МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н.Г.ЧЕРНЫШЕВСКОГО»

Кафедра математического обеспечения вычислительных комплексов и
информационных систем

РАСПРЕДЕЛЕННАЯ СИСТЕМА АВТОМАТИЗАЦИИ ЗАКАЗОВ КОНТРОЛЯ И ОПТИМИЗАЦИИ ПОСТАВОК ДЛЯ OZON

АВТОРЕФЕРАТ МАГИСТЕРСКОЙ РАБОТЫ

студента 2 курса 273 группы
направления 02.04.03 Математическое обеспечение и администрирование
информационных систем
факультета компьютерных наук и информационных технологий
Сапожникова Александра Анатольевича

Научный руководитель:
д. ф.-м. н.. _____ Д. К. Андрейченко

подпись, дата

Зав. кафедрой:
д. ф.-м. н.. _____ Д. К. Андрейченко

подпись, дата

Саратов 2023

ВВЕДЕНИЕ

Актуальность темы. Данная работа посвящена выявлению и устранению ряда проблем в существующей системе автоматизации заказов, контроля и оптимизации поставок ozon. Существующие системы подобного типа хорошо оптимизированы для больших складских помещений. Вместе с тем, для более эффективной доставки товаров в процессе интернет-торговли в крупных городах обычно используется достаточно большое число пространственно разнесенных относительно малых складских помещений, называемых даркторами.

Следовательно, актуальна задача разработки распределенной системы автоматизации заказов, контроля и оптимизации поставок ozon для ее использования в организации с большим количеством относительно малых складских помещений.

Все цепочки поставок следует рассматривать с точки зрения работы с подсистемой электронной торговли ozon fresh, изначально предназначенной для торговли товарами массового потребления с коротким сроком хранения. Необходимо проанализировать основные различия и сходства функционирования больших складов и даркторов, и на основе проведенного анализа сформулировать требования к разрабатываемой информационной системе и ее концептуальной архитектуре.

Требование стабильности и масштабируемости подсистемы хранения данных приводит к необходимости использования СУБД PostgreSQL. Параллельная асинхронная обработка данных требует использования брокера сообщений Kafka. Для обеспечения требуемой производительности разработку оптимизированных для целевого процессора программных компонент информационной системы следует проводить на языке Golang. Для развертывания программных компонент распределенной системы используются контейнеры Docker и Swarm для управления ими. Это в основном определяет стек технологий и средств разработки, используемых в работе.

Цель магистерской работы - разработка распределенной системы автоматизации заказов, контроля и оптимизации доставок для Ozon рассчитанную на пространственно разнесенных относительно малых складских помещений.

Поставленная цель определила **следующие задачи**:

1. Изучить особенности процессов работы даркстора с точки зрения электронной торговли в ozon.
2. Выполнить анализ основных различий и сходства функционирования больших складов и даркстороов и сформулировать требования к архитектуре разрабатываемой информационной системы.
3. На основе PostgreSQL создать БД с профилями пользователей, настроечными данными для работы со складскими помещениями и т.д.
4. Обеспечить параллельную асинхронную обработку информации на основе брокера сообщений Kafka.
5. На основе Golang разработать программные компоненты распределенной системы.
6. Обеспечить развертывание компонент разработанной системы автоматизации заказов, контроля и оптимизации поставок на основе Docker Swarn.

Методологические основы распределенной система автоматизации заказов контроля и оптимизации поставок для ozon представлены в работах Siddik S. G., Evers B., Kumar V., Schulte-Zurhausen E..

Теоретическая значимость магистерской работы. Дает представление о процессах на крупных складах и распределенных системах в e-commerce, а также описывает различные компоненты, необходимые для их работы.

Практическая значимость магистерской работы. Представляет собой разработку эффективной распределенной системы автоматизации заказов, контроля и оптимизации поставок для работы с подсистемой электронной торговли ozon fresh, которая может быть использована в реальной жизни для оптимизации бизнес-процессов и повышения эффективности работы в сфере e-commerce и логистики. Весь исходный код системы предоставлен, что позволяет использовать его в качестве основы для создания новых систем и функциональностей.

Структура и объём работы. Магистерская работа состоит из введения, 3 разделов, заключения, списка использованных источников и 3 приложений. Общий объем работы – 91 страница, из них 65 страниц – основное содержание,

включая 8 рисунков и список использованных источников информации – 22 наименований.

КРАТКОЕ СОДЕРЖАНИЕ РАБОТЫ

Первый раздел «Анализ требований, предъявляемых к распределенной си- стеме автоматизации заказов, контроля и оптимизации поставок ozon»

В современных условиях развития технологий и доставки, существует множество типов складов, каждый из которых обладает своими особенностями и специализацией. Одним из наиболее распространенных типов складов является большой склад и даркстор.

Целью работы является изучение и создания определённых процессов, протекающих не на большом складе, а на дарксторе. В первую очередь, следует обратить внимание на различия в размере и функциональности данных типов складов.

Большой склад требует более серьезной организации работы, необходимости обработки товаров, наблюдения за уровнем запасов и общего контроля за процессами складирования.

В свою очередь, даркстор требует более точного выбора условий хранения, а также конкретного типа оборудования. Важен подход к выбору складских систем, надежности и технологий управления, а также пользователям требуется высококвалифицированный персонал. Дарксторы очень специфичны и могут выглядеть по разному.

Помимо этого, необходимо учитывать важность обработки заказов и доставки товаров, что на дарксторе является ключевым фактором, а на большом складе это обычно не является самой важной задачей. Однако, несмотря на все существующие различия, существуют и общие принципы работы на обоих типах складов. Например, необходимость повышения эффективности работы, контроля за запасами и учета движения товаров, организации надежной системы безопасности и управления персоналом.

Распределенные системы являются важным элементом современных складов, особенно тех, которые используют концепцию даркстора. Кроме того, на даркстораx может использоваться различная автоматизация.

Разработка информационной системы в e-commerce является сложным процессом, который включает в себя ряд вызовов и проблем. Информационная система e-commerce обрабатывает огромное количество данных, таких как товары, категории, описание товаров, фотографии, цены, клиентская и платежная информация, заказы, отзывы клиентов и т.д. Обработка таких объемов данных является сложной задачей, требующей определенных технических знаний и навыков, а также инфраструктуры.

Эта система содержит множество чувствительных данных, должна быть защищена от взлома, хакерских атак, кражи личности и других видов угроз безопасности. Это требует определенных технических знаний и использования различных инструментов защиты данных.

Покупатели используют различные устройства и браузеры при посещении сайта e-commerce. Для того, чтобы обеспечить наилучший опыт пользователей, рекомендуется создание доступного сайта, который может корректно отображаться на всех устройствах и браузерах и иметь одинаковую функциональность.

Информационная система e-commerce обычно интегрируется с другими информационными системами, такими как CRM, ERP, бухгалтерские программы и т.д. Чтобы обеспечивать корректную и эффективную интеграцию, необходимо учитывать стандарты и протоколы обмена данных между системами, что приводит к частым коммуникациям между командами разработки и следовательно замедляет процесс добавление новой функциональности из-за блокировок.

К тому же рынок e-commerce является динамичным и постоянно меняющимся. Про это подробно расписано в статье «Современные тенденции развития рынка электронной коммерции», хоть в ней рассматриваются процессы до 2021 года включительно, но видно, что даже в 2023 году этот процесс активно развивается.

Таким образом, разработка информационной системы в e-commerce является сложным процессом, который требует определенных технических знаний и навыков, а также инфраструктуры. Ее основными вызовами являются обработка большого объема данных, защита данных от угроз безопасности, создание доступного сайта для всех устройств и браузеров, эффективная ин-

теграция с другими информационными системами и быстрая адаптация к динамичным изменениям рынка e-commerce.

Второй раздел «Проектирование архитектуры и программных компонент информационной системы»

Одним из ключевых процессов на складах даркстор является приемка товаров, которая осуществляется через ворота приемки и временные слоты.

Ворота приемки на складах позволяют упростить процесс приемки товаров и обеспечить его максимальную эффективность. Ворота приемки делятся на несколько типов в зависимости от грузов, которые они могут принимать.

Временные слоты на складах предоставляют возможность для организации удобного и эффективного времени приемки товаров. Благодаря временным слотам, клиенты могут заранее выбирать для себя удобное время для приемки товаров. Кроме того, на даркскорах временные слоты также помогают снизить время, которое необходимо для ожидания приемки товаров.

Для обеспечения мобильности настройки параметров приемки на даркскорах было решено использовать не обычное мобильное приложение или сайт, а бот в мессенджере. Можно с уверенностью сказать, что боты в мессенджерах обладают множеством преимуществ перед мобильными приложениями, включая простоту использования, скорость обработки запросов, отсутствие необходимости в установке обновлений и дополнительного программного обеспечения и другое.

Если подходить к проектированию, то для ботов есть один нюанс. На сайтах или приложениях состояние пользователя и куда ему можно перейти храниться у него на устройстве в виде ui. В случае бота состояние пользователя и возможные переходы необходимо явно указывать и сохранять в самой бизнес логике приложения.

Современные системы электронной коммерции должны работать в реальном времени и обеспечивать быстрое и точное выполнение заказов. Существует множество инструментов, которые позволяют улучшить и оптимизировать процессы обработки данных. В данной работе будет рассмотрена архитектура асинхронного импорта дополнительных данных для заказов, которая может быть использована в электронной коммерции.

Для эффективного функционирования системы, необходимо иметь доступ к дополнительным данным, которые могут влиять на обработку заказов. Такими данными могут быть информация об инвентаре, информация о доступности товаров, и другие параметры, которые являются критическими для быстрого и эффективного выполнения заказов. Зачастую, эти данные могут быть получены из внешних систем, например, из CRM, ERP или других систем.

В данный момент используется синхронная система импорта дополнительных данных для заказов, однако этот подход себя изжил. Он был очень удобен, когда бизнесу требовалось быстро создать систему с возможностью загружать данные через импорты, но сейчас уже не появляется новые импорты, а существующие усложняются с каждым днём, что приводит к их долгой работе.

Асинхронный механизм импорта данных на основе очередей сообщений является одной из наиболее эффективных архитектурных решений для импорта дополнительной информации в контексте системы электронной коммерции.

В итоге, в данном разделе были рассмотрены ключевые процессы на складах даркстор, включая приемку товаров через ворота приемки и временные слоты. Для обеспечения мобильности и удобства использования был выбран подход с использованием бота в мессенджере. Была также рассмотрена архитектура асинхронного импорта дополнительных данных для заказов в системах электронной коммерции, на основе очередей сообщений. Этот подход является более эффективным и гибким, чем синхронный импорт данных, который был использован ранее.

Третий раздел «Реализация и анализ эффективности распределенной системе автоматизации заказов, контроля и оптимизации поставок ozon»

Для компонентов описанных во втором разделе было выбрано использовать микросервисную архитектуру, а не монолитную.

Микросервисная архитектура стала популярной в последнее время из-за ее надежности, масштабируемости и адаптивности. Эта архитектура позволяет независимо разрабатывать и развивать компоненты приложения, что дает

возможность быстрой адаптации к изменениям в бизнес-процессах и повышает гибкость системы в целом. Такой подход позволяет ускорить поставку программного обеспечения и повысить качество разработки, а также может приводить к уменьшению издержек на разработку и поддержку проектов.

Микросервисная архитектура позволяет независимо разрабатывать и изменять отдельные компоненты приложения, а также эффективно масштабировать их для обеспечения высокой производительности и доступности системы. По этим причинам решение для дакрсторов и коммерции можно писать как два отдельных приложения, а не создавать из них монолит.

При разработке микросервисного приложений на языке Golang важно выбрать правильную библиотеку для работы с базой данных, так как это критически важный элемент архитектуры. Была выбрано использовать jackc/pgx для работы с базой данных PostgreSQL. Использование нативной библиотеки для работы с базой данных может быть более предпочтительным в некоторых случаях, так как обеспечивает больший контроль над процессами взаимодействия с базой данных и более читаемый код. В этом случае важно выбрать хорошо документированную и надежную библиотеку для работы с базой данных.

В ходе проектирования данной системы в прошлой главе, было принято решение использовать бота в качестве интерфейса. Для реализации бота в проекте была использована библиотека gopkg.in/telebot для Go. Она предоставляет удобный API для взаимодействия с telegram ботом и значительно упрощает процесс разработки.

Если же обращаться к реализации самой логики изменения настроек склада, то там каких-то особенностей выходящих из начальной архитектуры и требующих пояснений не было, поэтому обьяснение реализации здесь опускается.

Как уже прописывалось важным компонентом для асинхронного импорта является kafka и работа с ней. Естественно для работы с Kafka понадобиться библиотека, способная обеспечить простое и надежное взаимодействие с кластером Kafka и обработку сообщений. В качестве библиотеки была выбрана Shopify/sarama. Она обладает широким набором возможностей, таких как поддержка SSL, SASL, автоматический балансировщик нагрузки, много-

поточная обработка сообщений и многое другое. Кроме того, данная библиотека разрабатывается и поддерживается командой Shopify, что обеспечивает высокую степень надежности и качества ее кода, помимо этого команда гарантирует поддержку двух последних версий kafka.

Обработки разных типов импортов через условные оператор if не удобно и не эффективно. Лучше это делать через стратегию. Если в будущем появится новый формат импорта или изменится существующий формат, достаточно просто добавить новый класс-стратегию для обработки этого формата без изменения существующего кода.

Запуск описанных выше приложений через docker предоставляет несколько преимуществ, которые могут сделать процесс настройки, развертывания и поддержки приложений проще и более эффективным. Docker позволяет упаковывать приложения в контейнеры и обеспечивает повторяемость при развертывании приложений.

Выбор инструментов для управления контейнерами зависит от размера и сложности приложения, а также от потребностей в масштабируемости и управлении ресурсами. Выбор между Docker Swarm и Kubernetes зависит от конкретной задачи и контекста. Для небольших и средних приложений в Docker обычно используется Docker Swarm, так как он обладает простотой настройки, и достаточно легок в использовании. Естественно для данной работы будет достаточно использовать Docker Swarm.

В docker-swarm нет ничего для запуска крон, что не очень удобно, но crazy-max/swarm-cronjob предоставляет удобный способ контроля периодических задач в кластере. Он дополняет нехватающий функционал, который позволяет запускать сервисы, но не имеет встроенного механизма запуска периодических задач, таких как крон.

Благодаря использованию swarm-cronjob, это дало возможность использовать стандартный формат крон-заданий для настройки и управления периодическим запуском контейнеров в Docker Swarm.

В двух приложениях без дополнительных интеграций используется много чувствительной информации как минимум это логин и пароль для подключения к базе. Безопасность играет очень важную роль в современной инфраструктуре. Утечка конфиденциальных данных может привести к серьез-

ным последствиям, включая угрозу безопасности пользователей и компании, а также юридические проблемы. Использование Vault позволяет централизованно хранить и управлять секретами, обеспечивает безопасность данных и защищает их от несанкционированного доступа. Кроме того, Vault предоставляет надежные механизмы аутентификации и авторизации, обеспечивая доступ только уполномоченным пользователям или процессам.

Развёртывания Vault в инфраструктуре docker swarm очень похожа на postgreSQL, но со своими небольшими особенностями, которые не требует особого внимания.

Таким образом, было выбрано использование микросервисной архитектуры и библиотеки jackc/pgx для работы с базой данных PostgreSQL. Для работы с Kafka была выбрана библиотека Shopify/sarama. Обработку различных форматов импортов сделан через стратегию. Был использован Docker Swarm для запуска контейнеров. Также был применён инструмент swarm-cronjob для управления периодическими задачами. Для обеспечения безопасности был использован Vault для централизованного хранения и управления секретами.

Заключение

Данная работа описывает процессы на большом складе и их отличия и сходства от процессов в даркстоpе. Рассматриваются распределенные системы на даркстоpе, их компоненты и необходимость их использования. Кроме того, описываются основные сложности работы в e-commerce.

Разработана эффективная распределенная система автоматизации заказов, контроля и оптимизации поставок для работы с подсистемой электронной торговли ozon fresh, предназначенная специально для торговли товарами массового потребления с коротким сроком хранения на большом количестве пространственно разнесенных даркстоpов.

Эффективность подсистемы хранения данных обеспечивается применением СУБД Postgree SQL. Параллельная асинхронная обработка данных основана на использовании брокера сообщений Kafka. Для обеспечения требуемой производительности, оптимизированные для целевого процессора программные компоненты реализованы на языке программирования Golang. Разверты-

вание программных компонент разработанной распределенной системы достаточно просто и надежно обеспечивается на основе Docker Swarm.

Отдельные части магистерской работы были опубликованы:

Сапожников А. А. Дизайн и разработка архитектуры бота в мессенджерах: ключевые аспекты и примеры реализации // Студенческий. - 2023. - №20 (232) часть 2. - С. 33-35.

Основные источники информации:

1. Evers, B. The effects of specialisation on logistics networks: A comparative case analysis / B. Evers, R. Obermaier, J. Putzke // Procedia CIRP. — 2018. — Vol. 72. — Pp. 417–422.
2. Kumar, V. Warehouse operations management: A review of the literature / V. Kumar, S. Verma // International Journal of Industrial Engineering Computations. — 2021. — Vol. 12, no. 3. — Pp. 473–494.
3. Schulte-Zurhausen, E. Time-slot management for inbound logistics at a multi- user warehouse: A case study / E. Schulte-Zurhausen, B. Hellingrath // International Journal of Production Economics. — 2016. — Vol. 181. — Pp. 322–333.
4. Siddik, S. G. Design and Control of Warehouse Automation Systems / S. G. Siddik, M. R. Gareynder. — Springer, 2020.
5. Rabl, T. Join processing in scalable distributed stream processing systems / T. Rabl, M. Sadoghi, H.-A. Jacobsen, S. Mankovskii // Distributed and Parallel Databases. — 2016. — Vol. 34, no. 2. — Pp. 137–166.
6. Тропынина, Н. Е. Современные тенденции развития рынка электронной коммерции / Н. Е. Тропынина, П. О. Логинов // Инновационная экономика: перспективы развития и совершенствования. — 2022. — С. 84–89.