МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Тольяттинский государственный университет»

Кафедра «Прикладная математика и информатика»
(наименование)

09.03.03 Прикладная информатика
(код и наименование направления подготовки, специальности)

Разработка социальных и экономических информационных систем
(направленность (профиль) / специализация)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

на тему «Разработка и внедрение скрипта для автоматизированного парсинга данных с сайтов по продаже автозапчастей»

Обучающийся Д.В. Ямкин
(Инициалы Фамилия) (личная подпись)

Руководитель к.э.н., доцент, Т.А. Раченко
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Консультант к.п.н., доцент, А.В. Егорова
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Тольятти 2024

Аннотация

Бакалаврская работа на тему «Разработка и внедрение скрипта для автоматизированного парсинга данных с сайтов по продаже автозапчастей».

Объектом исследования выпускной квалификационной работы является бизнес-процесс закупочной деятельности в ООО «Торговый дом Токус».

Предметом исследования выпускной квалификационной работы является информационная система автоматизированного парсинга данных с сайтов по продаже автозапчастей в условиях ООО «Торговый дом Токус».

Целью исследования является разработка информационной системы для автоматизации закупочной деятельности для отдела продаж.

Для достижения поставленной цели необходимо решить следующие задачи: выполнить анализ деятельности ООО «Торговый дом Токус», построить модели бизнес-процессов закупочной деятельности компании и выявить недостатки, проанализировать текущие решения в области автоматизации закупочной деятельности, разработать структуру данных для решения, обосновать выбор инструмента реализации, описать функционал системы и рассчитать экономический эффект от внедрения.

Работа включает в себя введение, три основных главы, заключение и список используемой литературы.

Общий объем документа - 60 листов, содержащих 39 рисунков, 3 таблицы. В списке используемой литературы и используемых источников представлены 25 информационных источников.

ABSTRACT

The title of the graduation work is « Development and Implementation of a Script for Automated Data Parsing from Auto Parts Sales Websites ».

The senior paper consists of an introduction, three parts, a conclusion, tables and a list of references including foreign sources.

The object of the graduation work is the information system for automated data parsing from auto parts sales websites under the conditions. We touch upon the problem of process automation at «Torgoviy dom Tokus» LLC.

The aim of the work is to develop an information system to automate the purchasing activities of the sales department.

The graduation work may be divided into several logically connected parts which are: analysis of the activities of Tokus Trading House LLC; analysis of existing procurement activities; analyze current solutions in the field of procurement automation; justification of the need to automate procurement activities for the sales department; choosing a development tool; logical and physical modeling; development and testing of the solution.

The special part of the project gives details about results of implementing an automated purchasing solution for the sales department.

In conclusion we'd like to stress that this work is relevant not only in automating the processes for the sales department at «Torgoviy dom Tokus» LLC, but also for other companies related to procurement activities

Оглавление

Введение

Актуальность разработки информационных систем для автоматизированного парсинга данных с сайтов по продаже автозапчастей обусловлена необходимостью оптимизации работы специалистов, осуществляющих поиск и закупку деталей для машин на сайтах по продаже автозапчастей, работу с приходными закупочными накладными. Внедрение автоматизированного парсинга данных в процесс поиска необходимых комплектующих для заказчика по оптимальной цене и дате доставки позволяет сократить время на обработку одной заявки.

Объект исследования: закупочная деятельность в ООО «Торговый дом Токус».

Предметом исследования выпускной квалификационной работы является анализ, разработка и внедрение скрипта для автоматизированного парсинга данных с веб-сайтов, специализирующихся на продаже автозапчастей в условиях компании ООО «Торговый дом Токус».

Целью данной работы является разработка и внедрение скрипта для автоматизированного парсинга данных с сайтов по продаже автозапчастей.

Задачи работы:

1. Исследование деятельности компании ООО «Торговый дом Токус», в частности отдела закупок;
2. Изучение основных принципов парсинга данных, методов и инструментов, обзор;
3. Определение функциональных и нефункциональных требований к скрипту парсинга данных;
4. Выбор средств реализации и обоснование выбора;
5. Разработка логической модели данных;
6. Написание и отладка скрипта;
7. Тестирование скрипта на различных источниках данных;

8. Разработка механизмов обработки ошибок и обновления данных;
9. Оценка производительности и эффективности скрипта.

В первой главе было проведено функциональное моделирование процесса закупки автозапчастей. Была рассмотрена организационная структура компании, детально описаны процессы ведения закупок. Выполнено построение модели бизнес-процессов закупочной деятельности, а также проанализирован перечень недостатков, связанных с ручным режимом поиска автозапчастей на различных сайтах. В результате анализа существующих систем автоматизации поиска автозапчастей на сайтах было выявлено, что их функционал не полностью соответствует специфике исследуемой организации и требуется разработка и внедрение скрипта для автоматизированного парсинга данных с сайтов по продаже автозапчастей.

Во второй главе было проведено построение логической модели информационной системы, направленной на оптимизацию процессов закупки автозапчастей. Эта модель включает различные типы связей между сущностями и определяет классификаторы, играющие важную роль в функционировании системы. Особое внимание было уделено выходным документам: их структуре, содержанию и назначению. Отдельным этапом было определение реквизитного состава информационных объектов, что позволило учесть все детали и особенности, связанные с данными, обрабатываемыми в системе.

В третьей главе была выполнена разработка архитектуры клиент-серверной системы, написан и отлажен скрипт на языке программирования Python; выполнено тестирование и отладка скрипта на различных источниках данных; разработаны механизмы обработки ошибок, а также выполнена оценка производительности и эффективности скрипта.

Глава 1 Функциональное моделирование процесса автоматизации поиска автозапчастей для закупок в компании ООО «Торговый дом Токус»

1.1 Технико-экономическая характеристика отдела закупа организации

Профиль деятельности компании ООО «Торговый дом Токус» связан с производством и реализацией автозапчастей по всему миру, бред автозапчастей под котором компания представлена на рынке – «CR». Клиентами компании являются: автодилеры, автосалоны, СТО, мастерские, физические лица. Ниже, на рисунке 1 представлена организационная структура компании.

Рисунок 1 – Схема организационной структуры ООО «Торговый дом Токус»

Отдел закупок исследуемой компании включает в себя менеджера по рекламации который ответственный за управление и решение проблем, связанных с жалобами и рекламациями со стороны клиентов. А также

менеджеры направления ответственные за планирование, координацию и контроль деятельности внутри организации с целью эффективного достижения стратегических целей компании, а именно закупка автозапчастей.

Процесс отдела закупок исследуемой компании включает в себя:

– получение накладных на ежедневную закупку автозапчастей;
– согласование количества необходимых автозапчастей;
– поиск автозапчастей на сайтах;
– закупка автозапчастей согласно политики закупки компании.

Рассмотрим подробнее каждый из пунктов:

– Получение накладных на ежедневную закупку автозапчастей заключается в том, что, отдел закупок составляет список необходимых автозапчастей на основе текущих заявок на день.

– Согласование количества необходимых автозапчастей состоит в том, что, заявка на автозапчасти может потребовать согласования с руководством или финансовым отделом компании, особенно если она связана с крупными расходами. Руководство может пересмотреть список и утвердить его, учитывая бюджетные ограничения и стратегические приоритеты.

– Поиск автозапчастей на сайтах происходит после того, как заявка утверждена, выбирается сайт, на котором можно произвести закупку согласно специфике автозапчасти.

– Закупка автозапчастей согласно политики закупки компании. Закупка проходит строго по правилам компании, а именно: цена на запчасть находится в ценовом коридоре на день, рейтинг поставщика не ниже 4 звезд, срок поставки не более 5 рабочих дней, наличие деталей не менее 2 штук.

В рамках данного раздела было проведено изучение бизнес-процесса закупочной деятельности компании. Автоматизация закупочных процессов позволяет повысить эффективность работы специалистов, сокращать время на формирование закупочных документов, работу с отчетностью.

1.2 Изучение основных методов и принципов парсинга данных

Методы парсинга данных — это набор техник и инструментов, используемых для извлечения структурированной информации из различных источников данных, таких как веб-страницы, текстовые документы или базы данных. Парсинг данных может быть выполнен как вручную, так и с использованием автоматизированных средств, таких как библиотеки программирования, регулярные выражения или специализированные инструменты. [3]

Рассмотрим основные термины и концепции в области парсинга. Наиболее важные из них включают в себя:

HTML (Hypertext Markup Language): язык разметки, используемый для создания веб-страниц.

XML (Extensible Markup Language): универсальный язык разметки для структурирования данных.

CSS (Cascading Style Sheets): язык таблиц стилей, применяемый для оформления веб-страниц.

Существует несколько основных методов парсинга данных, каждый из них заточен под определенные задача и типы данных. Рассмотрим некоторые из них:[4]

Парсинг HTML: это процесс анализа структуры веб-страницы и извлечения нужной информации. Для этого часто используются инструменты, такие как BeautifulSoup в Python. [2]

Парсинг XML: подобно парсингу HTML, но применяется к документам XML. Этот метод широко используется в обработке структурированных данных.

Парсинг с использованием регулярных выражений: в некоторых случаях, когда данные не структурированы или сложно представлены в виде HTML/XML, можно применить регулярные выражения для извлечения информации. [5]

Рассмотрим примеры использования парсинга данных: [6]

Сбор данных с веб-сайтов: автоматический сбор информации о продуктах, ценах, новостях и т.д.

Анализ лог-файлов: извлечение и анализ данных о посещениях веб-сайта, запросах к серверу и т.д.

Обработка структурированных данных: извлечение информации из XML-файлов, JSON-объектов и других форматов.

Так же стоит затронуть алгоритмы парсинга данных: [7]

Принцип согласованности: Парсер должен следовать грамматике языка и проверять, соответствуют ли разобранные элементы правилам грамматики. Если элемент не соответствует грамматике, возникает ошибка.

Принцип локальности: Парсер должен анализировать только ограниченный контекст входного текста в определенный момент времени. Он не должен зависеть от всего входного потока сразу, чтобы обеспечить эффективность и масштабируемость.

Принцип предсказуемости: Парсер должен иметь предсказуемое поведение при разборе входного текста. Это означает, что для каждого элемента входного текста должно быть определено единственное правило разбора, которое должно быть применено.

Принцип полноты: Парсер должен способен разобрать все корректные входные конструкции, соответствующие грамматике языка. Он не должен пропускать никакие допустимые элементы или структуры.

Принцип эффективности: Парсер должен быть эффективным с точки зрения времени и памяти. Он должен разбирать текст с приемлемой производительностью, особенно для больших или сложных входных данных.

Принцип точности: Парсер должен обеспечивать точность разбора, то есть не допускать ошибок или ложных срабатываний при анализе входного текста.

1.3 Анализ бизнес-процесса закупки автозапчастей

В качестве методологии моделирования бизнес-процесса выбрана нотация IDEF0.

В методологии IDEF0 реализованы возможности декомпозиции бизнес-процессов во время анализа прикладных задач, что позволяет выявить потенциальные области оптимизации. Это может привести к сокращению трудозатрат, уменьшению затрат и избеганию дублирования функций. Также возможно построение моделей с различных точек зрения, включая моделирование процессов реинжиниринга.

Была разработана диаграмма бизнес-процессов для закупки автозапчастей в компании. Входная информация, необходимая для этого процесса, включает следующее:

– документы, содержащие информацию о заявках на покупку конкретных запчастей, на текущий день;

– физические товары, доступные для приобретения компанией самостоятельно;

– долгосрочные соглашения между компанией «Торговый дом Токус» и контрагентами о поставках автозапчастей.

Эта входная информация представлена как на бумажных, так и на электронных носителях в неструктурированной форме.

Контекстная диаграмма процесса ведения закупки автозапчастей представлена на рисунке 2.

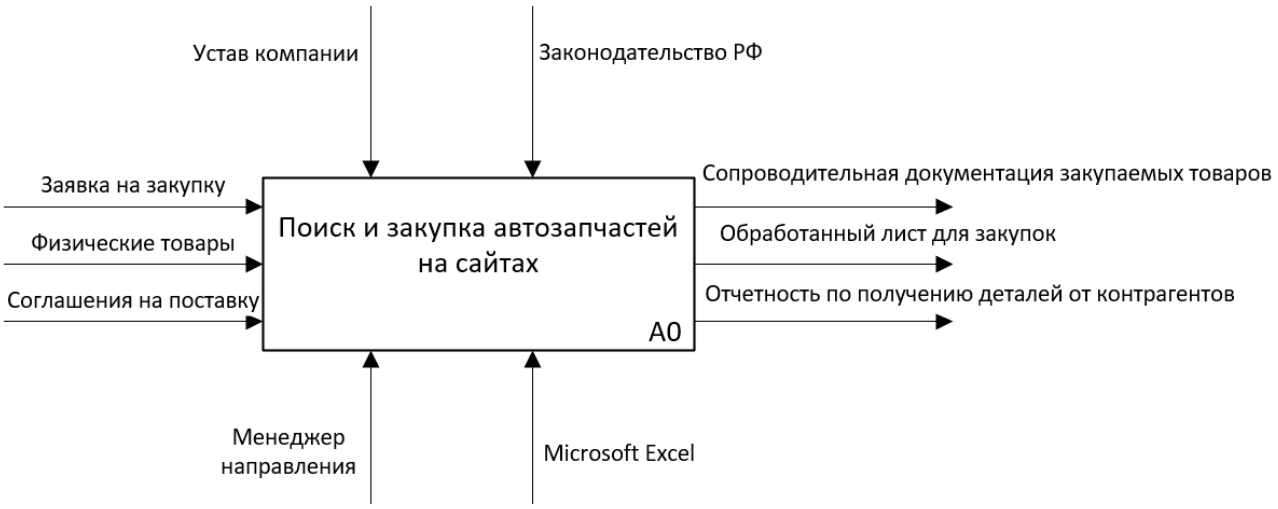

Рисунок 2 – Контекстная диаграмма поиска и закупки автозапчастей на сайтах

Структура выходной информации в бизнес-процессе поиска автозапчастей на сайтах включает:

- сопроводительная документация закупаемых товаров;
- обработанный лист для закупок;
- отчетность по получению деталей от контрагентов.

В качестве исполнителей выступают менеджеры направления отдела закупок.

Вся деятельность декларируется уставом компании «Торговый дом Токус» и Законодательством РФ «О защите прав потребителя».

Далее проведена детализация бизнес-процессов закупок автозапчастей.

Бизнес-процессы закупки автозапчастей включают в себя: получение заявки от покупателя и ежедневного отчетного документы от руководства, формирование общий закупки на день, подбор критериев для поиска согласно заявке покупателя и оптимизация полей документа под заявку, поиск автозапчастей на сайтах, высчитывание потенциальной моржи и прибыли, формирование отчетности о закупке на день и получение отчетов об реализации и выставление деталей на продажу.

Детализация процесса поиска и закупки автозапчастей приведена на

рисунке 3.

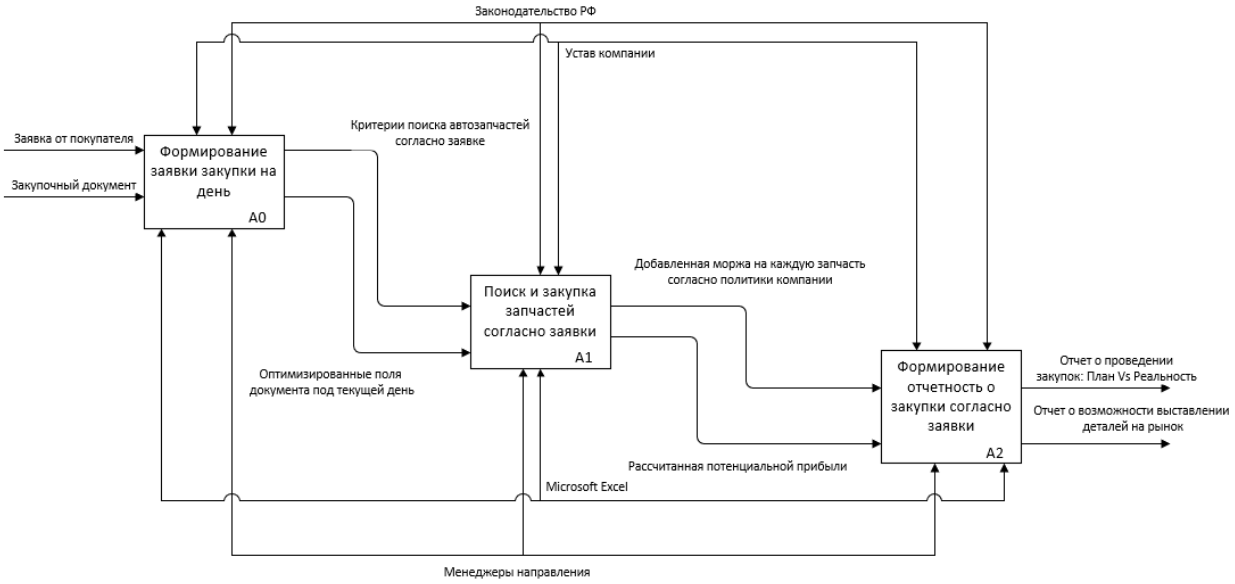

Рисунок 3 – Детализация процесса поиска и закупки автозапчастей согласно заявке

На рисунке 4 приведена декомпозиции процесса формирование заявки закупки на день.

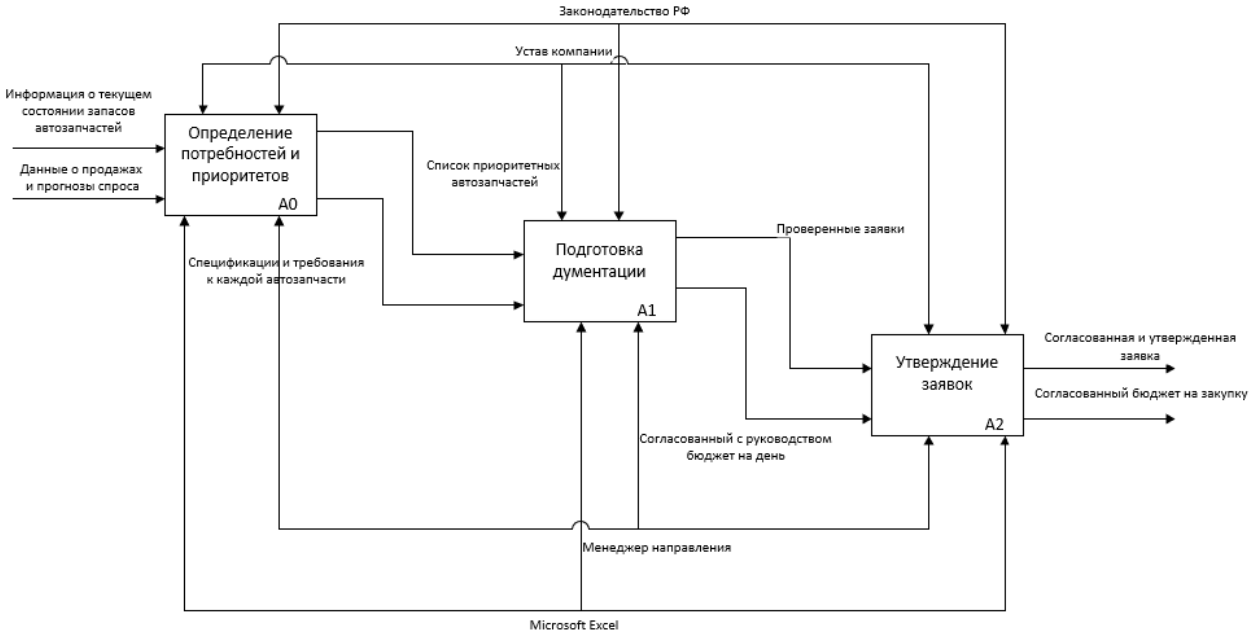

Рисунок 4 –Декомпозиции процесса формирование заявки закупки на день

Исходя из данных на рисунке 4, процесс формировании заявки закупки на день включает в себя: определение потребностей и приоритетов, подготовка документации, утверждение заявок на день.

Диаграмма декомпозиции процесса поиска и закупки запчастей согласно заявке приведена на рисунке 5.

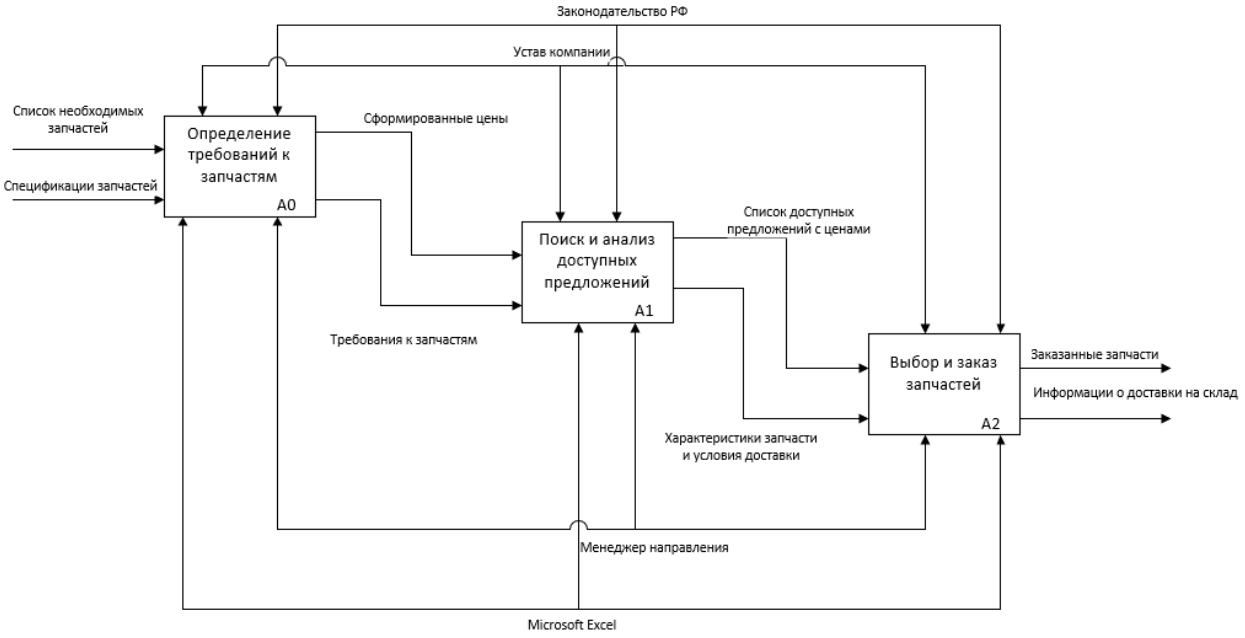

Рисунок 5 – Диаграмма декомпозиции процесса поиска и закупки запчастей согласно заявке

Процесс поиска и закупки запчастей, согласно заявке, предполагает: определение требований к автозапчастям, сравнение с ценами, утвержденными компанией на закупку, уточнение требований к закупке, поиск и анализ доступных предложений на рынке, выбор и заказ автозапчастей.

На рисунке 6 показана диаграмма процесса формирования отчётности о закупке согласно заявке.

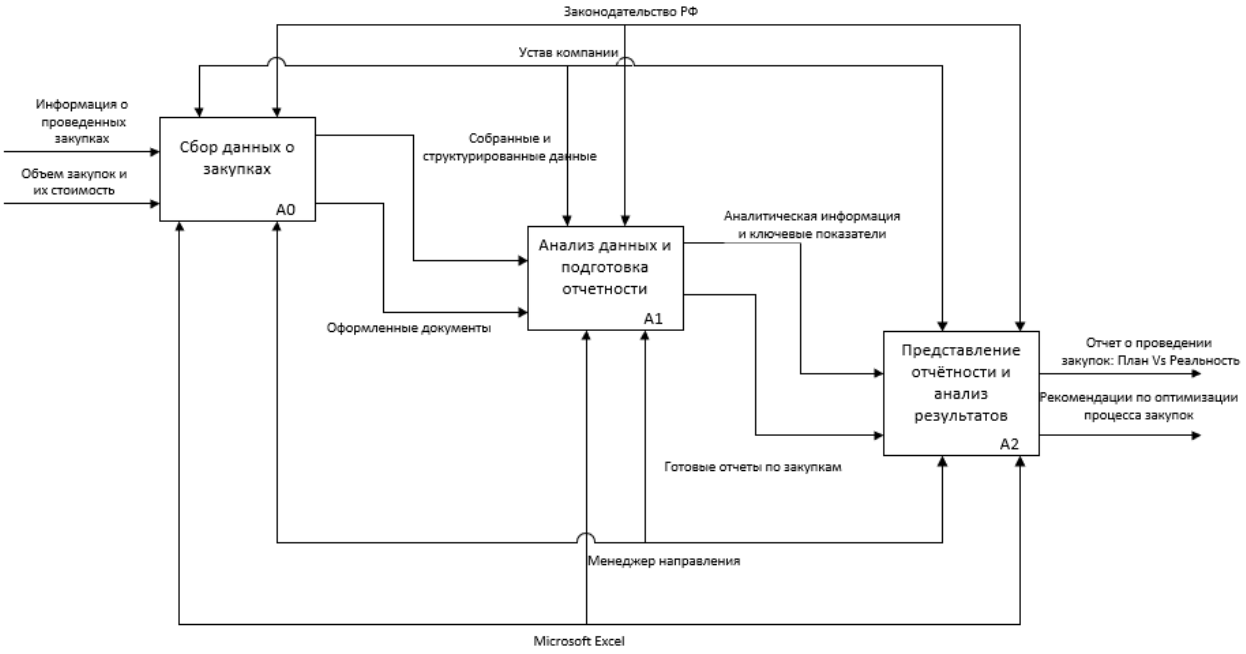

Рисунок 6 – Декомпозиция процесса формирования отчётности по закупке.

Как видно из диаграммы на рисунке 6, процесс состоит из 3 этапов: сбор данных о закупках (информация о проведенных закупках и объем закупок и их стоимость), анализ и подготовка отчётности (собранные и структурированные данные и оформление отчетных документов), предоставление отчётности и анализ результатов (готовые отчеты по закупкам и аналитическая информация).

1.4 Обоснование необходимости автоматизированного варианта решения и формирование требований системы закупки автозапчастей

По итогам проведённого анализа бизнес-процессов закупки автозапчастей в ООО «Торговый дом Токус» было показано, что в существующем процессе закупки автозапчастей выявлены следующие проблемы:

– отсутствие инструмента, позволяющего автоматически осуществлять поиск автозапчастей на сайтах;

– отсутствие инструмента, позволяющего автоматически сохранять

данные после поиска в подготовленные документы;

– множество ручных операций увеличивает вероятность ошибок в документах и недостоверности данных.

Таким образом, в настоящее время закупочная деятельность завязана на дополнительных действиях таких как: ручной поиск автозапчастей и формирование закупочных листов в ручную, а именно документ в Microsoft Excel где необходимо оптимизировать все поля документа, согласно политики компании (наличие не менее 2 шт., доставка не более 5 дней), отбор этих значений каждый раз делается вручную, что значительно увеличивает время, потраченное специалистами на однотипную работу каждый день, предполагает повышенный риск выгорания на рабочем месте и «замыливание глаза», что может привести к критическим ошибкам в процессе закупок и компания понесет необязательные затраты или заведение судебных дел о некачественно оказанной услуге. Внедрение информационной системы автоматической осуществляющей поиск автозапчастей на сайтах согласно заявки позволит автоматизировать описанные выше процессы, снизить их трудоемкость, убрать возможность непредвиденных затрат и открытие судебных дел на компанию. Перечень задач, подлежащих автоматизации, включает:

– автоматизация поиска автозапчастей;

– автоматическая редакции закупочного листа на день;

– автоматическое формирование документа «План закупок» согласно заявке на день.

Таким образом, внедрение скрипта для автоматизированного парсинга данных с сайтов по продаже автозапчастей, в котором реализованы указанные задачи, обеспечит возможности повышения эффективности отдела закупок исследуемой организации.

1.5 Анализ методов и инструментов для автоматизированного парсинга данных с сайтов по продаже автозапчастей

В соответствии с поставленными задачами автоматизации определен перечень программных решений необходимый для реализации:

– учет специфики бизнес-процессов конкретной компании;

– простота в освоении;

– возможность приведение к единому стилю заявки на закупку; а именно замена «жаргонных» названий компаний производителей на правильные названия, которые можно найти на сайтах по продаже автозапчастей;

Производитель	Интернет (Emex)	Интернет (Armtek)	Везде
DAEJIN	DJE (Dae Jin)		
CHUNWOO	Cheon Woo Industry		
JD	Just Drive		
REDSKIN	RedSkin		
MBS	Hyundai / KIA	MOBIS	
SANGSIN			HI-Q;Sangsin Brake;Sangsin;SANGSIN
RHEE JIN	Vict Rhee Jin		
VALEO			VALEO PHC;VALEO;Valeo, PHC;
ТОСОЛ-СИНТЕЗ			TOSOL;TS
ORIGINAL	Hyundai / KIA	MOBIS	
SPRINGER	ЧМЗ		
CARDEX	CAR-DEX		
PARTS MALL	Parts-Mall		
FYC	FYC AUTOPARTS		
S-OIL	S-OIL SEVEN		
DCC	DCEC		

Рисунок 7 – Product файл, согласно которому происходит авторедактирование заявки на день

– учет критериев поиска согласно заявке, а именно срок поставки, доступное количество деталей для покупки, рейтинг поставщика;

– формирование отчетов «План закупок» по таким полям как: Компания производитель, артикул, наименование детали, рейтинг поставщика, сколько будет идти, количество, цена детали, наименование

поставщика, последние обновление информации о автозапчасти;

	1	2	3	4	5	6	7	8	9
1	Компания	Артикул	Наименов	Рейтинг п	Сколько б	Количеств	Цена дета	Наименов	Последне
2	Hyundai /	58400 4F0	Тормозно	4.7	2 дн.	6	13 061 ₽	QXSS	19 апр 2024
3	Hyundai /	21320 472	Уплотнит	5.0	2 дн.	58	399 ₽	ZOOA	19 апр 2024
4	Hyundai /	21320 472	Уплотнит	4.8	2 дн.	23	405 ₽	QRNT	19 апр 2024
5	Hyundai /	21320 472	Уплотнит	4.5	2 дн.	60	430 ₽	IISW	19 апр 2024
6	Hyundai /	21320 472	Уплотнит	4.6	1 дн.	58	452 ₽	TLQA	21 апр 2024
7	Hyundai /	21320 472	Уплотнит	4.8	2 дн.	13	459 ₽	JOKQ	21 апр 2024
8	Hyundai /	21320 472	Уплотнит	1.5	2 дн.	23	459 ₽	TLJI	19 апр 2024
9	Hyundai /	21320 472	Уплотнит	4.5	2 дн.	29	464 ₽	MZHC	21 апр 2024
10	Hyundai /	21320 472	Уплотнит	4.4	2 дн.	99	464 ₽	ZZZZ	19 апр 2024
11	Hyundai /	21320 472	Уплотнит	4.8	1 дн.	14	466 ₽	YFVS	21 апр 2024
12	Hyundai /	21320 472	Уплотнит	1.0	2 дн.	3	484 ₽	LHRK	20 апр 2024
13	Hyundai /	21320 472	Уплотнит	1.0	3 дн.	3	484 ₽	LHRA	20 апр 2024
14	Hyundai /	21320 472	Уплотнит	3.8	2 дн.	48	486 ₽	SVYA	21 апр 2024
15	Hyundai /	21320 472	Уплотнит	1.4	2 дн.	48	486 ₽	SINL	21 апр 2024
16	Hyundai /	21320 472	Уплотнит	1.6	2 дн.	10	492 ₽	KCCB	19 апр 2024
17	Hyundai /	21320 472	Уплотнит	2.6	3 дн.	99	499 ₽	FRZA	22 апр 2024
18	Hyundai /	21320 472	Уплотнит	1.0	4 дн.	84	508 ₽	KMGB	22 апр 2024
19	Hyundai /	21320 472	Уплотнит	1.9	3 дн.	99	511 ₽	VKVX	22 апр 2024
20	Hyundai /	21320 472	Уплотнит	3.8	4 дн.	7	513 ₽	PISL	19 апр 2024

Рисунок 8 – Формирование отчетного документа «План закупок»

– режим функционирования - круглосуточно (365/24/7);
– количество одновременно работающих пользователей - не менее 3;
– система должна иметь возможность расширения.

Эти параметры сопоставления помогут проанализировать имеющиеся решения, чтобы определить, нужно ли разрабатывать собственную систему, приобретать готовое решение или приобретать готовое с последующей модификацией под требования текущей компании.

1.6 Обзор технологий и языков программирования для разработки скрипта парсинга данных

Согласно установленным параметрам, было проведено сопоставление функциональных возможностей различных программных продуктов, доступных на рынке и способных автоматизировать процессы закупки. Результаты сравнения представлены в таблице 1.[11]

Таблица 1 - Сравнение возможностей программных решений

Показатель	1С:Управление торговлей	1С:ERP Управление предприятием
Интеграция с поставщиками	Имеет возможность интеграции с базами данных поставщиков и электронными торговыми площадками.	Предоставляет средства интеграции с внешними системами поставщиков и электронными торговыми площадками.
Мониторинг цен и наличия	Предоставляет средства мониторинга цен и наличия автозапчастей у различных поставщиков.	Обеспечивает мониторинг цен и наличия у различных поставщиков, а также на складах компании.
Аналитика и отчетность	Включает инструменты для анализа закупочных операций, составления отчетов о закупках и оценки эффективности закупочных процессов.	Предоставляет расширенные возможности аналитики и отчетности, включая финансовый анализ и прогнозирование спроса.
Учет специфики бизнес-процессов	Может быть настроен под конкретные бизнес-процессы компании, включая специфику закупки автозапчастей.	Предоставляет гибкие инструменты настройки и конфигурирования, что позволяет учесть особенности бизнес-процессов.
Простота в освоении	Имеет простой и интуитивно понятный интерфейс, что облегчает процесс освоения для новых пользователей.	Предоставляет гибкий интерфейс и обучение пользователям, что помогает быстрее освоить систему.
Автоматизация процесса заказа	Позволяет автоматизировать процесс формирования и отслеживания заказов автозапчастей,	Обеспечивает автоматизацию всего цикла закупки, включая управление заказами, поставками и оплатой.

По итогам рассмотрения функционала наиболее распространенных

систем закупкой было показано, что рассмотренные решения позволяют автоматизировать типовые процессы закупки автозапчастей, но требуют доработки для адаптации к работе в условиях исследуемой компании, а именно:

- возможность приведение к единому стилю заявки на закупку;
- учет критериев поиска согласно заявке;
- формирование отчетов в Microsoft Excel;
- режим функционирования - круглосуточно (365/24/7).

Также согласно установленным требованиям, было проведено сопоставление языков программирования, подходящих для разработки скрипта парсинга данных и способных автоматизировать процессы закупки. Результаты сравнения представлены в таблице 2.[12]

Таблица 2 - Функциональные возможности языков программирования

Язык программирования	Описание языков программирования	Преимущества	Недостатки
Python	Python - высокоуровневый язык программирования с простым и понятным синтаксисом. Он широко используется в различных областях, включая веб-разработку, научные вычисления, анализ данных и автоматизацию процессов. Python также известен своими богатыми библиотеками, которые облегчают разработку парсеров и скриптов автоматизации.	Простой в изучении и использовании Обширная библиотека для парсинга данных (например, BeautifulSoup, Scrapy) Много инструментов для автоматизации, таких как Selenium Широко используется в сфере анализа данных и автоматизации процессов	Иногда медленнее в сравнении с компилируемыми языками

Продолжение таблицы 2

JavaScript	JavaScript - язык программирования, который применяется преимущественно в веб-разработке. Он обладает динамической типизацией и может быть использован как для написания клиентского, так и серверного (Node.js) кода. Благодаря своей популярности, существует множество библиотек и фреймворков, которые облегчают разработку парсеров и автоматизацию веб-приложений.	Широкое распространение и поддержка веб-технологий Возможность парсинга данных с помощью библиотеки Cheerio (для Node.js) Автоматизация веб-приложений с помощью Puppeteer	Меньше библиотек и инструментов для парсинга данных по сравнению с Python Менее подходит для парсинга неструктурированных данных
Ruby	JavaScript - язык программирования, который применяется преимущественно в веб-разработке. Он обладает динамической типизацией и может быть использован как для написания клиентского, так и серверного (Node.js) кода. Благодаря своей популярности, существует множество библиотек и фреймворков, которые облегчают разработку парсеров и автоматизацию веб-приложений.	Простой синтаксис и читаемость кода Библиотека Nokogiri для парсинга HTML и XML Фреймворк Watir для автоматизации браузеров	Меньшее сообщество и меньше поддерживаемых библиотек по сравнению с Python Не самый популярный выбор для разработки парсеров и скриптов автоматизации

Продолжение таблицы 2

Java	Java - объектно-ориентированный язык программирования, широко применяемый в корпоративной среде. Он известен своей платформонезависимостью и высокой производительностью. Java используется для написания различных видов приложений, включая веб-приложения, мобильные приложения и скрипты автоматизации.	Широко используется в корпоративной среде Библиотеки Jsoup для парсинга HTML и XML Инструмент Selenium для автоматизации браузеров	Более громоздкий и сложный в использовании по сравнению с другими языками Требует больше кода для достижения тех же результатов

В качестве языка программирования был выбран Python, поскольку его набор библиотек максимально подходит под задачи автоматизации.

1.7 Определение функциональных и нефункциональных требований к скрипту парсинга данных

Функциональные требования определяют то, что система должна делать, включая функции и операции, которые должны быть предоставлены пользователю.[13]

– парсинг данных: Скрипт должен быть способен извлекать данные из определенных источников, таких как веб-страницы, файлы JSON, XML и т. д;

– фильтрация и обработка данных: Скрипт имеет способности фильтровать и обрабатывать данные в соответствии с определенными критериями;

– хранение данных: скрипт должен быть способен сохранять извлеченные данные в определенном формате xls на сервере;

– управление ошибками и исключениями: Скрипт должен быть устойчив к возможным ошибкам, таким как изменение структуры данных на источнике.

Нефункциональные требования определяют качественные атрибуты, которые должны присутствовать в системе.

– производительность: Скрипт должен быть способен обрабатывать большие объемы данных эффективно и быстро;

– надежность: Скрипт должен быть стабильным и не подверженным сбоям при обработке различных типов данных;

– масштабируемость: Скрипт должен быть гибким и легко масштабируемым для обработки изменяющегося объема данных;

– сопровождаемость: Скрипт должен быть хорошо документирован и легко поддерживаемым для разработчиков и администраторов;

– удобство использования: Интерфейс скрипта должен быть интуитивно понятным и легким в использовании для конечных пользователей.

Алгоритм работы скрипта парсинга данных с сайтов автозапчастей:

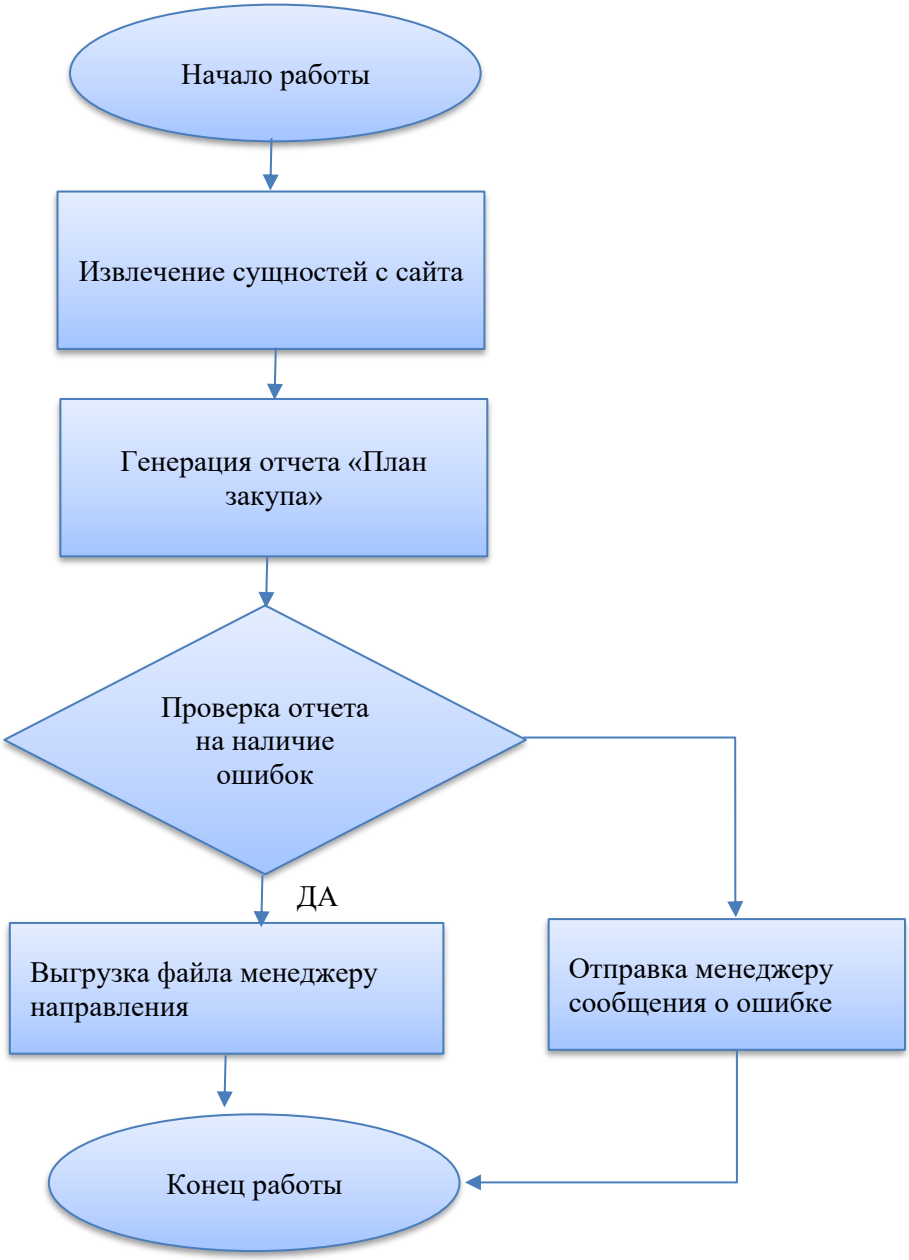

Рисунок 9 – Процесс работы скрипта парсинга данных с сайтов автозапчастей

Разработка парсинга данных:[14]

– реализация парсера для извлечения данных из сайтов по продаже автозапчастей;

– интеграция с библиотеками для работы с различными форматами данных;

– фильтрация и обработка данных;

– разработка механизмов фильтрации и обработки данных в соответствии с заданными критериями.

Организация хранения данных:

– создание модуля для сохранения обработанных данных в формате XLS на сервере;

– обеспечение безопасности данных и механизмов управления доступом к хранилищу.

Управление ошибками и исключениями:

– разработка обработчиков ошибок для обеспечения устойчивости скрипта к возможным сбоям;

– мониторинг и журналирование ошибок для последующего анализа и улучшения системы.

Настройка производительности:

– оптимизация алгоритмов обработки данных для эффективного использования ресурсов;

– параллельная обработка данных для ускорения процесса.

Тестирование на надежность скрипта:

– тщательное тестирование скрипта на различных типах данных и условиях работы;

– внедрение механизмов резервного копирования данных для предотвращения потери информации;

– масштабируемость скрипта;

– архитектурное решение, позволяющее легко масштабировать скрипт для обработки больших объемов данных.

Документирование работы скрипта:

– документирование кода и процессов работы системы для удобства поддержки и развития;

– обеспечение чистоты кода и модульности для упрощения

внесения изменений.

Настройка пользовательского интерфейса:

– создание пользовательского интерфейса с интуитивно понятными командами для конечных пользователей;

– предоставление документации и руководств пользователя для быстрого освоения системы.

1.8 Функциональное моделирование бизнес-процесса «Применение автоматизированного парсинга данных с сайтов по продаже автозапчастей»

На рисунках 10-11 приведены диаграммы бизнес-процессов закупки автозапчастей после внедрения информационной системы. Автоматизированный парсинг данных далее в работе будет называться – АПД «SIMON» Согласно рисунку 1.

Рисунок 10 – Контекстная диаграмма поиска и закупки автозапчастей на сайтах «Как должно быть»

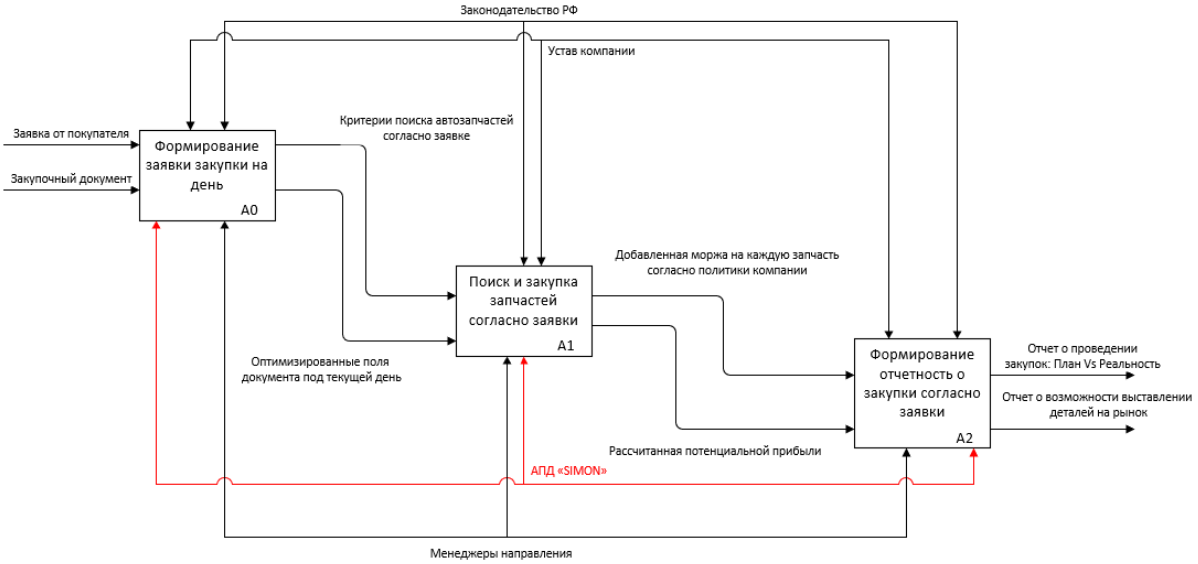

Рисунок 11 – Диаграмма декомпозиции процесса поиска и закупки автозапчастей «Как должно быть»

Таким образом, изменения в бизнес-процессах закупки автозапчастей включают внедрение информационной системы парсинга сайтов автозапчастей, обеспечивающие возможности автоматизации каждого из участков работы, включающего работу заявкой, поступившей в работу на день.

Выводы по первой главе:

Проанализирована организационная структура компании и процессы поиска и закупки автозапчастей. Было выполнено моделирование функциональных аспектов процесса поиска и закупки автозапчастей на сайтах до и после применения автоматизированного парсинга данных. Выявлены функциональные и нефункциональные требования к парсингу данных. Выбрана технология для реализации скрипта, а также язык программирования. В результате анализа существующих систем автоматизации закупки автозапчастей было выявлено, что их функционал не полностью соответствует потребностям и особенностям рассматриваемой компании. В связи с этим было принято решение разработать собственное программное решение.

Глава 2 Логическое проектирование скрипта для автоматизированного парсинга данных с сайтов по продаже автозапчастей

2.1 Анализ потенциальных источников данных, включая сайты по продаже автозапчастей

Анализ потенциальных источников данных играет важную роль в разработке скрипта для автоматизированного парсинга. В контексте данного исследования, внимание уделяется сайтам, специализирующимся на продаже автозапчастей, которые представляют собой богатый источник информации о товарах, их ценах, наличии и других параметрах.

Сайты по продаже автозапчастей имеют разные характеристики и особенности включая ассортимент, способ предоставления данных и условия при которых эти данные можно получить. Сайты могут представлять как крупную онлайн-платформу, где торгуют множество продавцов со всей страны, до специализированных магазинов, которые ориентируются на определенные категории или бренды.

В качестве возможных источников данных были рассмотрены 7 сайтов по продаже автозапчастей и только 2 соответствуют запросам поиска компании «Торговый дом Токус».

Сайт https://www.zzap.ru/ не предоставляет подробную информацию о детали, также только около 15% запчастей, указанных в заявке, можно найти и только около 5% пройдут по всем критериям.

Сайт https://www.port3.ru/ обладает обширным каталогом автозапчастей, но слишком сильно перегружен и не имеет четкой структуры для автоматизированного поиска, данные извлечь путем парсинга из него практически не представляется возможным.

Интернет магазин https://plentycar.ru/ не представляет брендов или производителей, которые были бы интересны «Торговому дому Токус» для потенциальной покупки или перепродаже.

Магазин https://www.avtoall.ru/ является одним из крупнейших рессейлеров автозапчастей в Москве, но у него на сайте отсутствует информация о поставщиках и их рейтинге, что не приемлемо по политике компании.

Интернет магазин https://autolife42.ru/ также предоставляет большой выбор запчастей, на нем можно найти практически всю информацию о необходимой детали, но зачастую цены на них имеют стоимость выше рыночной и не соответствуют коридору цен, установленному компанией.

В нашей работе будут рассмотрены два крупных сайта по продаже автозапчастей: https://emex.ru/ и https://armtek.ru/. Это одни из самых распространённых крупных онлайн платформ, где можно купить автозапчасти любого бренда и нет ограничений от какого количества идет заказ. Также эти сайты является основными для поиска автозапчастей в исследуемой компании «Торговый дом Токус».

2.2 Идентификация основных сущностей и полей данных для сбора

Идентификация основных сущностей и полей данных является важным шагом в разработке скрипта для автоматизированного парсинга. Для достижения целей исследования и обеспечения эффективного сбора информации с сайтов по продаже автозапчастей были выделены следующие ключевые сущности и соответствующие поля данных по каждому из сайтов.[15]

Артикул товара: Эта сущность является основной поскольку по ней производится поиск на сайте. В артикуле зашито много данных таких как: модель автомобиля, для которого она, номер запчасти, производитель.

Производитель: Эта сущность имеет важное значение в поиске, так как, согласно заявке, покупатель может хотеть купить исключительно оригинальную деталь, которая будет стоить дороже, либо подобрать аналог полностью идентичный, но не брендовый, пример такого может быть деталь

на Hyundai – оригинал, а MBS -его дешевый аналог.

Название детали: Эта сущность также очень важна, так как под схожим артикулом или аналогом оригинальной детали может быть совсем другая запчасть.

Рейтинг поставщика: Эта сущность позволяет выяснить насколько надежный поставщик, сколько успешных заказов он выполнил за последние 90 дней, сколько было отказов от его запчастей.

Поставщик: Эта сущность может быть представлена на сайте в виде буквенного или цифирного формата в зашифрованном виде, это сделано специально, чтобы пользователи платформы не находили поставщиков в обход сайта, но у каждой компании, торгующей автозапчастями есть свой список с расшифровками поставщиков, чтобы быстро их индицировать и заказывать у проверенных продавцов.

Количество деталей: в рамках исследуемой компании нам очень важно учитывать сколько автозапчастей в наличии, поскольку если наличие деталей меньше 2 это может обозначать что либо детали будут долго идти, либо есть проблемы с их качеством с учетом что это не оригинальный товар.

Срок поставки: согласно уставу компании, срок поставки детали не может превышать 5 рабочих дней с момента покупки, поэтому данная сущность является критичной и важна для правильной реализации скрипта.

Цена: эта сущность позволяет выяснить находится ли автозапчасть в коридоре цен, в котором исследуемая компания может позволить себе продавать или покупать деталь.

Алгоритм для извлечения данных при автоматизированном парсинге:

Рисунок 12 - Алгоритм извлечения данных с пмощью АПД «SIMON»

Выбранные сущности помогут эффективно собирать и анализировать данные, обеспечивая точное и быстрое выполнение бизнес-задач следуя указанному выше алгоритму.

2.3 Выбор и обоснование алгоритмов и методов парсинга данных

Для успешного сбора данных с сайтов по продаже автозапчастей необходимо выбрать оптимальные алгоритмы и методы парсинга, учитывающие особенности структуры и формата представления информации на сайтах. В контексте данной работы был определен следующий алгоритм парсинга данных:[16]

Использование библиотек парсинга: для упрощения процесса парсинга данных будут применяться специализированные библиотеки в Python, такие как BeautifulSoup и Selenium. [1]

Рассмотрим подробнее каждую библиотеку.

BeautifulSoup — это библиотека для парсинга HTML и XML документов на языке Python. Она позволяет удобно и эффективно извлекать данные из веб-страниц, основываясь на их структуре. Эта библиотека поможет нам извлекать из страницы сайты необходимые нам сущности, чтобы в дальнейшем с ними работать.[25]

Selenium — это инструмент для автоматизации веб-браузера. Он позволяет вам управлять браузером программно, открывать веб-страницы, заполнять формы, кликать по элементам, считывать данные и многое другое. Поскольку наш скрипт будет работать автоматизировано для него понадобится отдельный браузер, также с помощью этой библиотеки мы будем открывать модальные окна, в которых хранится необходимая для парсинга информация, например, данные о поставщиках, сроках и датах поставки, раскрытие полного списка запчастей.

Учитывая задачи, которые были поставлены перед нами по реализации скрипта для автоматического парсинга данных с сайтов было принято решение использовать комбинацию библиотеки BeautifulSoup для парсинга HTML-кода веб-страниц и библиотеки Selenium для автоматизации веб-браузера. Вот несколько причин, обосновывающих этот выбор:

Гибкость и простота использования: BeautifulSoup предоставляет

простой и интуитивно понятный интерфейс для парсинга HTML-кода, что делает процесс извлечения данных из веб-страниц быстрым и удобным. С его помощью легко находить нужные элементы и извлекать из них информацию.

Эффективность парсинга: BeautifulSoup оптимизирован для работы с HTML-кодом и обеспечивает высокую скорость парсинга данных. Он предоставляет мощные инструменты для обработки HTML-документов и извлечения информации из них.

Автоматизация действий в браузере: Библиотека Selenium позволяет вам автоматизировать действия веб-браузера, такие как открытие страниц, клики по элементам и т. д. Это необходимо, поскольку нам понадобится входить в личный кабинет сайтов, проходить «капчу».

Кроссплатформенность и поддержка различных браузеров: Selenium поддерживает работу с различными браузерами, включая Chrome, Firefox, Safari и другие. Это обеспечивает отказоустойчивость скрипта, если по результатам тестирования какой-то из браузеров будет работать некорректно.

Рассмотрим подробнее другую библиотеку, которая поможет нам реализовать скрипт. Поскольку скрипт должен иметь какую-то оболочку было принято решение, что о будет работать внутри телегам бота, а значит нам понадобится библиотека: aiogram — это библиотека на языке Python для создания ботов Telegram. Она предоставляет удобный и мощный интерфейс для взаимодействия с API Telegram, что делает разработку ботов быстрой и эффективной. С помощью этой библиотеки наш бот будет эффективен и отзывчив, способен быстро обрабатывать запросы.

Также в скрипте будут затронуты множество библиотек связанных с Google поскольку нам необходима резервная архивация всех файлов на Google Disk согласно требованиям безопасности и сохранения целостности данных даже после уничтожения парсера.

Библиотеки позволяющие парсить Markdown-разметки, игнорировать различные регистры, некорректное написание слов. Это необходимо поскольку на сайте не всегда может быть правильно написано название детали

или ее описание, что может привести к ошибке в парсинге данных.

Алгоритм парсинга HTML-кода с использованием библиотеки BeautifulSoup следующий:

Получаем HTML-код веб-страницы, используя HTTP-запрос к целевому URL-адресу.

Импортируем библиотеку BeautifulSoup и создать объект BeautifulSoup, передавая ему полученный HTML-код в качестве входных данных.

Используя методы и функции BeautifulSoup, найти нужные элементы HTML-структуры, содержащие данные, которые необходимо извлечь. Это может быть выполнено путем поиска по тегам, классам, идентификаторам или другим атрибутам элементов.

Для каждого найденного элемента применить методы или атрибуты BeautifulSoup, чтобы извлечь необходимую информацию.

Сохранить извлеченные данные в нужном формате или передать их для дальнейшей обработки или сохранения в базу данных.[17]

Обработать ошибки и исключения, которые могут возникнуть во время парсинга, и предусмотреть соответствующие механизмы восстановления или пропуска некорректных данных.

2.4 Моделирование функциональных требований АПД «SIMON»

Для проектирования АПД «SIMON» будет использоваться диаграммы UML, а именно диаграмма последовательности и прецедентов.

Диаграмма прецедентов использования — это графическое представление взаимодействия между системой и её окружением (пользователями, другими системами и т.д.). Она используется для моделирования функциональных требований системы и идентификации основных акторов (пользователей или внешних систем), которые взаимодействуют с системой, а также основных функций (прецедентов), которые эта система выполняет для этих акторов.

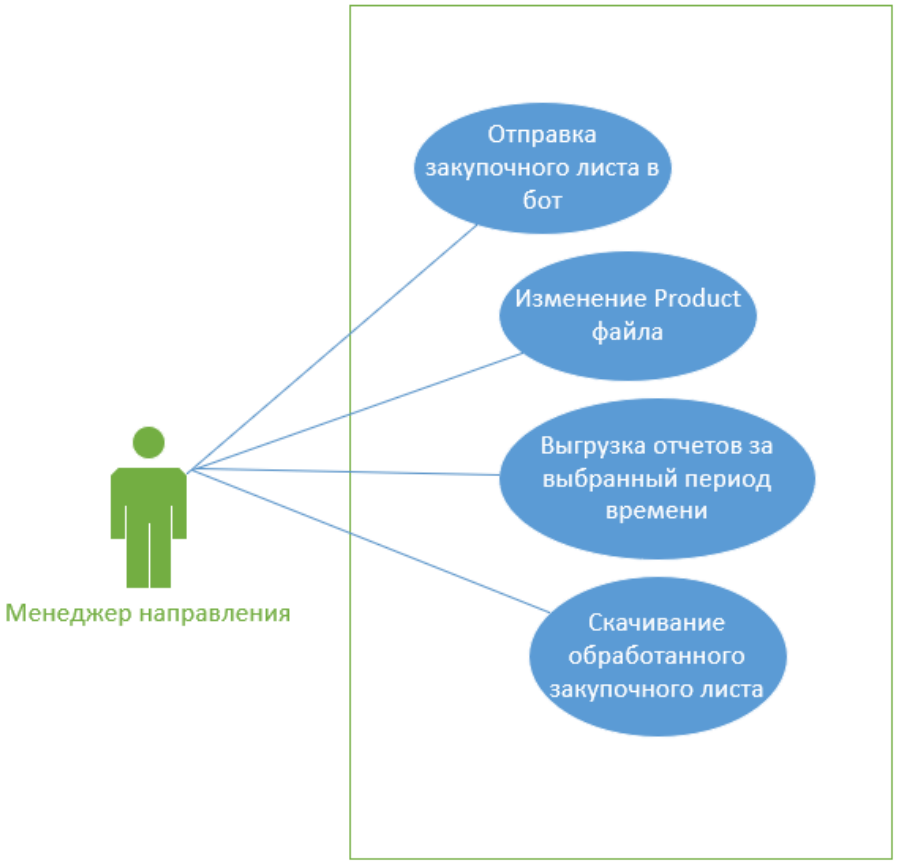

Рисунок 13 - Диаграмма прецедентов использования АПД «SIMON»

Для менеджера направления доступны следующие функции:

Отправка закупочного листа в боте: менеджер имеет доступ к боту по ссылке, через команду запуск он активирует бота и бот ждет файла для обработки, после получения файла бот начинает его обрабатывать

Изменение product файла: у менеджера есть возможно запросить этот файл и отредактировать его, либо добавить новых производителей

Выгрузка отчетов за последний день, 7 дней и за все время: эта функция позволяет производить аналитические работы по изменению цен или количества деталей тех или иных запчастей, также эта функция предоставляет возможность резервного копирования данных на компьютер менеджера.

Получение обработанного закупочного листа: менеджер получает отчет

«План закупок» после того, как АПД «SIMON» провела поиск запчастей на сайте согласно заявке

Диаграмма последовательности — это тип диаграммы в UML, который отображает взаимодействие между объектами в системе в определённой последовательности времени. Далее представлена сама диаграмма для АПД «SIMON».

Рисунок 14 - Диаграмма последовательности АПД «SIMON»

В диаграмме последовательности представлены четыре основных действия, которые выполняются, а именно:

Менеджер направления заходит в бот и загружает лист закупок на день

Бот уточняет какой конкретно сайт необходимо парсить в нашем случае emex.ru или Armtek.ru.

Пользователь выбирает сайт

Бот запускает цикл парсинга данных

BPMN— это графический язык для моделирования бизнес-процессов.

Он используется для описания бизнес-процессов и их взаимодействия между различными участниками или системами в рамках организации.

Визуализация процессов: BPMN позволяет представить бизнес-процессы в понятной графической форме. Это делает процессы более доступными для понимания и анализа как бизнес-пользователями, так и IT-специалистами. Для чего может быть полезна BPMN:

Оптимизация процессов: Анализ BPMN-диаграмм позволяет выявить узкие места и неэффективные этапы в бизнес-процессах. Это помогает бизнесу оптимизировать свою деятельность и повышать производительность.

Согласование и коммуникация: Использование единого языка для описания процессов помогает улучшить коммуникацию между различными участниками бизнеса, такими как менеджеры, аналитики и разработчики.

Автоматизация: BPMN может использоваться как основа для автоматизации бизнес-процессов с помощью специализированных систем управления бизнес-процессами (BPMS). После моделирования процессов в BPMN их можно легко перевести в исполняемый код.

Анализ и управление рисками: путём моделирования процессов в BPMN можно выявить потенциальные риски и уязвимости в бизнес-процессах, что помогает в разработке стратегий управления рисками.

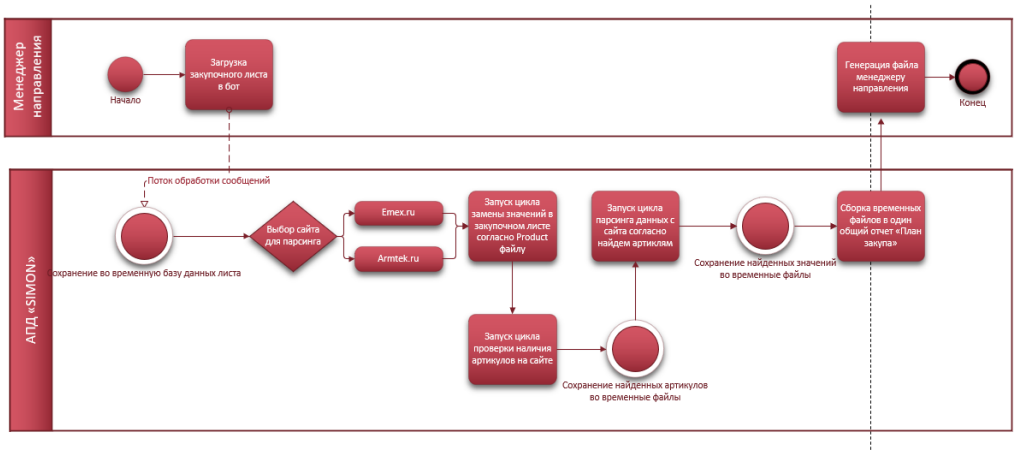

Рисунок 15 - Модель бизнес-процессов BPMN в компании ООО «Торговый дом Токус» после внедрения АПД «SIMON»

Выводы по второй главе:

Были проанализированы потенциальные источники данных для парсинга автозапчастей, определены два основных сайта, которые удовлетворяют всем критериям исследуемой компании. Выбраны сущности для парсинга данных, описаны их назначения и прописан алгоритм извлечения их с сайта. Рассмотрены библиотеки позволяющие реализовать парсинг данных и выбраны подходящие согласно задачам. Проведено функциональное моделирование требований к АПД «SIMON». Построена диаграмма прецедентов использования, где рассмотрены взаимодействие менеджера направления с АПД «SIMON», разработана диаграмма последовательности, а также проведено моделирование бизнес-процессов по средствам графического представления BPMN.

Глава 3 Физическое проектирование АПД «SIMON»

3.1 Разработка архитектуры клиент-серверной системы

Проектирование клиент-серверной системы начинается с настройки сервера, установки на него всех необходимых утилит, далее запрос с сервера уходит в телеграм бот, который в свою очередь запускает парсер, а он в свою очередь отправляет в базу данных все свои файлы в базу данных.

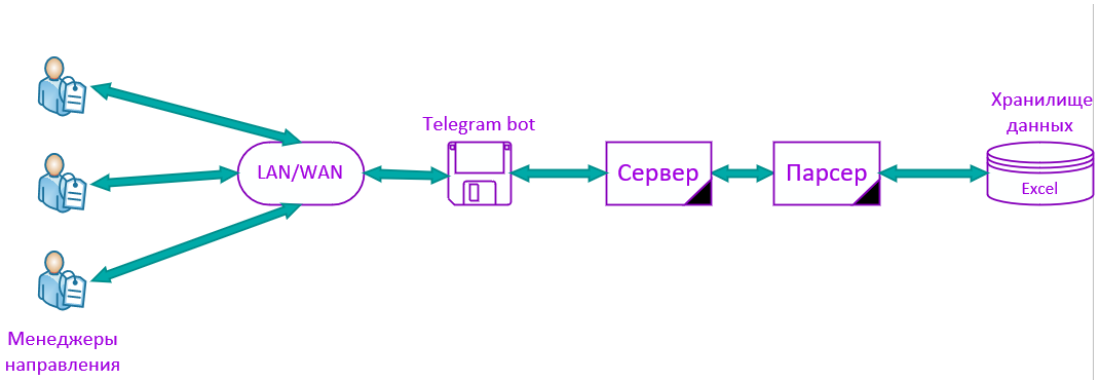

Рисунок 16 – Архитектура клиент-серверной системы АПД «SIMON»

В качестве сервера используется компьютер на Windows 10.

Как только телеграмм бот получает на вход файл происходит сохранение его во временную память по средствам функции chunked(),она же и разбивает файл на равное количество значений кратное количеству браузеров для обработки. В первом цикле проверки артиклей на сайтах. В chunked хранится по 10 значений, во втором цикле, по 5 значений, это сделано дня минимизации ошибок, если один из временных списков будут поврежден или записан с ошибкой, то общий отчет «План закупа» будет сформирован без ошибок. [18]

Как только цикл отрабатывается у нас появляются временные файлы Excel, которые хранят в себе данные о парсинге, они сохраняются в базу данных, она выражена архитектурой папок. Первая запись происходит в папку .temp в которой хранятся все временные файлы. После того как все chunked

были отработаны формируется отчет «План закупа», который в свою очередь сохраняется в Sites разделенную на подпапки с сайтами с которыми работает АПД.

В случае если менеджер направления запросил у бота выгрузку отчетов за определенный период времени у нас создается zip архив с выбранными отчетами, и он автоматически отправляется на Google Disk в качестве резервного копирования данных, у нас это выполняет функцию защиты данных от непредвиденных обстоятельств, таких как потери соединения с сервером. Также в коде присутствует функци автоматической архивации файлов и отправки его на Google Disk. Эта функция запускается через автоматически каждую пятницу в 00:00. [19]

3.2 Разработка и отладка скрипта

Разработка начинается с написание парсера. Парсер для каждого из двух сайтов состоит из 3 частей таких как: файл с авторизаций на сайте, файл с циклом проверки наличия на сайте артикулов автозапчастей и сам основной парсер, который производит забор сущностей с сайта. [20]

Рассмотрим подробнее каждый из файлов на примере сайта emex.ru.[8]

```
if login_button:
    login_button[0].click()

    browser.implicitly_wait(20)
    time.sleep(3)

    browser.find_element(By.XPATH, """//*[@id="signInLoginInput"]""").clear()
    browser.find_element(By.XPATH,
"""//*[@id="signInPasswordInput"]""").clear()

    time.sleep(0.5)

    browser.find_element(By.XPATH,
"""//*[@id="signInLoginInput"]""").send_keys(emex_login)
    browser.find_element(By.XPATH,
"""//*[@id="signInPasswordInput"]""").send_keys(emex_password)
    browser.find_element(By.XPATH, '//button[@data-
testid="LoginPopup:button:signInLoginButton"]').click(
```

Рисунок 17 – Алгоритм поиска входа на сайт

В отрывке кода выше происходит авторизации с помощью функции поиска кода элемента browser.find_element находится окно авторизации в личном кабинете, и из файла confic вставляются логин и пароль, чтобы зайти на сайт и начать парсинг.[9]

```
#          Решение          через          requests          и          парсинг          html
#                                          https://emex.ru/f?detailNum=802624&packet=-1
browser.get(f"{emex_url}f?detailNum={article}&packet=-1")

time_sleep(60)

html_page                                =                                browser.page_source
html_element_1                                    =                                      ""
html_element_2                                    =                                      ""

soup                =                BeautifulSoup(html_page,                "html.parser")
main_frame    =    soup.find("div",    {"class":    "body-content",    "id":    "body-content"})
ex1          =          main_frame.find("div",          {"class":          "brand-selector"})

if                                                                                    ex1:
    def_ex1(soup,                            article,                            result_list)
else:
    if   button_table   :=   browser.find_elements(By.CSS_SELECTOR,   "span.dashed-link:nth-
child(2)"):
        button_table[0].click()
        if   sel_element_2_table   :=   browser.find_elements(By.CSS_SELECTOR,   """.brand-
selector"""):
            html_element_2        =        sel_element_2_table[0].get_attribute("innerHTML")

    soup_3            =            BeautifulSoup(html_element_2,            "html.parser")
    ex3           =           soup_3.find("div",           {"class":           "brand-list"})
    if                                                                                ex3:
        def_ex3(soup_3,                          article,                          result_list)
    else:
        if            sel_element_1           :=           browser.find_elements(By.XPATH,

"""/html/body/div[2]/div[3]/div/div[2]/div[3]"""):
            html_element_1          =          sel_element_1[0].get_attribute("innerHTML")

        soup_2           =           BeautifulSoup(html_element_1,           "html.parser")
        ex2     =     (main_frame.find("div",     {"class":     "search-rezult     "})     or
              soup_2.find("div",               {"class":               "search-rezult"}))
        if                                                                            ex2:
            def_ex2(soup_2, article, result_list)
```

Рисунок 18 - Решение через requests и парсинг html

Выше представлен код, где запускается цикл проверки наличия артиклей на выбранном сайте. Для начала из закупочного листа извлекается колонка 5. Далее они записываются во временную память и оттуда извлекаются по очереди для проверки, после проверки формируется файл txt в котором они будут хранится и вставляться при основном парсинге данных.

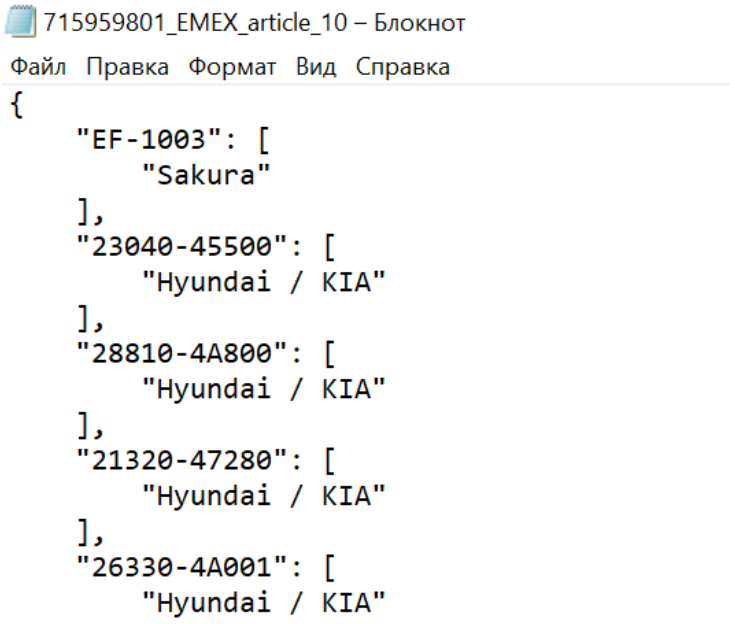

Рисунок 19 - Файл с найденными на сайте артикулами

```
if check_find_element(
    browser,
    if check_find_element(
        browser,
        f'//div[@data-bind="foreach: Brands"]//div[contains(text(),
"{company}")]/..',):
        browser.find_element(
            By.XPATH,
Brands"]//div[(contains(text(),"{company}"))]/..',
        ).click()
        time_sleep(0.5)
        browser.execute_script(
            '$(".filters-section .label:not(:contains(Искомый артикул))").click();')
  time_sleep(0.5)
        browser.execute_script(
            '$(".filtered-details .subpanel-details > div:not(:visible)").remove()')
        time_sleep(0.5)
        if check_find_element(
            browser,
            '//div[contains(@data-bind,"visible: isFooterVisible") and
not(contains(@style,"display: none;"))]//span[@data-bind="click: expand"]',):
```

Рисунок 20 – Код для проверки наличия артиклей на сайте

В данном Python файле, происходит по порядку извлечение всех полей по одному артикулу, и записывается в файл Excel, также отработаны все исключения, где нет значений, или какое-то неверное значение записано. Также отработаны все условия для потенциальной покупки запчасти такие как количество деталей и их срок поставки.[10]

```
if not c or c in [
    "0",
    "0.0",
    "0,0",
    "1",
    "1.0",
    "1,1",
    "2",
    "2.0",
    "2,0",
    "Под заказ",
```

Рисунок 21 – Код обработки исключения по количеству

Особое внимание стоит уделить отработки исключений при помощи переменной «c» мы проверяем соответствие найденных значений условиям записи в файл.

В процессе отладки была выявлена одна особенность сайта emex.ru, которая приводила к ошибке в отчетах «План закупа» и записывала даты которые не соответствуют политики компании, поэтому структура отбора дат была сделана более жестко и добавлен следующих код:

```
if dn not in [
    "1 дн.",
    "2 дн.",
    "3 дн.",
    "4 дн.",
    "5 дн.",
```

Рисунок 22 – Код обработки исключения по дате поставки

Данное условие ставит четкие рамки записи в отчет.

Далее рассмотрим архитектуру самого телеграмм бота. Который является физической оболочкой АПД, чтобы менеджеры направления могли взаимодействовать со скриптом.

Основной файл python называется parsing_handler.

Как только пользователь заходит в бот и активирует его он переходит в рабочее состояние

```
async def start_parsing__set_file(message: types.Message):
    dict_product = await load_product_file()

    write_to_user_data(message.chat.id,
                       data=[dict_product, "dict_product",
UserDataAction.ADD])

    await message.answer('Жду файл для обработки')
    await Parsing_states.waiting_for_file.set()
```

Рисунок 23 – Код ожидания файла от пользователя

После запуска бот присылает сообщение о том, что ожидает файл xlsx от менеджера направления за запуск скрипта парсинга.

```
async def waiting_for_file(message: types.Message, state:
FSMContext):
    """
    Функция вызывается, когда бот находится в состоянии
waiting_for_file и получает сообщение.
    Она обрабатывает загруженный пользователем файл и переводит бота в
состояние wait_input_parse_param.
    """
    if document := message.document:
        if document.file_name.split('.')[-1] != 'xlsx':
            await message.answer('Файл должен быть в формате `.xlsx`',
                                 parse_mode='Markdown')
            return

        user_id = message.from_user.id
        temp_file = await download_user_file(document, user_id)
```

Рисунок 24 – Код ожидания файла от пользователя

Как только файл получен происходит проверка его на правильность формата. Поскольку несколько раз менеджеры направления пытались загрузить либо битый файл либо файл другого формата, что приводило к поломке бота и его остановки была внедрена функция if document.file_name.split('.')[-1] != 'xlsx': которая позволила избежать этой ошибки.

```
# Замена производителей по общему списку
for item in list_product:
    default_name_company = item[1]
    cleaned_name_of_korea = ' '.join(default_name_company.split()).replace('KOREA',
'').strip()
    if default_name_company in dict_product:
        await replacement_product_file(dict_product, item, list_armtek, list_emex)
    elif cleaned_name_of_korea in dict_product:
        await replacement_product_file(dict_product, [item[0], cleaned_name_of_korea],
list_armtek, list_emex)
    else:
        list_emex.update({item[0]: list(set([cleaned_name_of_korea, Upper_format_name,
title_format_name]))})
        list_armtek.update({item[0]: list(set([cleaned_name_of_korea,
Upper_format_name, title_format_name]))})

print("EMEX:", list_emex)
print("ARMTEK:", list_armtek)
```

Рисунок 25 – Код замены производителей по списку

Как только проверка была пройдена запускается функция автоматической замены наименований производителей деталей согласно product файлу.

После того как файл готов к запуску АПД, бот спрашивает менеджера направления какой сайт он хочет выбрать для парсинга данных. И производит замену

```
    await message.answer(text=f"Отлично! Запускаю парсинг для `{company}`",
parse_mode='Markdown')

    p = []
    if company in ['оба', 'armtek']:
        p.append('armtek')
    if company in ['оба', 'emex']:
        p.append('emex')
```

Рисунок 26 – Код опроса пользователя на выбор сайта для парсинга

После выбора сайта запускается АПД.

Стоит уделить внимание немаловажной части АПД, которая подразумевает параллельный парсинг, а значит у нас должна быть возможность выбора необходимого количества браузеров для парсинга.

```
sync def emex_handler(message: types.Message,
list_companies_with_article_lists, is_article: bool = False, ):
    if is_article:

        articles_emex =
list_companies_with_article_lists['Article_EMEX']
        chat_id = message.chat.id

        chunks = split_dict(articles_emex, chunk_size=10)

        num_cores = multiprocessing.cpu_count() - 2
        num_cores = 4
        start_time = timeit.default_timer()
```

Рисунок 27 – Код параллельного парсинга сайта

С помощью переменной num_core мы можем задать количество браузеров, которые будут открываться при парсинге.

```
# Excel File Settings
worksheet = wb[f'{date}']
worksheet[f'A1'] = «Компания производитель»
worksheet[f'B1'] = «Артикул»
worksheet[f'C1'] = «Наименование детали»
worksheet[f'D1'] = «Рейтинг поставщика»
worksheet[f'H1'] = «Наименование поставщика»
worksheet[f'I1'] = «Последнее обновление»
worksheet[f'F1'] = «Количество деталей»
worksheet[f'E1'] = «Сколько будет идти деталь «
worksheet[f'G1'] = «Цена детали»
worksheet[f'J1'] = «Остановка:»
wb.save(output_file_name_exel)
```

Рисунок 28 – Код формирования отчета «План закупа»

Также в файле parsing_handler происходит формирование и запись в отчет «План закупа» в формате xlsx для обратной выгрузки его менеджеру направления. [21]

3.3 Разработка и отладка пользовательского интерфейса Telegram бота

Чтобы запуск бота происходил не команде /start мы добавляем функциональную кнопку «Запуск»

```
async def process_button_start(message: types.Message):
    await message.answer("🚀 Запуск", parse_mode='Markdown')
    await message.answer("Запуск `Parsing bot`...",
parse_mode='Markdown')
    await start_parsing__set_file(message)
```

Рисунок 29 – Код интерфейса телеграм бота

В данном отрывке кода представлена отработка важного исключения чтобы менеджер направления не мог никак иначе запустить бота, только как не нажав на кнопку была добавлена Emoji ракеты, так как написав в чат слово «Запуск» он мог активироваться и перейти в активное состояние. [22]

```
async def process_button_upload(message: types.Message):
    upload_message = await message.answer("📤 Выгрузка",
parse_mode='Markdown')
    write_to_user_data(user_id=message.chat.id,
                       stack_keyboards=[offloading_submenu_time,
KeyboardAction.ADD])
```

Рисунок 30 – Код выгрузки отчетов

Далее для удобства выгрузки отчетов за выбранный промежуток времени добавлена кнопка «Выгрузка»

```
await message.reply("Выберите действие:",
reply_markup=user_keyboard)
await message.answer("📜 Меню:", reply_markup=main_keyboard)
```

Рисунок 31 – Код для возвращения пользователя в «Меню»

Сообщение «Выберите действие» позволяет менеджеру направлению определиться с тем, что он хочет от бота и отработку какой функции необходимо запустить.

А кнопка «Меню» возвращает его в начало, а также сбрасывает у бота все в исходное положение, но при этом он остается активированным и ждет действий от менеджера направления.

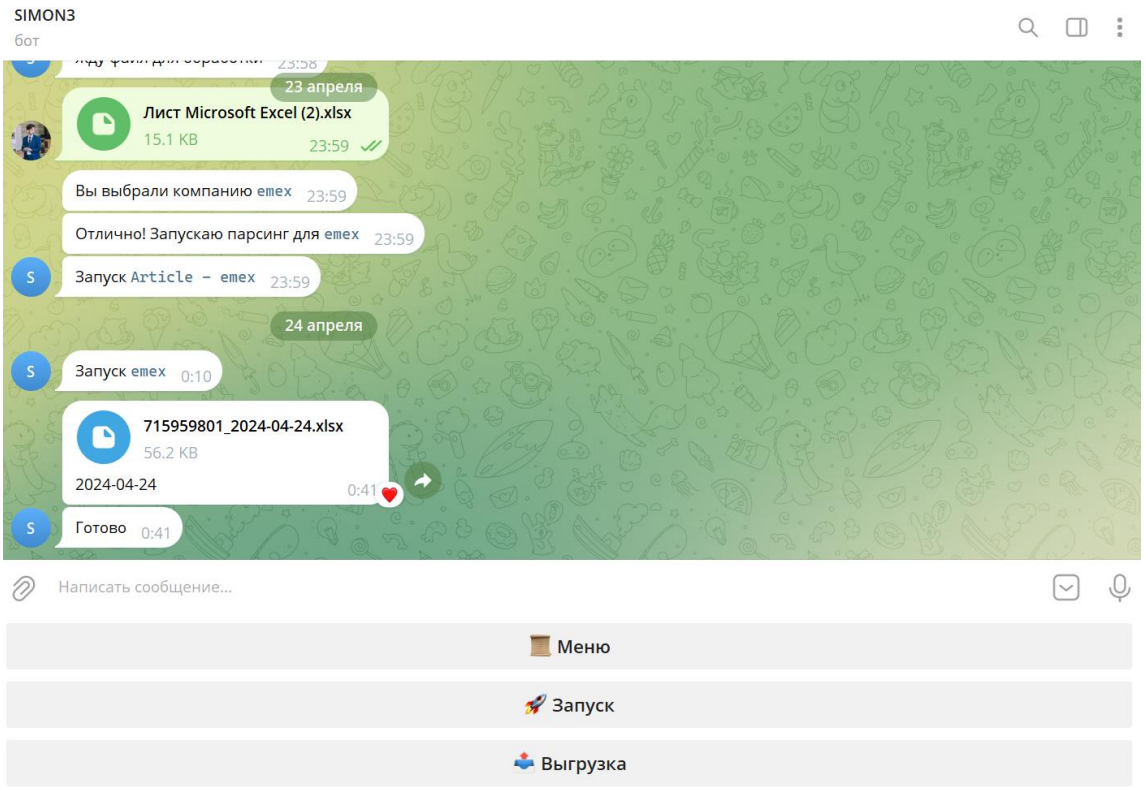

Рисунок 32 – Интерфейс Telegram бота

Рассмотрим подробнее каждую из кнопок кнопка «Меню»

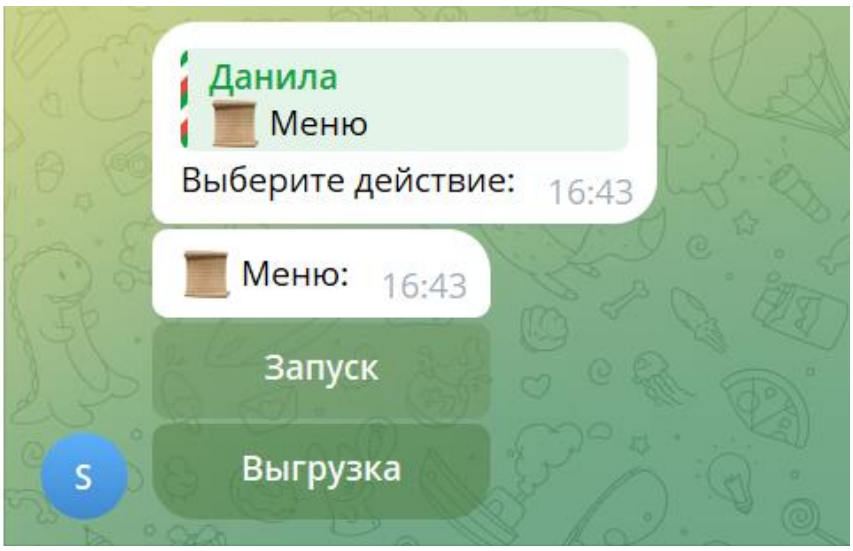

Рисунок 33 – Функционал кнопки «Меню»

Возвращает нас в начало и предлагает выбрать действие.

Кнопка «Выгрузка»

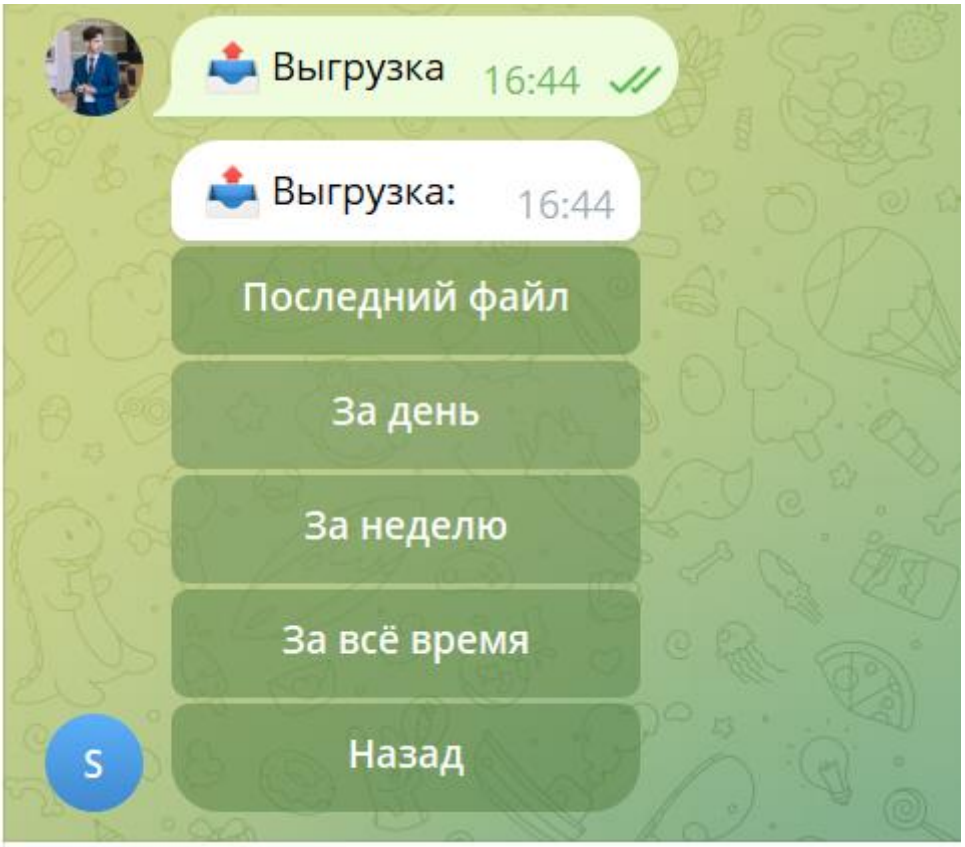

Рисунок 34 – Функционал кнопки «Выгрузка»

Предлагает нам выбрать период времени, за который нам надо выгрузить отчеты.

После выбора, например пункта «За все время» бот выдает нам следующую информацию: [24]

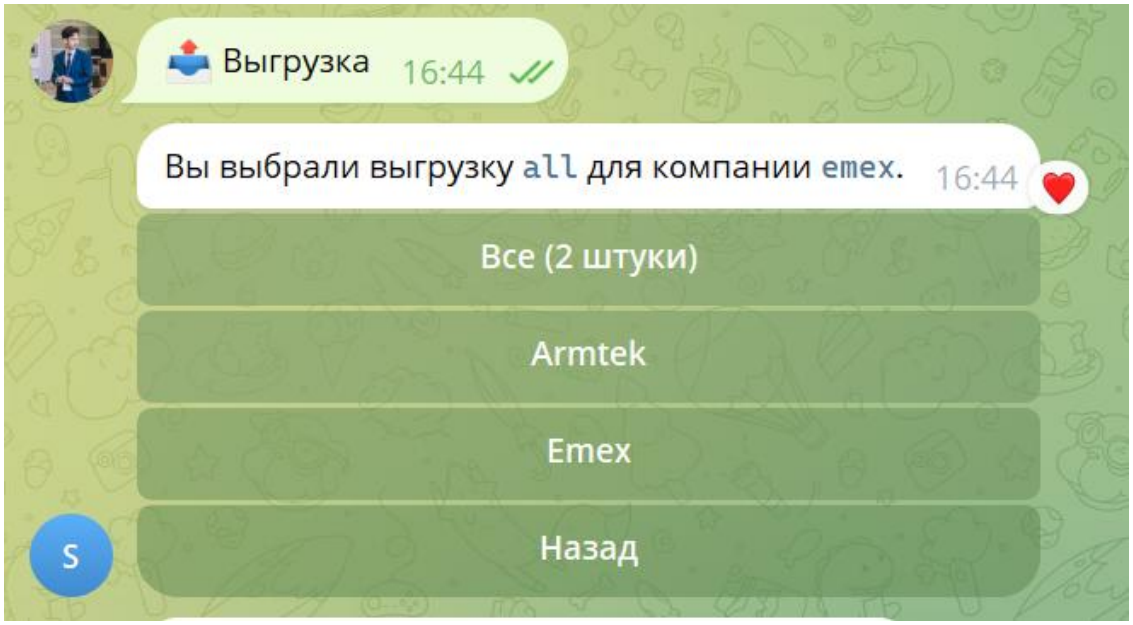

Рисунок 35 – Функционал кнопки «За все время»

Мы выбираем сайт Emex и получаем:

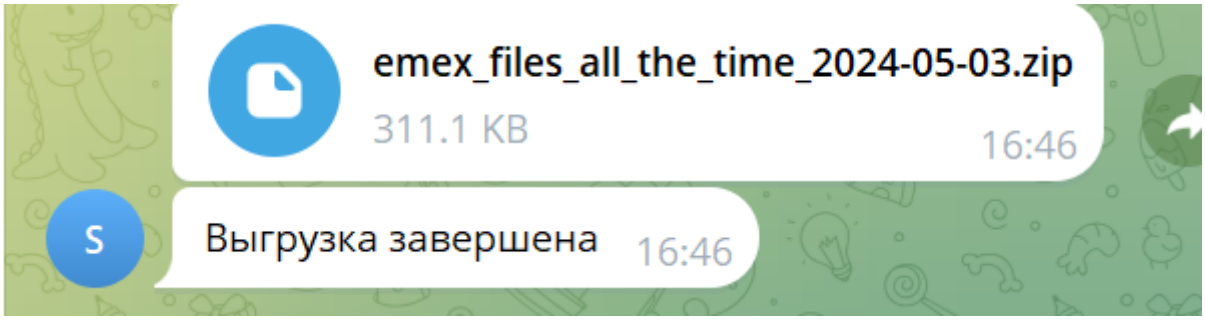

Рисунок 36 – Выгруженный zip архив отчетов «План закупа» за все время

Zip архив который можно сказать, в нем хранятся все отчеты «План закупа» за все время

emex_2024-01-15	Лист Microsoft Excel	6 КБ	Нет	6 КБ	0%	20.04.2024 17:43
emex_715959801_2024-02-18	Лист Microsoft Excel	20 КБ	Нет	20 КБ	0%	20.04.2024 17:43
emex_715959801_2024-04-20	Лист Microsoft Excel	19 КБ	Нет	19 КБ	0%	20.04.2024 18:21
emex_715959801_2024-04-22	Лист Microsoft Excel	9 КБ	Нет	9 КБ	0%	22.04.2024 13:56
emex_715959801_2024-04-24	Лист Microsoft Excel	57 КБ	Нет	57 КБ	0%	24.04.2024 0:42
emex_723782940_2024-02-10	Лист Microsoft Excel	12 КБ	Нет	12 КБ	0%	20.04.2024 17:43
emex_723782940_2024-02-13	Лист Microsoft Excel	13 КБ	Нет	13 КБ	0%	20.04.2024 17:43
emex_723782940_2024-02-18	Лист Microsoft Excel	25 КБ	Нет	25 КБ	0%	20.04.2024 17:43
emex_723782940_2024-02-19	Лист Microsoft Excel	79 КБ	Нет	79 КБ	0%	20.04.2024 17:43
emex_723782940_2024-02-20	Лист Microsoft Excel	52 КБ	Нет	52 КБ	0%	20.04.2024 17:43
emex_6835434024_2024-04-20	Лист Microsoft Excel	24 КБ	Нет	24 КБ	0%	20.04.2024 17:43

Рисунок 37 – Содержание zip архива выгрузки за все время

Как мы видим наш бот имеет интуитивно простой и понятный интерфейс, который позволяет с ним быстро взаимодействовать. А сопровождающие сообщения помогают информировать менеджера направления о состоянии парсинга данных.

3.4 Тестирование скрипта на различных источниках данных

Для тестирования скрипта выполним пробные запуски на выбранных двух сайтах Emex.ru и Armtek.ru.

Результат работы скрипта на сайте Armtek

	A	B	C	D	E	F	G	H	I	J
1	Компания	Артикул	Количеств	Цена дета	Поставка т	Наименов	Наименов	Последне	Склад	Возврат т
2	MOBIS	8191043A	5	1 741.07	02.05.2024	Личинка Н	610003	обновлен	ЦЗ Москв	14
3	MOBIS	8191043A	4	1 931.23	03.05.2024	Личинка Н	761296	обновлен	ЦЗ Москв	14
4	MOBIS	8191043A	4	2 096.09	04.05.2024	Личинка з	746171	обновлен	ЦЗ Москв	14
5	MOBIS	8191043A	4	3 551.72	05.05.2024	ЛИЧИНКА	695596	обновлен	ЦЗ Москв	14
6	MOBIS	8191043A	12	3 620.86	06.05.2024	Личинка Д	232438	обновлен	ЦЗ Москв	14
7	MOBIS	8191043A	8	3 761.80	07.05.2024	Личинка Д	387259	обновлен	ЦЗ Москв	14
8	MOBIS	8191043A	4	1 728.13	02.05.2024	Личинка з	782820	обновлен	ЦЗ Москв	14
9	MOBIS	8191043A	4	1 819.82	03.05.2024	Личинка з	626770	обновлен	ЦЗ Москв	14
10	MOBIS	8191043A	17	2 837.72	04.05.2024	Цилиндр	669270	обновлен	ЦЗ Москв	14
11	MOBIS	8191043A	4	1 751.65	05.05.2024	Личинка Н	843588	обновлен	ЦЗ Москв	14
12	MOBIS	834543A0	5	1 741.07	06.05.2024	Личинка Н	610003	обновлен	ЦЗ Москв	14
13	MOBIS	834543A0	4	1 931.23	07.05.2024	Личинка Н	761296	обновлен	ЦЗ Москв	14
14	MOBIS	834543A0	4	2 096.09	02.05.2024	Личинка з	746171	обновлен	ЦЗ Москв	14
15	MOBIS	834543A0	4	3 551.72	03.05.2024	ЛИЧИНКА	695596	обновлен	ЦЗ Москв	14
16	MOBIS	834543A0	12	3 620.86	04.05.2024	Личинка Д	232438	обновлен	ЦЗ Москв	14
17	MOBIS	834543A0	8	3 761.80	05.05.2024	Личинка Д	387259	обновлен	ЦЗ Москв	14

Рисунок 38 – Готовый отчет «План закупа» по сайту Armtek.ru

Как мы видим, был сформирован отчет, все данные были записаны в указанные колонки, условия компании для поиска были соблюдены.

	A	B	C	D	E	F	G	H	I	J	K
1	Компания	Артикул	Наименов	Рейтинг п	Сколько б	Количеств	Цена дета	Наименов	Последне	Остановка:	
2	Hyundai /	25239 480	Clutch-coc	3.7	2 дн.	4	11 104 ₽	ZETC	22 апр 2024		
3	Hyundai /	25239 480	Clutch-coc	1.0	3 дн.	5	11 965 ₽	SRUU	23 апр 2024		
4	Hyundai /	25239 480	Clutch-coc	3.0	3 дн.	4	12 508 ₽	OPJO	23 апр 2024		
5	Hyundai /	25239 480	Clutch-coc	2.7	3 дн.	3	13 287 ₽	BNCJ	22 апр 2024		
6	Hyundai /	25239 480	Clutch-coc	2.6	3 дн.	3	17 020 ₽	FRZA	23 апр 2024		
7	Hyundai /	25239 480	Clutch-coc	3.9	2 дн.	10	21 478 ₽	STSF	22 апр 2024		
8	YPR	23040414(	Кольца пс	4.9	2 дн.	13	5 935 ₽	ALIT	22 апр 2024		
9	YPR	23040414(	Кольца пс	4.5	2 дн.	13	5 935 ₽	ALJF	22 апр 2024		
10	YPR	23040414(	Кольца пс	4.2	2 дн.	3	5 935 ₽	BPCR	22 апр 2024		
11	YPR	23040414(	Кольца пс	1.0	3 дн.	3	6 714 ₽	FZUF	22 апр 2024		
12	YPR	23040414(	Кольца пс	3.5	2 дн.	13	7 002 ₽	STSB	22 апр 2024		
13	YPR	23040414(	Кольца пс	1.0	3 дн.	3	7 019 ₽	KCSA	22 апр 2024		
14	YPR	23040414(	Кольца пс	1.0	2 дн.	13	7 082 ₽	IPAH	22 апр 2024		
15	YPR	23040414(	Кольца пс	1.0	2 дн.	3	7 082 ₽	IPAA	22 апр 2024		
16	YPR	23040414(	Кольца пс	1.0	2 дн.	13	7 082 ₽	IPAB	22 апр 2024		
17	YPR	23040414(	Кольца пс	1.0	2 дн.	13	7 208 ₽	FZUE	22 апр 2024		
18	YPR	23040414(	Кольца пс	1.0	2 дн.	13	7 208 ₽	FZUC	22 апр 2024		

Рисунок 39 - Готовый отчет «План закупа» по сайту Emex.ru

На рисунке выше представлен результат работы на сайте emex.ru

По результатам тестирования можно сделать следующий вывод:

– Функционал обработки и замены производителей согласно product файлу работает;

– Запись сущностей в файл Excel происходит согласно указанным колонкам;

– Отработка исключений таких как срок поставки и количество деталей отработаны в полной мере;

– Обработанный отчет был выгружен назад и получен менеджером направления;

– Точность работы АПД на каждом из сайтов составила 99%;

3.5 Оценка производительности и эффективности скрипта

Для оценки производительности и эффективности скрипта выделим основные критерии для этого, а именно:[23]

- скорость выполнения;
- эффективность использования ресурсов;
- стабильность и надежность;
- точность и полнота данных;
- масштабируемость;
- интеграция с другими системами;
- обновление и поддержка.

Скорость выполнения парсинга данных гораздо выше по сравнению с ручным поиском менеджера направления. Для проведения примерных расчетов мы берем среднестатистические данные о одной заявке на закуп на день, которая состоит примерно из 200 позиций для поиска. Ниже представлены расчеты в таблице:

Таблица 3 – Скорость работы АПД «SIMON» в сравнении с менеджером направления

	1 рабочий день	1 месяц	1 год
АПД «SIMON»	1 час 40 минут	~34 часа	~8500 часов
Менеджер направления	3 часа 20 минут	~ 67 часов	~16750 часов

Как мы видим АПД тратит примерно в два раза меньше времени на поиск и запись в отчет одной автозапчасти, по сравнению с менеджером направления, но данный расчет был сделан с учетом одного работающего браузера, при их увеличении производительность растет в геометрической прогрессии.

Эффективность использования ресурсов: нагрузка АПД на сервер сравнима с одного человека работающему в браузере, соответственно на сервере с характеристиками:

– RAM 16 GB;
– SSD 512 GB;
– процессор core i5-12500H;
– видеокарта RTX 3060.

Может быть запущено до 20 одновременно работающих браузеров, что равняется 20 менеджерам направления при этом эта нагрузка равняется 1 работающему компьютеру.

Стабильность и надежность: АПД предусматривает бесперебойную работу при условии работы сервера, на который он загружен, все возможные ошибки, которые может допустить менеджер направления не вводят скрипт в критическую ошибку и не могут остановить парсинг.

Точность и полнота данных: АПД «SIMON» работает с 99% точностью выбирая из сайта все необходимые данные об автозапчастях и записывая их в отчет, 1% ошибок может происходить из-за битой разметки на сайте и какая-то автозапчасть была не полностью обработана, точность же менеджера направления составляет примерно 94% поскольку к концу дня у него может происходить замыливание глаза или человеческий фактор как, неправильно нажатая клавиша, что может привести к ошибке в цене или количестве.

Масштабируемость: на данный момент АПД ограничена в количестве одновременно работающих менеджеров направления в размере 3 человека на одну учетную запись, это ограничение можно обойти увеличением учетных записей для сайтов.

Интеграция с другими системами: АПД можно интегрировать с 1С, добавив функциональность по автоматической выгрузки по IP отчетов «План закупа» это будет полезно бухгалтерии для расчета сметы, прибыли и прочих показателей.

Обновление и поддержка: Поддержка и последующие обновления требуют работы 1 программиста, на данный момент скрипт вышел на автономный уровень работы, когда за ним практически необходимо присматривать, вмешательство человека будет лишь необходимо при

дополнительных разработках.

Выводы по третьей главе:

Была спроектирована архитектура клиент-серверной системы АПД «SIMON», выполнена разработка скрипта на языке Python и представлены отрывки кода с их описанием и назначением, отработаны различные виды ошибок возникающих в процессе тестирования и обоснована необходимость доработки скрипта, представлен пользовательский интерфейс Telegram бота с подробным рассмотрением каждой из функциональных кнопок, произведена оценка эффективности и производительности скрипта, по результатам которой он по всем параметрам превзошел человека, выполняющего ту же саму работу в ручную.

Заключение

В ходе выполнения данной работы были решены все поставленные задачи, что позволило всесторонне исследовать деятельность компании ООО «Торговый дом Токус», а также разработать и внедрить скрипт для парсинга данных. В результате проведенных исследований и разработки были достигнуты следующие цели:

Проведено тщательное исследование операционных процессов в отделе закупок компании ООО «Торговый дом Токус». В рамках этого этапа были изучены текущие методы работы отдела, выявлены основные проблемы и потребности в автоматизации процессов, связанных с обработкой и анализом данных. Это исследование стало фундаментом для последующей разработки скрипта для парсинга данных, позволяющего автоматизировать рутинные задачи и улучшить эффективность работы отдела.

В рамках второй задачи были рассмотрены современные подходы к парсингу данных. Изучены различные методы извлечения данных из веб-источников, включая HTML-разметку, XPath, CSS-селекторы и API. Были также проанализированы инструменты, такие как BeautifulSoup, Scrapy и Selenium, что позволило сделать обоснованный выбор оптимальных средств для реализации проекта.

На третьем этапе сформулированы функциональные требования, включающие необходимость извлечения данных из различных источников, гибкость настройки параметров парсинга, и возможность регулярного обновления данных. Нефункциональные требования касались производительности, устойчивости к ошибкам, безопасности и легкости в использовании скрипта.

На основе проведенного обзора инструментов для парсинга данных, было принято решение использовать комбинацию библиотек BeautifulSoup для парсинга HTML и Selenium для работы с динамическими веб-страницами. Такой выбор был обоснован необходимостью обеспечения высокой

производительности и гибкости решения.

Для эффективной организации и хранения извлекаемых данных была разработана логическая модель данных. Она включала определение основных сущностей и их взаимосвязей, что позволило структурировать данные таким образом, чтобы облегчить их дальнейшую обработку и анализ.

Был разработан и отлажен скрипт для парсинга данных. Реализованы основные функциональные возможности, предусмотренные требованиями, и проведены тестирование на нескольких контрольных примерах для

Проведено тестирование скрипта на различных веб-источниках для проверки его универсальности и способности корректно обрабатывать данные из разных структур и форматов. В результате тестирования были выявлены и устранены ошибки, что позволило повысить надежность и точность работы скрипта.

Были разработаны и внедрены механизмы обработки ошибок, что позволило обеспечить устойчивость работы скрипта в случае возникновения непредвиденных ситуаций.

В заключительной части работы была проведена оценка производительности и эффективности разработанного скрипта. Результаты показали, что скрипт успешно справляется с задачами парсинга данных, обеспечивая высокую скорость обработки и точность извлечения информации. Благодаря внедрению данного решения, отдел закупок ООО «Торговый дом Токус» получил инструмент для автоматизации рутинных операций, что позволило значительно улучшить оперативность и качество работы.

Таким образом, все задачи, поставленные в начале работы, были успешно выполнены. Проведенные исследования и разработка скрипта для парсинга данных позволили не только автоматизировать процессы обработки информации в отделе закупок, но и создать основу для дальнейшего улучшения и расширения функциональности системы.

Список используемой литературы

1. Бердников, Иван. "Python для анализа данных: основы." Ленанд, 2019. - 224 с.
2. Введение в Requests – Текст: электронный // digitalocean.com: [сайт]. – URL: https://www.digitalocean.com/community/tutorials/how-to-get-started-with-the-requests-library-in-python-ru
3. Дронов, В.А. Программирование. — СПб.: БХВ-Петербург, 2006. — 706 с.: ил.
4. Кнут, Д.Э. Искусство программирования: учеб. пособие: в 3 т.: пер. с англ. Т.1: Основные алгоритмы. - 3-е изд. - М. и др.: Вильямс, 2000. - 720 с.
5. Методы парсинга сайтов [Электронный ресурс]. — Режим доступа: http://seodrom.ru/parsing-saitov
6. Особенности HTML [Электронный ресурс]. — Режим доступа: http://lpgenerator.ru/blog/2013/10/21/chto-takoe-html-korotko-o-glavnom
7. Оценка трудозатрат – Текст: электронный // studwood.net: [сайт]. – URL: https://studwood.net/1990257/informatika/otsenka_trudozatrat_razrabotki_novogo_programmnogo_obespecheniya_osnove_modeli_sosomo
8. Панфилов, К. По ту сторону веб-страницы. — М.: ДМК Пресс, 2008. — 440 с.: ил.
9. Парсинг [Электронный ресурс]. — Режим доступа: https://www.seonews.ru/glossary/parsing/ (дата обращения: 28.05.2019)
10. Парсинг данных с сайта [Электронный ресурс]. — Режим доступа: https://semantica.in/blog/chto-takoe-parsing.html (дата обращения: 10.06.2019)
11. Парсинг сайтов. Описание принципов [Электронный ресурс]. — Режим доступа: http://web2033.com/parsing-basis
12. Поляков, Андрей. "Программирование на Python 3: курс для начинающих." ДМК Пресс, 2019. - 320 с.

13. Типичная структура web-сайта – Текст: электронный // cyberleninka.ru: [сайт]. – URL: https://cyberleninka.ru/article/n/aktualnost-sozdaniya-itipichnaya-struktura-web-sayta-kinoteatra/viewer (дата обращения: 01.05.2022)

14. Файл robots.txt – Текст: электронный // convertmonster.ru: [сайт]. – URL: https://convertmonster.ru/blog/seo-blog/kak-sozdat-pravilnyj-fajl-robots-txtnastrojka-direktivy/

15. Чтение и запись текстовых файлов. StreamReader и StreamWriter [Электронный ресурс]. — Режим доступа: https://metanit.com/sharp/tutorial/5.5.php (дата обращения: 06.06.19)

16. Что такое парсинг сайта, программы и примеры их использования – Текст: электронный // trinet.ru: [сайт]. – URL: https://www.trinet.ru/blog/kontent/parsing-sajta/ (дата обращения: 07.05.2022)

17. Bailey, Richard M. "Python Web Scraping: Hands-On data scraping and crawling using PyQT, Selnium, HTML and Python." Independently published, 2018. - 142 p.

18. Beautiful Soap – Текст: электронный // wiki.python.su: [сайт]. – URL: http://wiki.python.su/Документации/BeautifulSoup

19. BeautifulSoup [Электронный ресурс]. — Режим доступа: http://www.crummy.com/software/BeautifulSoup

20. Lawson, Richard. "Web Scraping with Python: Collecting More Data from the Modern Web." O'Reilly Media, 2018. - 308 p.

21. Mitchell, Andrew. "Web Scraping with Python: Successfully scrape data from any website with the power of Python." Independently published, 2019. - 134 p.

22. NumPy Tutorial – Текст: электронный // pythobyte.com: [сайт]. – URL: https://pythobyte.com/numpy-tutorial-a-simple-example-based-guideff3127f2/ (дата обращения: 15.04.2022)

23. Platt E. L. Network Science with Python and NetworkX Quick Start Guide / E. L. Platt [и др.]. – М. : Издательство Packt Publishing, 2019. – 190 с.

24. Python – Текст: электронный // ru.wikipedia.org: [сайт]. – URL: https://ru.wikipedia.org/wiki/Python (дата обращения: 15.04.2022)

25. Rosebrock, Adrian. "Python Web Scraping: Crawling and scraping using the requests, BeautifulSoup and Selenium libraries." PyImageSearch, 2020. - 284 p.