgreSQL и в виде CSV

21

файла для последующего анализа и прогнозирования.

– ML Service: реализует логику прогнозирования спроса и цен товаров. Использует подготовленные данные для обучения моделей на базе LightGBM, сохраняет обученные модели в формате pickle и предоставляет API для генерации прогнозов. Интегрирует Python-скрипты машинного обучения с Go-

сервером, что обеспечивает как гибкость при разработке моделей, так и производительность при обработке запросов. Так же данный сервис проводит анализ последних собранных данных для выявления наиболее перспективных товаров.

– Auth Service: управляет аутентификацией и авторизацией пользователей. Хранит учетные данные в PostgreSQL, генерирует и проверяет JWT-токены.

– API Gateway: выступает единой точкой входа для клиентского приложения, проверяет JWT-токены через Auth Service и маршрутизирует запросы к соответствующим микросервисам. Обеспечивает логирование запросов, обработку ошибок и предоставляет унифицированный интерфейс для взаимодействия с системой.

Такая архитектура обеспечивает чёткое разделение ответственности между компонентами, упрощает тестирование и поддержку системы, а также позволяет независимо масштабировать отдельные сервисы в зависимости от нагрузки.

2.5 Разработка диаграмм логической модели системы

Для лучшего понимания процессов необходимых для функционирования системы и её структуры были созданы несколько диаграмм.

Диаграмма последовательности, показана на рисунке 2.2. Она демонстрирует процесс обработки запроса на прогноз спроса – от отправки клиентом запроса через API Gateway до получения результата.

Диаграмма компонентов представлена на рисунке 2.3. Она иллюстрирует общую архитектуру, показывая связи между клиентским приложением, API

Gateway, микросервисами и базой данных PostgreSQL. Диаграмма подчеркивает модульность системы и каналы взаимодействия.

Эти визуализации помогают понять архитектурный дизайн, облегчают разработку и обеспечивают единое видение системы.

Рисунок 2.2 – Диаграмма последовательности

Рисунок 2.3 – Диаграмма компонентов системы

2.6 Проектирование жизненного цикла системы

Для разработки системы была выбрана итеративная модель жизненного цикла. Этот подход предусматривает последовательное выполнение этапов разработки с возможностью возврата к предыдущим стадиям для уточнения требований и исправления выявленных недочётов. Итеративная модель была выбрана благодаря её гибкости, которая позволяет адаптироваться к изменениям в процессе разработки, а также обеспечивать раннее тестирование и постепенное наращивание функциональности системы.

Процесс разработки организован в виде повторяющихся циклов, каждый из которых включает следующие этапы: анализ требований, проектирование архитектуры и компонентов, реализацию программного кода и тестирование промежуточных результатов. После завершения каждой итерации результаты оцениваются, что позволяет корректировать план дальнейшей работы.

Преимущества итеративной модели заключаются в следующем:

– гибкость: Возможность вносить изменения в требования на любом этапе разработки, что особенно важно для проекта с использованием

машинного обучения, где точность моделей может потребовать доработки;
– раннее выявление ошибок: Тестирование проводится после каждой итерации, что снижает риск накопления критических проблем к финальной стадии;
– постепенное развитие: Функциональность системы наращивается поэтапно, начиная с базовых компонентов (например, сбора данных) и заканчивая сложными функциями (прогнозирование спроса).
Схематическое изображение жизненного цикла представлено на рисунке 2.4.
25
Рисунок 2.4 – Пример работы итерационной модели жизненного цикла
2.7 Проектирование схемы базы данных
Проектирование базы данных является критически важным этапом разработки интеллектуальной системы, так как от эффективности структуры хранения данных зависят производительность, масштабируемость и надёжность системы. Разработанная схема базы данных обеспечивает поддержку всех ключевых функций системы – сбора данных, их обработки, прогнозирования спроса и предоставления аналитики, а также гарантирует гибкость для дальнейшего расширения функциональности.
Схема базы данных включает следующие основные сущности:
– пользователи (users): хранит информацию о пользователях системы, включая учетные данные для аутентификации. Используется микросервисом Auth Service;
– товары (products): содержит основную информацию о товарах, их характеристиках и категоризации. Используется Data Processor Service для хранения обработанных данных;
26
– исторические данные (product_historical_data): Хранит временные ряды [29] с информацией о ценах и продажах товаров за различные периоды. Используется для анализа и создания прогнозов;
– прогнозы (predictions): содержит результаты прогнозирования цен и продаж с привязкой к конкретным товарам. Используется ML Service для записи и API Gateway для чтения;
– сессии (sessions): хранит информацию о пользовательских сессиях и статусе аутентификации. Используется Auth Service;
– топы товаров (top_products): хранит еженедельные топы товаров с наибольшим прогнозируемым ростом спроса и цен. Используется ML Service для записи и API Gateway для чтения и отображения на главной странице пользовательского интерфейса.
Разработанная схема базы данных представлена на рисунке А.5.
Получившаяся база данных будет находиться в контейнере Docker под управлением СУБД PostgreSQL. Это позволит повысить безопасность благодаря дополнительной изоляции.
2.8 Вывод к разделу 2
В данном разделе спроектирована функциональная схема в методологии IDEF0 и разработана микросервисная архитектура системы для автоматизированной оценки спроса на продукт. Созданы компоненты клиентской части на базе React с TypeScript и серверной части, состоящей из специализированных микросервисов. Разработаны UML-диаграммы, визуализирующие логическую модель системы, выбрана итеративная модель

жизненного цикла и спроектирована схема базы данных PostgreSQL. Предложенное решение обеспечивает гибкость, масштабируемость и отказоустойчивость системы [30] в соответствии с современными стандартами программной инженерии [18].

27

3 ТЕХНОЛОГИЧЕСКИЙ РАЗДЕЛ

3.1 Разработка серверной части системы

При реализации каждого модуля было принято решение следовать принципам Чистой архитектуры – архитектурного подхода к проектированию программного обеспечения, цель которого сделать код легко читаемым, тестируемым, расширяемым и независимым от внешних деталей, таких как базы данных, брокеры сообщений или интерфейс пользователя.

Следуя этим принципам, была реализована структура, представленная на рисунке 3.1.

Рисунок 3.1 - структура проекта

Краткое описание для каждого элемента:

– assembly – инициализация и связывание компонентов приложения, создание экземпляров сервисов и репозиториев;

– config – конфигурационные структуры для загрузки параметров из .env файла или переменных окружения;

– controller – обработчики HTTP-запросов, валидация входных данных;

– repository – реализации для доступа к базе данных PostgreSQL и файловой системе;

28

– service – бизнес-логика обработки данных, включая очистку, нормализацию и подготовку для анализа;

– migrations – SQL-скрипты для создания и обновления структуры базы данных;

– scripts – вспомогательные Python-скрипты для обработки данных;

– internal – вспомогательные пакеты и утилиты, используемые внутри сервиса.

Такая структура позволяет достичь высокой модульности кода, упрощает тестирование и обеспечивает четкое разделение ответственности между компонентами сервиса обработки данных.

Сервисы используют систему миграций для управления схемой базы данных PostgreSQL, обеспечивая автоматическое создание необходимых таблиц и индексов при первом запуске или обновлении сервиса. Это гарантирует согласованность схемы данных при развертывании сервиса в различных средах.

Для обеспечения согласованного развертывания и запуска всех компонентов системы используются Docker и Docker Compose. Каждый микросервис имеет собственный Dockerfile, который описывает процесс создания образа: установку зависимостей, копирование исходного кода и настройку точки входа.

Оркестрация микросервисов осуществляется с помощью docker-compose.yml, который определяет взаимосвязи между контейнерами, настраивает сетевые соединения через и конфигурирует переменные окружения для каждого сервиса. Файл также описывает тома для постоянного хранения данных (PostgreSQL, RabbitMQ) и устанавливает зависимости между сервисами, гарантируя правильный порядок запуска. Благодаря такой

организации, вся система запускается единой командой «docker-compose up», что существенно упрощает процесс развертывания как в среде разработки, так и на production-серверах.

29

3.1.1 Разработка модуля авторизации

Сервис авторизации (auth-service) представляет собой компонент системы, отвечающий за аутентификацию пользователей, управление сессиями и обеспечение безопасного доступа к ресурсам системы.

Одна из ключевых функций auth-service — регистрация новых пользователей и их аутентификация. При регистрации сервис проверяет уникальность имени пользователя и email, после чего сохраняет информацию в базе данных PostgreSQL. Важной особенностью реализации является безопасное хранение паролей – они не хранятся в открытом виде, а хешируются с помощью алгоритма bcrypt [17], что обеспечивает защиту даже в случае компрометации базы данных.

Для обеспечения безопасности системы auth-service использует JWT – компактный и самодостаточный способ безопасной передачи информации между сторонами в виде JSON-объекта. Это позволяет реализовать механизм авторизации без необходимости хранения состояния сессии на сервере. На листинге 3.1 показана генерация JWT-токена.

Листинг 3.1 – Генерация JWT-токена

```
func (s *AuthService) GenerateToken(user *models.User) (string, error) {
claims := jwt.MapClaims{
"user_id": user.ID,
"username": user.Username,
"exp": time.Now().Add(time.Hour * 24).Unix(), // Токен действителен 24 часа
}
token := jwt.NewWithClaims(jwt.SigningMethodHS256, claims)
tokenString, err := token.SignedString([]byte(s.config.JWTSecret))
if err != nil {
return "", fmt.Errorf("signing token: %w", err)
}
```

30

Процесс авторизации запроса включает несколько шагов: клиент отправляет запрос с JWT-токеном в заголовке Authorization, API Gateway извлекает токен и отправляет запрос на проверку в auth-service, который проверяет валидность токена и возвращает информацию о пользователе. API Gateway, получив подтверждение, направляет запрос в соответствующий сервис.

3.1.2 Разработка модуля сбора данных

Сервис сбора данных (data-collector-service) представляет собой один из ключевых компонентов разработанной системы, ответственный за получение, первичную обработку и передачу данных о товарах для последующего анализа и прогнозирования спроса. Выделение этой функциональности в отдельный микросервис обусловлено необходимостью изолировать процесс получения данных от их обработки, что обеспечивает более эффективное масштабирование и устойчивость системы в целом.

Основная задача data-collector-service – загрузка данных из внешних источников и их трансформация в унифицированный формат для последующей передачи в сервис обработки данных.

Взаимодействие с другими компонентами системы реализовано через асинхронный обмен сообщениями с использованием брокера RabbitMQ. Это позволяет разделить процессы сбора и обработки данных, обеспечивая их независимость и отказоустойчивость. На листинге 3.2 представлен фрагмент кода, отвечающий за отправку сообщений в очередь.

Листинг 3.2 – Отправка собранных данных в очередь

```
return tokenString, nil
}
func (s *CollectorService) SendToQueue(data *entity.ProductData) error {
body, err := json.Marshal(data)
if err != nil {
```

31

Так же в данном сервисе реализован Планировщик задач (Scheduler) – он управляет периодическим запуском сбора данных согласно настраиваемому расписанию. Это обеспечивает регулярное обновление информации без необходимости ручного вмешательства. Запуск планировщика показан на листинге 3.3.

Листинг 3.3 – Запуск планировщика сбора данных

```
return fmt.Errorf("marshal product data: %w", err)
}
err = s.rabbitClient.PublishMessage(s.config.QueueName, body)
if err != nil {
return fmt.Errorf("publish message to queue: %w", err)
}
s.logger.Infof("Data for product %s sent to queue", data.ProductCode)
return nil
}
func (p *DataProcessor) StartScheduler(ctx context.Context, interval time.Duration) {
p.logger.Infof("Starting scheduler with interval: %v", interval)
if err := p.ProcessData(ctx); err != nil {
p.logger.Errorf("Initial data processing failed: %v", err)
}
ticker := time.NewTicker(interval)
defer ticker.Stop()
for {
select {
case <-ticker.C:
p.logger.Info("Scheduler triggered data processing")
```

32

3.1.3 Разработка модуля подготовки данных

Сервис подготовки данных (data-processor-service) выполняет важную роль в процессе анализа и прогнозирования спроса на товары. Его основная задача – преобразование собранных сырых данных в структурированный и очищенный формат, пригодный для машинного обучения и аналитики [17]. Сервис получает данные из очереди RabbitMQ, куда их отправляет data-collector-service, и выполняет многоэтапную обработку. Этот процесс включает в себя нормализацию значений, удаление дубликатов, обработку пропущенных значений и вычисление дополнительных признаков, которые будут использоваться для прогнозирования [20].

Важной особенностью сервиса является создание и расчет временных

характеристик, таких как скользящие средние, лаги продаж и цен, а также сезонные признаки (день недели, месяц, праздники). Эти признаки существенно повышают точность прогнозирования временных рядов. На листинге Б.1 представлена функция подготовки временных признаков.

Ключевым компонентом сервиса является Python-скрипт, который использует библиотеки pandas для обработки данных и подготовки их для машинного обучения. Скрипт выполняет несколько важных функций:

– преобразование типов данных и интерполяцию пропущенных значений в числовых полях;

– создание временных признаков (день недели, месяц, квартал);

if err := p.ProcessData(ctx); err != nil {

p.logger.Errorf("Scheduled data processing failed: %v", err)}

case <-ctx.Done():

p.logger.Info("Scheduler stopped")

return

}

}

}

– расчет лаговых значений продаж и цен (за 1, 3 и 7 дней);

– вычисление скользящих средних для цен и продаж (за периоды 3 и 7 дней);

– формирование целевых переменных для прогнозирования: цена через 7 дней и суммарные продажи за следующие 7 дней.

На листинге Б.2 представлена функция создания признаков из скрипта.

Важным компонентом сервиса является планировщик, который регулярно запускает процесс обработки данных согласно настраиваемому расписанию. Планировщик обеспечивает актуальность подготовленных данных и автоматизирует весь процесс от получения сырых данных до их преобразования в формат, готовый для машинного обучения.

После обработки данные разделяются на тренировочную и тестовую выборки и сохраняются как в файлах CSV для последующего использования в обучении моделей, так и в базе данных PostgreSQL для долговременного хранения и анализа. Сервис обеспечивает целостность данных на всех этапах обработки благодаря тщательно проработанной системе логирования и обработки ошибок.

3.1.4 Разработка модуля машинного обучения

Сервис машинного обучения (ML-service) представляет собой компонент системы, отвечающий за прогнозирование спроса и цен на товары на основе обработанных данных. Модуль реализован как гибридный сервис, где высокопроизводительный Go-сервер обеспечиваетAPI для взаимодействия с другими компонентами системы, а Python-скрипты выполняют обучение и применение моделей машинного обучения [26].

Основной функционал сервиса включает три ключевые операции: обучение моделей на исторических данных, генерацию прогнозов для конкретных товаров и подготовка подборок наиболее перспективных товаров. Сервис проверяет наличие обученных моделей при запуске и, если они отсутствуют, автоматически запускает процесс обучения, используя подготовленные наборы данных из сервиса обработки.

На листинге 3.4 представлен фрагмент кода сервиса, отвечающий за проверку наличия моделей и их автоматическое обучение при необходимости:

Листинг 3.4 – Проверка наличия моделей и их обучения

Центральным элементом ML-компонента является Python-скрипт, реализующий функциональность для обучения моделей на основе библиотеки LightGBM. Эта библиотека была выбрана благодаря высокой эффективности для задач регрессии с большим количеством категориальных признаков, что характерно для данных о товарах [27].

Процесс обучения моделей включает несколько ключевых этапов:

– загрузка и валидация тренировочных и тестовых данных;

– подготовка признаков, включая преобразование категориальных переменных;

```
func (s *MLPredictionService) CheckModelsExist() bool {
modelFiles := []string{
filepath.Join(s.modelPath, "price_model.pkl"),
filepath.Join(s.modelPath, "sales_model.pkl"),
filepath.Join(s.modelPath, "feature_info.json"),
}
for _, file := range modelFiles {
if _, err := os.Stat(file); os.IsNotExist(err) {
s.logger.Warnf("Model file not found: %s", file)
return false
}
}
s.logger.Info("All model files found")
return true
}
```


– обучение двух отдельных моделей [21]: для прогнозирования цен и для прогнозирования продаж;

– сохранение обученных моделей в формате pickle для последующего использования.

Особенностью реализации является оптимизация гиперпараметров моделей LightGBM, включая ограничение глубины деревьев и применение техники ранней остановки (early stopping) для предотвращения переобучения. Модели обучаются независимо друг от друга на одних и тех же признаках, но с разными целевыми переменными.

Модели работают с разнородными данными о товарах, которые включают как числовые, так и категориальные признаки. На листинге Б.3 показана структура признаков, используемых для прогнозирования. На листинге Б.4 представлен фрагмент кода, отвечающий за предсказание с использованием обученных моделей. Результаты прогнозирования сохраняются в базе данных PostgreSQL для последующего анализа и отображения в пользовательском интерфейсе.

Дополнительно, сервис машинного обучения обеспечивает еженедельное обновление подборок товаров с наибольшим прогнозируемым ростом спроса и цен. Для этого система выполняет пакетное прогнозирование для всех товаров, используя обученные модели LightGBM, и рассчитывает прогнозы спроса и цен. На основе этих прогнозов товары ранжируются по величине прогнозируемого роста спроса и по величине прогнозируемого

увеличения цены, формируя два списка: топ товаров с наибольшим ростом спроса и топ товаров с наибольшим ростом цены. Эти топы обновляются раз в неделю и сохраняются в базе данных PostgreSQL. Такой подход позволяет пользователям оперативно получать актуальную информацию о наиболее перспективных товарах, что способствует принятию обоснованных бизнес-решений.

3.1.5 Разработка API-gateway

Сервис API Gateway является компонентом системы, обеспечивающим единую точку входа для всех запросов от клиентской части. Основная задача сервиса – маршрутизация запросов к соответствующим микросервисам, аутентификация пользователей и кэширование ответов.

Реализованный на Go с использованием фреймворка Gin, API Gateway перехватывает входящие HTTP-запросы, проверяет JWT-токены через Auth Service и направляет запросы к нужным микросервисам. Важной особенностью реализации является локальное кэширование часто запрашиваемых данных, что значительно снижает нагрузку на другие компоненты системы и улучшает время отклика.

Такой подход централизует логику взаимодействия с клиентским приложением, упрощает разработку и поддержку системы, а также обеспечивает единообразие интерфейсов и повышает безопасность за счет изоляции внутренних сервисов от прямого внешнего доступа.

3.2 Разработка клиентской части системы

Клиентская часть системы (ui-service) разработана как современное одностраничное веб-приложение (SPA) с использованием технологий React и TypeScript.

Взаимодействие с серверной частью реализовано через RESTful API с использованием библиотеки Axios для выполнения HTTP-запросов. Для обеспечения типобезопасности и повышения надежности кода применяется TypeScript, который позволяет выявлять потенциальные ошибки на этапе компиляции. Архитектура SPA обеспечивает быструю загрузку и плавное взаимодействие с пользователем, минимизируя запросы к серверу за счет клиентской обработки данных.

Стилизация интерфейса выполнена с использованием подхода utility-first CSS через библиотеку Tailwind CSS, что дает гибкость оформления компонентов и согласованность дизайна во всем приложении.

При переходе на страницу прогнозирования пользователь имеет возможность получать прогнозы спроса и цен для выбранных товаров.

Поддерживается обычный и расширенный режим ввода данных для получения прогнозов. Пример страницы с полученным прогнозом приведен на рисунках 3.2 и 3.3.

Так же для получения истории по уже совершенным прогнозам пользователь может авторизоваться в системе. На рисунке 3.4 показана страница истории совершенных запросов. Пример страницы авторизации приведен на рисунке 3.5.

Рисунок 3.2 – Главная страница с перспективными товарами

Рисунок 3.3 – Страница с получением прогноза

Рисунок 3.4 – Страница истории прогнозов

Рисунок 3.5 – страница входа в аккаунт пользователя

Приложение реализует адаптивную систему оформления с поддержкой светлой и темной цветовых схем. Такой подход к организации пользовательского интерфейса не только соответствует современным стандартам веб-разработки, но и значительно повышает комфорт при работе с системой в различных условиях внешнего освещения, снижая нагрузку на зрение пользователя. Пример интерфейса с активированной светлой темой представлен на рисунке 3.6.

Рисунок 3.6 – Главная страница со светлой темой

3.3 Тестирование системы

Тестирование является критически важным этапом разработки, обеспечивающим корректность работы всех функций системы и соответствие требованиям. Для разработанной интеллектуальной системы оценки спроса было проведено комплексное тестирование как пользовательского интерфейса, так и серверной части с API. Так же в каждом сервере были написаны юнит-тесты и интеграционные тесты, что так же повысило надежность системы.

3.3.1 Чек-лист тестирования

Перед началом тестирования был составлен список основных функций веб-приложения, которые необходимо проверить. Для этого был создан чек-лист, в котором перечислены основные тест-кейсы и состояние их выполнения. Тестирование проводилось в браузерах «Google Chrome» и «Mozilla Firefox». Чек-лист представлен в таблице 3.1.

Таблица 3.1 – Чек-лист тестируемого веб-приложения

Тестируемые функции	Google Chrome	Mozilla Firefox
Регистрация и авторизация		
Регистрация пользователя	Выполнено	Выполнено
Авторизация пользователя	Выполнено	Выполнено
Переключение тем оформления	Выполнено	Выполнено
Прогнозирование		
Просмотр подборок перспективных товаров	Выполнено	Выполнено
Создание стандартного прогноза	Выполнено	Выполнено
Создание минимального прогноза	Выполнено	Выполнено
Визуализация результатов прогноза	Выполнено	Выполнено
Аналитика и отчеты		
Просмотр истории пользователя	Выполнено	Выполнено

3.3.2 Тест-кейсы

Следующим этапом тестирования является создание тест-кейсов. Тест-кейс – это алгоритм действий, по которому предлагается тестировать определенную функцию системы. В тест-кейсах подробно описаны шаги, которые необходимо сделать для подготовки к тесту, сама проверка и ожидаемый результат.

В таблице 3.2 представлены основные тест-кейсы для проверки системы.

Таблица 3.2 – Проверка тест-кейсов

Регистрация пользователя		
Шаги выполнения	Ожидаемый результат	Фактический результат
– Вести данные учетной записи – Нажать кнопку «Зарегистрироваться»	– Данные пользователя добавлены в базу данных – Получен доступ к функционалу системы – Открылась главная страница с подборками товаров	Фактический результат совпадает с ожидаемым
Авторизация пользователя		
Шаги выполнения	Ожидаемый результат	Фактический результат
– Вести данные учетной записи – Нажать кнопку «Войти»	– Получен доступ к функционалу системы – Открылась главная страница с подборками товаров	Фактический результат совпадает с ожидаемым
Переобучение модели прогнозов		
Шаги выполнения	Ожидаемый результат	Фактический результат
– Нажать кнопку «Модель обучена» или «Модель не обучена» (в зависимости от статуса обучения модели) – Дождаться всплывающего уведомления «Модель успешно переобучена»	– Модель переобучена на актуальных данных и сохранена для использования	Фактический результат совпадает с ожидаемым
Получение истории прогноза		
Шаги выполнения	Ожидаемый результат	Фактический результат
– Нажать кнопку «История		

прогнозов»
– Выгружены и получены все
запросы текущего
пользователя
Фактический результат
совпадает с ожидаемым
Получение прогнозов
42
3.3.3 Тестирование API с использованием Swagger
Помимо тестирования пользовательского интерфейса, было проведено тестирование REST API системы с использованием Swagger UI. Этот инструмент позволяет интерактивно взаимодействовать с API без написания кода, что значительно упрощает процесс тестирования.
API Gateway предоставляет документацию Swagger, которая описывает все доступные эндпоинты, методы и параметры.
На рисунке 3.7 представлен интерфейс Swagger UI для тестирования API.
Рисунок 3.7 – Интерфейс Swagger UI
На рисунке 3.8 представлен пример тестирования API авторизации пользователя в системе.
Шаги выполнения Ожидаемый результат Фактический результат
– На главной странице
выбрать режим прогноза –
расширенный или обычный
– Ввести данные о продукте
– Нажать кнопку «Получить
прогноз»
– Получение прогноза на цену
и спрос продукта
– Сохранение в базу данных
информации о проведенном
прогнозе
Фактический результат
совпадает с ожидаемым
43
Аналогичным образом были протестированы все доступные API серверной части системы. Тестирование с использованием Swagger UI позволило проверить корректность работы API.
В ходе тестирования были выполнены запросы для проверки различных сценариев, включая обработку ошибок и валидацию данных. Это позволило убедиться в стабильности и надежности API при взаимодействии с серверной частью системы [24] .
Рисунок 3.8 – тестирование авторизации пользователя в системе
3.4 Расчёт вычислительной и емкостной сложности
В данном разделе высчитывается вычислительная и емкостная сложности основных алгоритмов интеллектуальной системы оценки спроса на продукт. Для оценки алгоритмической сложности используется нотация Big O, которая описывает асимптотическое поведение функций при увеличении размера входных данных. Вычислительная сложность характеризует время выполнения операции, а емкостная - объем памяти, необходимый для хранения

данных и промежуточных результатов.

44

Результаты расчета вычислительной и емкостной сложности

представлены в таблице 3.3.

Таблица 3.3 – Результаты расчета вычислительной и емкостной сложности

Продолжение таблицы 3.3

Где n - количество записей или элементов данных, d - количество

деревьев в ансамбле моделей LightGBM.

Операции API Gateway, такие как registerUser, loginUser и validateToken,

имеют константную сложность O(1), так как они включают фиксированное

количество операций независимо от размера системы. Кэширование ответов

также выполняется за константное время, что значительно улучшает

производительность при повторных запросах.

Функции получения списков (getProductList) имеют линейную

сложность O(n), поскольку необходимо обработать все n элементов.

Наибольшую вычислительную сложность имеют операции обработки

данных и обучения моделей. Функция calculateFeatures с квадратичной

сложностью O(n2) выполняет вычисление скользящих средних и лаговых

Функция Описание функции Вычислительная

сложность

Емкостная

сложность

registerUser Регистрация

пользователя

O(1) O(1)

loginUser Авторизация

пользователя

O(1) O(1)

validateToken Проверка JWT-токена O(1) O(1)

processDataBatch Обработка партии

данных

O(n log n) O(n)

calculateFeatures Вычисление

признаков

O(n2) O(n)

removeDuplicates Удаление дубликатов O(n log n) O(n)

trainModels Обучение моделей O(d × n log n) O(d × n)

predictPrice Прогнозирование

цены

O(log n) O(1)

predictSales Прогнозирование

продаж

O(log n) O(1)

getTopProducts Расчет топа товаров O(n log n) O(n)

cacheResponse Кэширование ответа

API

O(1) O(1) [23]

45

значений для продуктов в разных временных окнах. Обучение моделей

(trainModels) достигает сложности O(d × n log n [23]) из-за алгоритмов построения

деревьев решений в LightGBM.

При этом операции прогнозирования (predictPrice, predictSales) имеют логарифмическую сложность O(log n), так как для предсказания требуется только прохождение по уже построенным деревьям. Это обеспечивает высокую скорость генерации прогнозов в режиме реального времени.

Большинство функций системы имеют линейную или константную емкостную сложность, что позволяет эффективно использовать память. Только функция обучения моделей требует существенных объемов памяти, пропорциональных количеству деревьев и размеру обучающей выборки (O(d × n)).

Проведенный анализ показывает, что система обладает хорошей производительностью для пользовательских операций благодаря оптимальной сложности API-функций и кэширования. Основная вычислительная нагрузка сосредоточена в процессах обработки данных и обучения моделей, которые выполняются асинхронно и по расписанию, не влияя на отзывчивость пользовательского интерфейса.

3.5 Вывод к разделу 3

В данном разделе реализована технологическая часть системы оценки спроса, включающая серверную часть на принципах чистой архитектуры и клиентскую часть на React с TypeScript. Разработаны модули авторизации, сбора и подготовки данных, машинного обучения и API Gateway. Проведено тестирование с использованием чек-листов, тест-кейсов и Swagger UI, а анализ сложности подтвердил высокую производительность. Решение соответствует современным стандартам программной инженерии, обеспечивая надежность и масштабируемость.

4 ЭКОНОМИЧЕСКИЙ РАЗДЕЛ

4.1 Организация и планирование работ по теме

В рамках ВКР будет спроектирована и разработана интеллектуальная система для автоматизированной оценки спроса на продукт с использованием технологий машинного обучения. Данная система позволит бизнесу получать точные прогнозы спроса на основе анализа данных из различных источников [13].

В процессе выполнения работы было задействовано три человека:

1) руководитель ВКР (Аждер Татьяна Борисовна, кандидат технических наук, доцент кафедры ИиППО) – отвечает за четкую формулировку задачи, при необходимости вносит корректировки в процесс работы и контролирует её выполнение;

2) консультант по экономической части (Чижанькова Инна Владимировна, кандидат экономических наук, доцент кафедры экономики) – консультирует при выполнении экономической части выпускной квалификационной работы;

3) разработчи [11] к [5] (Мухаметшин Александр Ринатович, студент ИКБО-20-21) – анализирует предметную область, производит проектирование системы, разрабатывает интеллектуальную систему.

Модель взаимодействия участников работы представлена на рисунке 4.1. Модель разработана с помощью ПО draw.io.

Рисунок 4.1 – Модель взаимодействия участников

4.1.1 Этапы разработки

Определив участников работы, можно подробно расписать этапы разработки, и кто на каждом этапе принимает участие. Этапы разработки

отображены на таблице 4.1. 5

47

Таблица 4.1 – Этапы разработки

Продолжение таблицы 4.1

No Название этапа Исполнитель Трудоемкость,

дни/чел

Продолжительность

работ, дни 5

1 Исследовательский раздел

9

1.1 Анализ существующих

конкурентных решений Разработчик 3

1.2

Формирование требований

к продукту на основе

существующих разработок

Разработчик 2

Руководитель 1

1.3 Выбор инструментов и

методов создания системы Разработчик 2

1.4 Постановка задачи к

проектированию и

разработке системы

Разработчик 2

Руководитель 1

2 Проектный раздел

16

2.1 Проектирование

функциональной схемы

Разработчик 2

Руководитель 1

2.2 Проектирование

архитектуры системы

Разработчик 3

Руководитель 1

2.3 Проектирование

клиентской части системы

Разработчик 2

Руководитель 1

2.4 Проектирование серверной

части системы

Разработчик 3

Руководитель 1

2.5

Разработка диаграмм

логической модели

системы

Разработчик 2

2.6 Проектирование

жизненного цикла системы

Разработчик 1

Руководитель 1

2.7 Проектирование схемы

базы данных

Разработчик 3

Руководитель 1

3 Технологический раздел

35

3.1 Разработка серверной

части системы Разработчик 14

3.2 Разработка клиентской

части системы Разработчик 14

3.3 Тестирование системы Разработчик 5

Руководитель 2

3.4 Расчёт вычислительной и

емкостной сложности

Разработчик 2

Руководитель 1

4 Экономический раздел

4.1 9

Организация и

планирование работ по

теме

Разработчик 5

Руководитель 2

Консультант 1

48

В итоге построения таблицы с этапами разработки можно наблюдать, что

разработка системы займет 69 дней.

4.1.2 Календарный план выполнения работ

После определения этапов разработки можно составить календарный

план выполнения работ. Данный план отображен на таблице 4.2.

Таблица 4.2 – Календарный план выполнения работ

Продолжение таблицы 4.2

4.2 Расчет стоимости

проведения рабо 11 т по теме

Разработчик 4

Руководитель 2

Консультант 1

Итого 69

Этап Дата начала Дата

окончания

Количество

рабочих дней Исполнители

1 Исследовательский раздел

1.1 Анализ существующих

конкурентных решений 10.02.2025 12.02.2025 3 Разработчик

1.2 Формирование требований к

продукту на основе существующих

разработок

13.02.2025 14.02.2025 2 Разработчик

Руководитель

1.3 Выбор инструментов и методов

создания системы 15.02.2025 17.02.2025 2 Разработчик

1.4 Постановка задачи к

проектированию и разработке

системы

18.02.2025 19.02.2025 2 Разработчик

Руководитель

2 Проектный раздел

2.1 Проектирование

функциональной схемы 20.02.2025 21.02.2025 2 Разработчик

Руководитель

2.2 Проектирование архитектуры

системы 24.02.2025 26.02.2025 3 Разработчик

Руководитель

2.3 Проектирование клиентской

части системы 27.02.2025 28.02.2025 2 Разработчик

Руководитель

2.4 Проектирование серверной

части системы 01.03.2025 04.03.2025 3 Разработчик

Руководитель

2.5 Разработка диаграмм

логической модели системы 05.03.2025 06.03.2025 2 Разработчик

2.6 Проектирование жизненного

цикла системы 07.03.2025 07.03.2025 1 Разработчик

Руководитель

2.7 Проектирование схемы базы

данных 10.03.2025 12.03.2025 3 Разработчик

Руководитель

3 Технологический раздел

3.1 Разработка серверной части

системы 13.03.2025 28.03.2025 14 Разработчик

3.2 Разработка клиентской части 29.03.2025 14.04.2025 14 Разработчик

49

Составив календарный план работы, можно наблюдать, что с учетом выходных дней, полная разработка системы займет 86 дней.

Для визуализации календарного графика была составлена диаграмма Ганта с помощью ПО GanttPRO, которая представлена на рисунке В.1.

4.2 Расчет стоимости проведения работ по теме

Себестоимость анализа, проектирования и разработки интеллектуальной системы оценки спроса на продукт складывается из затрат по следующим статьям [7]:

«Сырье и материалы» + ТЗР (15%) от ∑ итого по [5] материалам;

«Основная заработная плата»;

«Дополнительная заработная плата» 20-30% от основной заработной платы»;

«Страховые взносы» - 30% от ФОТ, а также 0,2% ставка за травматизм»;

«Амортизация»;

«Прочие расходы».

4.2.1 Статья «Сырье и материалы»

К данной статье относятся материалы, полуфабрикаты и изделия 6 , что затрачиваются в процессе создания системы и подготовки ВКР. В общую стоимость также будет включены транспортно-заготовительные расходы, которые будут взяты на уровне 15% от общей стоимости затрат по данной статье 6 . Стоимость материалов представлена на таблице 4.3.

системы

3.3 Тестирование системы 15.04.2025 19.04.2025 5 Разработчик

Руководитель

3.4 Расчёт вычислительной и

емкостной сложности 21.04.2025 22.04.2025 2 Разработчик

Руководитель

4 Экономический раздел

4.1 Организация и планирование

работ по теме 23.04.2025 28.04.2025 5

Разработчик

Руководитель

Консультант

4.2 Расчет стоимости проведения

работ по теме 29.04.2025 06.05.2025 4

Разработчик

Руководитель

Консультант

50

Таблица 4.3 – Стоимость материалов

Таким образом общая сумма затрат по статье «Сырье и материалы» составит 4140 руб.

4.2.2 Статья «Основная заработная плата»

К данной статье относится оплата труда научных работников, инженерно-технических работников и рабочих, непосредственно занятых выполнением конкретной работы, а также заработная плата работников внештатного состава, привлекаемых к ее выполнению. Расчет заработной платы будет происходить исходя из 5 затраченных дней на разработку на каждом этапе. Оплату труда руководителя и консультанта возьмем с сайта МИРЭА, и она составит 161230 рублей.

Дневная тарифная ставка (ТС) для месячного оклада (ОК) будет рассчитана по формуле (1).

где:

ОК – месячный оклад участника проекта;

НРВ – годовой фонд рабочего времени при рабочей неделе в 40 часов.

Обозначив необходимые данные, можно составить таблицу 4.4, в который будет описана основная заработная плата каждого исполнителя проекта.

Таблица 4.4 – Расчет основной заработной платы

No

п/ 2 п 18

Наименование

материалов

Единица

измерения Количество Цена за единицу

(руб.)

Стоимость

(руб.)

1 Ручка шт. 1 100 100

2 Бумага А4 упаковка 12 1 400 400

3 Флешка 16 ГБ шт. 1 1000 1000

4 Картридж для принтера шт. 1 2100 2100

Итого материалов 3600

Транспортно-заготовительные расходы 540

Итого 4140

ТС= ОК×12НРВ = ОК×12299 руб./д. 1)

No

п/п Наименование этапа Исполнитель Месячный

оклад (руб.) 18

Трудоемкость

(чел/дни)

Оплата за

день

(руб.)

Оплата за

этап

(руб.)

1 Исследовательский Руководитель 161230 2 6471 12942 27

51

Таким образом общая сумма затрат по статье 27 «Основная заработная

плата» составит 121116 руб.

4.2.3 Статья «Дополнительная заработная плата»

К данной статье относятся выплаты, предусмотренные

законодательством о труде за неотработанное по уважительным причинам

время; оплата очередных и дополнительных отпусков; времени, связанного с

выполнением государственных и общественных обязанностей; выплата

вознаграждения за выслугу лет и т.п.

В среднем расходы по данной статье составляют 20-30% от суммы

основной заработной платы и рассчитываются по формуле (2).

где:

– ДЗП – дополнительная заработная плата;

– ОЗП – основная заработная плата.

– Для расчета значение ДЗП принято в размере 20%.

Таким образом, дополнительная заработная плата составит:

Д ЗП =0,2 × 121116 = 24223,2 руб.

В данной статье также вводится понятие фонда оплаты труда (ФОТ).

ФОТ представляет собой сумму основной заработной платы и дополнительной

заработной платы 17 . Таким образом, ФОТ данного проекта составит:

ФОТ = 121 116 + 24223,2 = 145 339,2 руб.

4.2.4 Статья «Страховые взносы»

К данной статье относятся расход 6 ы на обязательные страхования, такие

как:

раздел Разработчик 4000 9 161 1449

2 Проектный раздел Руководитель 161230 6 6471 38826

Разработчик 4000 16 161 2576

3 Технологический

раздел

Руководитель 161230 3 6471 19413

Разработчик 4000 35 161 5635

4 Экономический

раздел

Руководитель 161230 4 6471 25884

Консультант 161230 2 6471 12942

Разработчик 4000 9 161 1449

Итого 121116

ДЗП= 20…30 %× ОЗП (2)

52

обязательное пенсионное страхование;

обязательное социальное страхование на случай временной нетрудоспособности и в связи с материнством;

обязательное медицинское страхование.

Для данной статьи установлен единый тариф страховых взносов в виде 30% от ФОТ. Также в расчёте необходимо учесть ставку взносов на травматизм, которая может составлять от 0,2 9 % до 12 8,5%. Для РТУ МИРЭА данная ставка составляет 0,2%.

СВ = 0,302× 145 339,2 ≈ 43892,4 руб.

Таким образом, сумма затрат по статье «Страховые взносы» составит 43892,4 руб.

4.2.5 Статья «Амортизация»

К данной статье относятся расходы на закупку оборудования, на котором производилась работа над проектом и отчетом. Данные расходы будут считаться с помощью амортизации.

Амортизация – это отчисления части стоимости основных фондов (например, стоимости оборудования) для возмещения их износа. Амортизация включается в издержки производства.

В данной работе использовался персональный компьютер, ноутбук и принтер. Срок полезного использования персонального компьютера, ноутбука и принтера составляет 36 месяцев.

Амортизационные отчисления будут вычисляться по формуле (3).

где:

A — месячная сумма амортизационных отчислений;

C — первоначальная стоимость объекта;

T — срок полезного использования в месяцах 1 .

Обозначив все необходимые данные, можно составить таблицу 4.5, где

A= СТ 3)

53

будут подсчитаны амортизационные отчисления.

Таблица 4.5 – Амортизационные отчисления

Таким образом, сумма затрат по статье «Амортизация» составит 22688,9 руб.

4.2.6 Статья «Прочие расходы»

К данной статье относятся расходы, что не относятся к производству напрямую, только косвенно, как, например, содержание и ремонт зданий, сооружений, оборудования, инвентаря.

Сумма данных расходов определяется по формуле (4).

где:

– ОЗП – основная заработная плата;

– В данном расчете значение НР принято за 100%.

Таким образом, сумма затрат по статье «Прочие расходы» составит 121116 руб [8] .

4.2.7 Полная себестоимость работ

Подсчитав расходы по всем статьям, можно составить таблицу 4.6, в которой будет отображено полная стоимость работ.

Таблица 4.6 – Полная себестоимость проекта

Продолжение таблицы 4.6

Наименование оборудования | Первоначальная стоимость объекта (руб.) | Срок полезного использования (мес.) | Месячная сумма амортизационных отчислений (руб.) | Период эксплуатации в месяцах | Сумма (руб.)

Ноутбук [1] 100000 36 2777,7 4 11 111,1

Сервер 104200 36 2 894,4 4 11 577,8

Итого22 688,9

НР= 100…130 %×ОЗП (4)

No п/п Номенклатура статей расходов Затраты (руб.) Доля затрат (%)

1 Сырье и материалы 4140,00 1,2

2 Основная заработная плата 121116,00 35,9

3 Дополнительная заработная плата 24223,20 7,2

4 Страховые взносы 43892,40 13,0

5 Амортизация 22688,90 6,7

6 Прочие расходы 121116,00 35,9

Итого 337176,5 100,0

54

Для визуализации долевого состава статей затрат в общей себестоимости представим круговую диаграмму [5] на рисунке 4.2. На данной диаграмме будут отражены затраты по каждой из статей, а также их доля затрат в процентах.

Рисунок 4.2 – Круговая диаграмма долей затрат

Полученные результаты работы будут использоваться внутри университета (организации), поэтому расчет договорной цены не целесообразен.

4.3 Выводы к разделу 4

В четвертом разделе было выполнено планирование работ по теме “Интеллектуальная система оценки спроса на продукт”, а также был проведен расчет стоимость затра 9 т на реализацию поставленной задачи.

ЗАКЛЮЧЕНИЕ

В рамках выполнения выпускной квалификационной работы проведен детальный анализ предметной области, связанной с разработкой интеллектуальной системы оценки спроса на продукт. Изучены существующие аналоги, такие как Ozon Seller, Moneyplace и MPStats, что позволило выделить их сильные и слабые стороны, а также определить ключевые требования к разрабатываемой системе. На основе проведенного анализа выбраны современные инструменты и технологии разработки: языки программирования Python и Go, фреймворки для реализации микросервисов, базы данных PostgreSQL, а также технология контейнеризации Docker, обеспечивающая изолированную и масштабируемую среду для развертывания системы.

Спроектирована микросервисная архитектура системы, включающая модули сбора данных, их обработки, прогнозирования спроса с использованием машинного обучения и предоставления аналитики через веб-интерфейс и API.

В ходе выпускной квалификационной работы реализованы ключевые компоненты системы: разработаны и интегрированы модули сбора и обработки данных, а также модуль машинного обучения для прогнозирования спроса. Применение современных технологий обеспечило гибкость и масштабируемость системы. Результаты работы демонстрируют создание высокоточной и адаптивной интеллектуальной системы, способной эффективно прогнозировать спрос на продукт. Разработанное решение выделяется на фоне аналогов благодаря универсальности источников данных, использованию алгоритмов машинного обучения и простоте интеграции, что делает его ценным инструментом для оптимизации бизнес-процессов в условиях цифровизации.

Within the framework of the final qualification work a detailed analysis of the subject area related to the development of an intelligent system for assessing the demand for a product was carried out. The existing analogs such as Ozon Seller,

Moneyplace and MPStats were studied, which allowed to highlight their strengths and weaknesses, as well as to determine the key requirements for the developed system. Based on the analysis, modern development tools and technologies were selected: Python and Go programming languages, frameworks for implementing microservices, PostgreSQL databases, as well as Docker containerization technology, which provides an isolated and scalable environment for application deployment.

The microservice architecture of the system was designed, including modules of data collection, data processing, demand forecasting using machine learning and providing analytics via web interface and API.

In the course of the final qualification work the key components of the system were realized: the modules of data collection and processing, as well as the module of machine learning for demand forecasting were developed and integrated. Application of modern technologies provided flexibility and scalability of the

system. The results of the work demonstrate the creation of a highly accurate and adaptive intelligent system capable of effectively forecasting demand for a product. The developed solution stands out from its peers due to the versatility of data sources, the use of machine learning algorithms and ease of integration, which makes it a valuable tool for optimizing business processes in the context of digitalization.

57

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Карпов А.В., Сидоров Е.Н. Оптимизация управления запасами с применением методов прогнозирования спроса // Логистические системы в глобальной экономике. – 2023. – No 3. – С. 78-92.

2. Петрова М.И. Сравнительный анализ точности методов машинного обучения в задачах прогнозирования рыночного спроса // Вестник искусственного интеллекта. – 2024. – No 1. – С. 145-158.

3. СМКО МИРЭА 7.5.1/03.П.30-19 Положение о выпускной квалификационной работе обучающихся по образовательным программам высшего образования – программам бакалавриата, программам специалитета, программам магистратуры. – М.: РТУ МИРЭА, 2019. – 56 с.

4. ГОСТ 7.32-2017. Система стандартов по информации, библиотечному и издательскому делу. Отчет о научно-исследовательской работе. Структура и правила оформления. – М.: Стандартинформ, 2018. – 32с.

5. ГОСТ Р 7.0.100-2018. Система стандартов по информации, библиотечному и издательскому делу. Библиографическая запись. Библиографическое описание. – М.: Стандартинформ, 2018. – 128 с.

6. СМКО МИРЭА 7.5.1/03.П.67-19 Положение о проверке выпускных квалификационных работ на объем заимствований и размещения в электронно-библиотечной системе. – М.: РТУ МИРЭА, 2019. – 18 с.

7. ФГОС ВО 3++ по направлению подготовки 09.03.04 Программная инженерия (уровень бакалавриат) [Электронный ресурс]. – Режим доступа: http://fgosvo.ru/uploadfiles/FGOS%20VO%203++/Bak/090304_B_3_17102017.pdf (дата обращения: 16.04.2025).

8. Ozon Seller [Электронный ресурс]. – Режим доступа: https://seller.ozon.ru/ (дата обращения: 08.04.2025).

9. Moneyplace [Электронный ресурс]. – Режим доступа: https://moneyplace.io/ (дата обращения: 08.04.2025).

10. MPStats [Электронный ресурс]. – Режим доступа: https://mpstats.io/ (дата обращения: 08.04.2025).

58

11. Gin Web Framework [Electronic resource]. – URL: https://gin-gonic.com/ (accessed: 15.04.2025).

12. Integration Definition for Function Modeling (IDEF0). – National Institute of Standards and Technology, 2021. – 128 p.

13. React: A JavaScript library for building user interfaces [Electronic resource]. – URL: https://reactjs.org/ (accessed: 17.04.2025).

14. JSON Web Tokens [Electronic resource]. – URL: https://jwt.io/ (accessed: 06.05.2025).

15. Microsoft Research. LightGBM: A Highly Efficient Gradient Boosting Decision Tree [Electronic resource]. – URL: https://github.com/microsoft/LightGBM (accessed: 10.04.2025).

16. Huyen J. Designing Machine Learning Systems. – O'Reilly Media,

2022. – 382 p.
17. Lakshmanan V., Robinson S., Munn M. Machine Learning Design Patterns: Solutions to Common Challenges in Data Preparation, Model Building, and MLOps. – O'Reilly Media, 2021. – 408 p.
18. Newman S. Building Microservices. – O'Reilly Media, 2021. – 616 p.
19. Donovan A., Kernighan B. The Go Programming Language. – Addison-Wesley Professional, 2021. – 400 p.
20. McKinney W. Python for Data Analysis: Data Wrangling with pandas, NumPy, and Jupyter. – O'Reilly Media, 2022. – 576 p.
21. Гудфеллоу Я., Бенджио И., Курвилль А. Глубокое обучение. – ДМК Пресс, 2021. – 652 с.
22. Филиппов А.А. Прогнозирование спроса в маркетплейсах: современные подходы и методы // Вестник цифровой экономики. – 2023. – No 2. – С. 45-58.
23. Белов С.В., Морозова В.А. Применение методов машинного обучения для прогнозирования спроса на товары // Информационные технологии в экономике и управлении. – 2024. – Т. 14, No 1. – С. 112-125.
59
24. Docker Documentation [Electronic resource]. – URL: https://docs.docker.com/ (accessed: 10.05.2025).
25. PostgreSQL Documentation [Electronic resource]. – URL: https://www.postgresql.org/docs/ (accessed: 15.05.2025).
26. Huyen J. Practical MLOps: Operationalizing Machine Learning Models. – O'Reilly Media, 2022. – 304 p.
27. Воронцов К.В. Машинное обучение: курс лекций [Электронный ресурс]. – Режим доступа: http://www.machinelearning.ru/wiki/index.php?title=Машинное_обучение_(курс_лекций,_К.В.Воронцов) (дата обращения: 05.05.2025).
28. Иванов Д.С. Микросервисная архитектура в современных бизнес-приложениях // Программная инженерия. – 2023. – Т. 5, No 2. – С. 78-89.
29. Петров А.И., Смирнов Б.К. Анализ временных рядов в задачах прогнозирования спроса // Прикладная информатика. – 2024. – No 3. – С.45-62.
30. Saltzer J., Kaashoek M. Principles of Computer System Design: An Introduction. – Morgan Kaufmann, 2021. – 576 p.
60
Приложение А
Проектирование
Рисунок А.1 - Контекстная диаграмма функциональной схемы
61
Рисунок А.2 – Декомпозиция функциональной схемы
Рисунок А.3 – Декомпозиция блока «Обработка данных»
62
Рисунок А.4 – Декомпозиция блока «Прогнозирование спроса»
Рисунок А.5 – Схема базы данных системы
63
Приложение Б
Разработка
Листинг Б.1 – Расчет временных признаков для данных

```
func (p *Processor) CalculateTimeFeatures(productData []models.ProductData) []models.EnrichedProductData {
groupedData := make(map[string][]models.ProductData)
```

```
for _, item := range productData {
groupedData[item.ProductCode] = append(groupedData[item.ProductCode], item)}
var enrichedData []models.EnrichedProductData
for _, items := range groupedData {
sort.Slice(items, func(i, j int) bool {
return items[i].Date.Before(items[j].Date)
})
// Расчет лагов и скользящих средних
for i := range items {
enriched := models.EnrichedProductData{
ProductData: items[i], }
// Добавление временных признаков
enriched.DayOfWeek = items[i].Date.Weekday().String()
enriched.Month = items[i].Date.Month().String()
enriched.IsWeekend = items[i].Date.Weekday() == time.Saturday || items[i].Date.Weekday() == time.Sunday
// Расчет лагов для не первых элементов
if i > 0 {
enriched.PriceLag1 = items[i-1].Price
enriched.SalesQuantityLag1 = items[i-1].SalesQuantity}
// Расчет скользящих средних
if i >= 3 {
var priceSum, salesSum float64
64
```

Листинг Б.2 – Создание временных признаков для данных

```
for j := i-3; j < i; j++ {
priceSum += items[j].Price
salesSum += float64(items[j].SalesQuantity)
}
enriched.PriceRollingMean3 = priceSum / 3
enriched.SalesQuantityRollingMean3 = salesSum / 3
}
enrichedData = append(enrichedData, enriched)
}}
return enrichedData
}
def create_features(df):
"""Создание признаков для модели."""
logger.info("Creating features")
# Добавление временных признаков
df['day_of_week'] = df['date'].dt.dayofweek
df['month'] = df['date'].dt.month
df['quarter'] = df['date'].dt.quarter
# Сортировка данных
df = df.sort_values(['product_name', 'date'])
# Лаги и скользящие средние
lag_periods = [1, 3, 7]
rolling_periods = [3, 7]
for product in df['product_name'].unique():
product_df = df[df['product_name'] == product]
if len(product_df) < 7: # Уменьшен порог до 7 записей
```

65

Листинг Б.3 – Структура признаков, используемых для прогнозирования

Продолжение листинга Б.3

```
continue
# Лаги
for lag in lag_periods:
df.loc[df['product_name'] == product, f'sales_quantity_lag_{lag}'] = product_df['sales_quantity'].shift(lag)
df.loc[df['product_name'] == product, f'price_lag_{lag}'] = product_df['price'].shift(lag)
# Скользящие средние
for window in rolling_periods:
df.loc[df['product_name'] == product, f'sales_quantity_rolling_mean_{window}'] = (
product_df['sales_quantity'].rolling(window=window).mean().values
)
df.loc[df['product_name'] == product, f'price_rolling_mean_{window}'] = (
product_df['price'].rolling(window=window).mean().values
)
def _prepare_features(self, df: pd.DataFrame):
# Категориальные признаки
categorical_features = ['brand', 'region', 'category', 'seller',
'day_of_week', 'month', 'quarter']
# Числовые признаки
numerical_features = [
'price', 'original_price', 'discount_percentage', 'stock_level',
'customer_rating', 'review_count', 'delivery_days', 'is_weekend',
'is_holiday', 'sales_quantity_lag_1', 'price_lag_1',
'sales_quantity_lag_3', 'price_lag_3', 'sales_quantity_lag_7',
'price_lag_7', 'sales_quantity_rolling_mean_3', 'price_rolling_mean_3',
```

66

Листинг Б.4 – Функция прогнозирования

```
'sales_quantity_rolling_mean_7', 'price_rolling_mean_7'
]
# Объединение признаков
feature_names = numerical_features + categorical_features
# Преобразование категориальных признаков
for cat_feat in categorical_features:
if cat_feat in df.columns:
df[cat_feat] = df[cat_feat].astype('category')
return df[feature_names], feature_names, categorical_features
def predict(self, product_data: Dict[str, Any]) -> Dict[str, float]:
if self.price_model is None or self.sales_model is None:
if not self.load_models():
raise ValueError("Models not trained or loaded properly")
# Преобразование данных в DataFrame
df = pd.DataFrame([product_data])
# Подготовка признаков для прогнозирования
for cat_feat in self.categorical_features:
if cat_feat in df.columns:
df[cat_feat] = df[cat_feat].astype('category')
# Прогнозирование
X = df[self.feature_names]
```

price_pred = self.price_model.predict(X)[0]

sales_pred = self.sales_model.predict(X)[0]

67

return {

"predicted_price": float(price_pred),

"predicted_sales": float(sales_pred)

}

68

Приложение В

Рисунок В.1 – Диаграмма Ганта календарного графика