Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение
высшего образования
«Уральский федеральный университет
имени первого Президента России Б.Н. Ельцина»

Институт радиоэлектроники и информационных технологий – РТФ

Школа бакалавриата

ОТЧЕТ

По проекту
«Разработка админки для управления настройками бота отслеживания цен на маркетплейсах»

по дисциплине «Проектный практикум»

Заказчик: Пиксаева А.Ю. _____

Куратор: Пиксаева А.Ю.

СКБ-ЛАБ _____

Студенты команды OVERCODE

Тиунов А.И. _____

Козлова А.А. _____

Зарипов К.Ю. _____

Ефимов В.С. _____

Карсканов Н.А. _____

Екатеринбург, 2025

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В современных условиях стремительного развития электронной торговли и динамичного ценообразования на маркетплейсах растет потребность в автоматизированных инструментах ценообразования и аналитики. Предлагаемый продукт позволит коммерческим и аналитическим отделам компаний эффективно настраивать мониторинг ценовых изменений, быстро исследовать новые торговые площадки и отслеживать активность конкурентов по своим продуктам.

Существующий бот уже доказал свою эффективность в отслеживании цен на платформе Wildberries, однако отсутствие сквозного интерфейса для администрирования ограничивает гибкость его применения и дальнейшее развитие функциональности. Управление правилами и настройками в текущей реализации требует прямого взаимодействия с кодом или базой данных, что делает его неудобным и недоступным для широкой аудитории, не обладающей технической подготовкой.

Разработка удобной и интуитивно понятной административной панели решает данную проблему, предоставляя интерфейс для управления всеми основными сущностями: маркетплейсами, правилами отслеживания и пользователями. Благодаря этому значительно снижается порог вхождения для новых пользователей и администраторов, а сам сервис становится более масштабируемым и удобным в поддержке.

1 Цель и задачи проекта

Упростить и централизовать управление настройками сервиса для отслеживания цен на маркетплейсах путём разработки веб панели администрирования, которая позволит:

– Настраивать перечень отслеживаемых маркетплейсов;

– Определять и редактировать правила мониторинга цен;

– Управлять связями «пользователь – правило – маркетплейс»;

– Расширять функционал бота за счёт подключения новых площадок и типов проверок.

Для достижения поставленной цели будет выполнен следующий комплекс задач:

– Спроектировать и реализовать полноценный веб-интерфейс с CRUD-операциями для маркетплейсов, правил отслеживания и их связей с пользователями.

– Интегрировать дополнительный маркетплейс в систему мониторинга.

– Обеспечить надёжность и читаемость кода, покрыть юнит-тестами.

– Подготовить систему к развертыванию на сервере.

Реализация этих задач позволит создать удобный, масштабируемый и надёжный и просто разворачиваемый инструмент для заказчика.

2 Требования заказчика

В ходе встречи с заказчиком были сформулированы следующие ключевые функциональные и нефункциональные требования

2.1 Функциональные требования:

2.1.1 Административная панель (веб-интерфейс)

Удобный и понятный пользовательский интерфейс для администратора.
CRUD-функции (создание, редактирование, удаление, просмотр) для следующих сущностей:

Маркетплейсы - платформа (Wildberries, Aliexpress и др.), на которой производится мониторинг цен.

Правила отслеживания цен - набор условий, при которых система уведомляет пользователя об изменениях цены (например, снижение ниже заданного порога).

Связи между пользователями, маркетплейсами и правилами - возможность назначать конкретные правила конкретным пользователям на конкретных площадках.

2.1.2 Расширение поддерживаемых маркетплейсов:

Возможность добавления новых платформ без существенных изменений в архитектуре системы.

2.2 Нефункциональные требования

2.2.1 Масштабируемость

– Архитектура должна позволять расширение количества маркетплейсов и пользователей без снижения производительности.

Надёжность и отказоустойчивость:

– Система должна корректно работать при частичных сбоях (например, временная недоступность одного из маркетплейсов).

– Все изменения должны сохраняться и не приводить к потере данных при ошибках.

2.2.2 Безопасность

– Доступ к административной панели должен быть защищён (авторизация, HTTPS и т.п.).

– Защита от SQL-инъекций, XSS и других типичных уязвимостей.

2.2.3 Юзабилити (удобство использования)

– Интерфейс должен быть интуитивно понятен для человека без технической подготовки.

– Все действия должны сопровождаться понятной обратной связью (успешно выполнено, ошибка, предупреждение и т.д.).

3 Календарный план

Таблица 1 – Календарный план проекта

Календарный план																	
Название проекта: Разработка админки для управления настройками бота отслеживания цен на маркетплейсах																	
Руководитель проекта: Пиксаева А.Ю.																	
№	**Задача**	**Ответственный**	**Неделя**														
			1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
Инициирование																		
1.1	Разработка плана	Зарипов К.Ю.																
Подготовка																		
2.1	Подготовка среды разработки	Команда																
2.2	Встретиться с куратором	Команда																
Работа над проектом																		
3.1	Провести системный анализ	Карсканов Н.А.																
3.2	Провести аналитику конкурентов	Зарипов К.Ю.																
3.3	Добавить автоматическую контейнеризацию	Ефимов В.С.																
3.4	Провести UI/UX анализ	Козлова А.А.																
3.5	Внедрить маркетплейс	Карсканов Н.А.																
3.6	Внести правки после сис. анализа	Ефимов В.С.																
3.7	Рефлексия спринта	Тиунов А.И																
3.8	Подготовка КТ1	Зарипов К.Ю. Тиунов А.И																
3.9	Создать эндпоинты правил	Ефимов В.С.																
3.10	Создать макет	Козлова А.А.																
3.11	Создать эндпоинты маркетплейсов	Ефимов В.С.																
3.12	Тестирование бэкенда	Зарипов К.Ю. Карсканов Н.А.																

Продолжение таблицы 1 Календарный план проекта

Календарный план																
Название проекта: Разработка админки для управления настройками бота отслеживания цен на маркетплейсах																
Руководитель проекта: Пиксаева А.Ю.																
			Неделя													
№	Задача	Ответственный	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
3.13	Рефлексия спринта	Тиунов А.И.																
3.14	Подготовка КТ2	Зарипов К.Ю. Тиунов А.И																
3.15	Верстка	Тиунов А.И.																
3.16	Тестирование фронта	Зарипов К.Ю.																
3.17	Отладка	Команда																
3.18	Рефлексия спринта	Тиунов А.И.																
3.19	Подготовка КТ3	Зарипов К.Ю. Тиунов А.И																
3.20	Доработки	Команда																
Защита проекта																		
4.1	Защитить проект	Команда																

До начала активной стадии разработки команда провела серию подготовительных мероприятий. После встречи с заказчиком на основании полученных требований команда сформировала план работы над проектом, в который вошли все основные этапы реализации, а также сроки их выполнения.

Для эффективной коммуникации и отслеживания прогресса использовалась GitHub-доска задач, где каждая задача имела ответственного, описание и дедлайн. Также использовалась система меток и статусов что позволяло прозрачно управлять рабочим процессом и своевременно выявлять потенциальные узкие места.

4 Анализ аналогов

На рынке существует ряд решений, направленных на мониторинг цен на маркетплейсах. Ниже представлены похожие и наиболее известные сервисы с кратким сравнительным анализом по ключевым параметрам.

Таблица 2 – Анализ аналогов

Продукты	Metacommerce	Wbmonitor	Price-track	Marketparser	Cheaper
Отслеживает	Офлайн-Розницу	Все маркетплейсы	Lamoda	Все маркетплейсы	Все маркетплейсы
Цена	Недоступна	Бесплатно	Бесплатно	от 9500 с лимитами	Бесплатно
Поддержка	Актуальный сервис	Часть не работает	Проект заброшен	Актуальный сервис	Работает с перебоями
Функционал	Уведомления, Ассортимент, Отслеживание ротации	История, Уведомления	Уведомления	Уведомления, Отчеты	Поиск товаров
UI/UX	Функционал разбит на много страниц/подменю	Сломан на смартфоне	Консольное приложение	Без серьезных недостатков	Доступен только как приложение на смартфоне

Функционально большинство сервисов ограничены только уведомлениями и базовой аналитикой (цена, история). Инструменты гибкой настройки правил мониторинга, настройки пользователей или интеграции с ботами отсутствуют либо доступны только в виде платных решений.

Интерфейсы у ряда решений перегружены, неудобны или вовсе отсутствуют (например, консольные версии или только мобильные приложения).

Поддержка многих сервисов оставляет желать лучшего: есть открытые проекты, которые были заброшены, или работают с неполадками неделями.

Стоимость некоторых продуктов может делать их малодоступными для малого бизнеса или частных пользователей.

Преимущества разрабатываемого продукта:

– Бесплатный и открытый инструмент.

– Расширяемая архитектура с возможностью подключения любых маркетплейсов.

– Удобный веб-интерфейс для настройки правил, пользователей и площадок.

– Интеграция с Telegram-ботом и потенциальная возможность SaaS.

– Стабильная поддержка и актуальность проекта за счёт открытой архитектуры.

Наш проект должен закрыть нишу бесплатных, self-hosted решений для малого бизнеса закрывая функциональные и технические пробелы существующих решений, предлагая гибкий, удобный и поддерживаемый инструмент для мониторинга цен.

5 Архитектура проекта

Система построена по микросервисному принципу и включает три основных сервиса, а также общие инфраструктурные компоненты:

1) API-шлюз - Реализован на Java с использованием Spring Boot. Выполняет маршрутизацию HTTP-запросов от веб-админки и Telegram-бота к соответствующим микросервисам. Обеспечивает единую точку входа и отвечает за аутентификацию/авторизацию. Передаёт события в шину сообщений Kafka

2) Scrapper - Написан на Java/Spring Boot. Подписывается на топики в Kafka и по заданным правилам периодически парсит страницы маркетплейсов. Обрабатывает результаты - сравнивает цены с пороговыми значениями, формирует события “ изменение цены” и публикует их в Kafka.

3) Telegram-бот - Тоже Spring Boot приложение, интегрированное с Telegram API. Подписывается на события из Kafka и отправляет соответствующие уведомления пользователям. При получении команд от администратора через бот запрашивает у gateway текущие настройки и передаёт команды на изменение конфигурации.

4) Kafka - Гарантирует надёжную и асинхронную передачу событий между gateway, scrapper и ботом.

5) PostgreSQL - Хранит конфигурацию системы: перечень маркетплейсов, правила мониторинга, связи между пользователями и правилами, историю уведомлений. Подключается ко всем микросервисам через Spring Data JPA

6) Фронт энд - React + TypeScript. Обеспечивает весь функционал администрирования: CRUD на маркетплейсы, правила. Работает по REST-контракту с API-шлюзом.

Для развертывания проекта и разработки использовался Docker, а исходный код хранился в Github.

Выбор Java и Spring Boot обусловлен их надёжностью и производительностью в построении серверных приложений, удобством конфигурирования и управления доступа к данным. В комбинации с Kafka такая связка позволяет выстроить надёжную и масштабируемую шину событий, благодаря которой микросервисы развязаны во времени и легко масштабируются при росте нагрузки. Контейнеризация через Docker упрощает развертывание в разных окружениях и гарантирует воспроизводимость работы сервисов независимо от хостовой системы

Фронт энд на React с TypeScript обеспечивает создание отзывчивого и интерактивного интерфейса администратора с полной типизацией, что снижает количество ошибок на этапе разработки и облегчает поддержку кода

6 Ход разработки

В процессе разработки мы придерживались Agile-подхода с спринтами привязываясь к контрольным точкам, что позволило гибко реагировать на изменения требований и своевременно демонстрировать рабочие наработки. В начале каждого спринта команда собиралась на планирование, где обсуждался прогресс, формировался список задач и оценивался объём работы по каждой карточке в Github. Это позволяло быстро согласовать текущий статус задач и при необходимости перенастроить распределение усилий, а по завершении спринта мы проводили рефлексию вырабатывали улучшения для следующих итераций и уточняли критерии готовности каждого элемента функционала.

Распределение задач строилось на учёте сильных сторон участников. Каждый занимался частью проекта и инструментами, которыми был знаком лучше всего. Но взаимная работа позволяла наращивать компетенции в смежных областях. Такое чёткое распределение ролей и постоянный обмен знаниями обеспечили слаженный и эффективный ход работ.

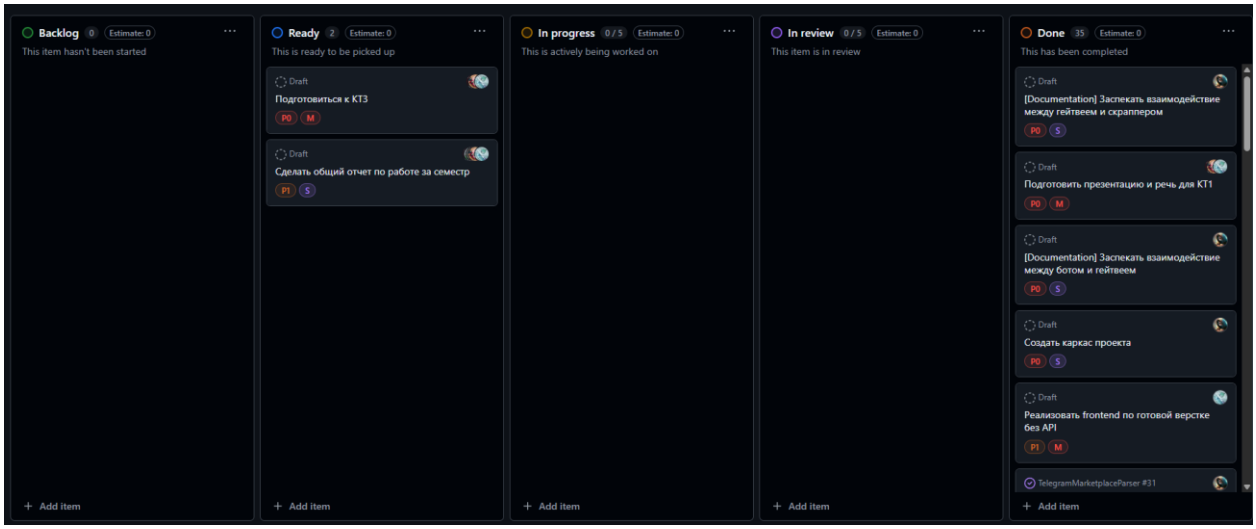

Рисунок 1 – Доска задач

7 Индивидуальный вклад участников проекта

7.1 Карсканов Никита – Бэкенд разработчик

Занимался улучшением архитектуры микросервиса Scrapper, который отвечает за получение и обработку ценовой информации. Им был проведён системный анализ текущего состояния микросервиса, выявлены недостатки - высокая связанность компонентов, низкая масштабируемость и затруднения при добавлении новых источников данных.

На основе анализа Никита переработал архитектуру сервиса, выделив слои ответственности, улучшил структуру обработки данных и систему логирования, что повысило устойчивость и гибкость сервиса. Также им был интегрирован новый маркетплейс - Aliexpress.

7.2 Ефимов Виталий – Тимлид

Разработал API-интерфейс для управления маркетплейсами, что стало ключевым элементом серверной логики. Этот API обеспечил единый способ взаимодействия с различными торговыми платформами и упростил интеграцию новых маркетплейсов в систему.

Виталий был реализован модуль настройки и управления правилами мониторинга цен, предназначенный для использования в админ-панели. Он разработал гибкий механизм создания и редактирования правил, что позволило администраторам точно настраивать условия сбора данных.

7.3 Тиунов Андрей – Спикер, Фронт энд разработчик

Отвечал за клиентскую часть админ-панели. Он спроектировал одностраничное приложение. Для навигации между разделами была внедрил систему роутинга на React Router, а для взаимодействия с серверной частью - HTTP-клиенты с централизованной обработкой запросов и ошибок: в случае неуспешного ответа об ошибке выводились всплывающие уведомления,

повышающие удобство работы пользователя. Кроме того, Андрей озвучивал и записывал видеопрезентации контрольных точек.

7.4 Козлова Александра – Дизайнер, Фронтенд разработчик

Создала подробную концепция интерфейса в Figma, где учтены принципы UI/UX для повышения интуитивности и удобства. После согласования макета выполнена адаптивная верстка на HTML и CSS с продуманной структурой компонентов и визуальной иерархией элементов управления. На основе анализа пользовательских сценариев интерфейс оптимизировался под реальные задачи, что сделало панель более логичной и простой в использовании. Аналитик и тестировщик

7.5 Кирилл Зарипов – Тестировщик, Аналитик

Занимался сбором и документированием требований, разработкой тестов и проведением функционального и регрессионного тестирования, благодаря чему выявленные дефекты оперативно исправлялись, а качество кода не падало с добавлением новых фич.

ЗАКЛЮЧЕНИЕ

В ходе работы над проектом разработанная админ-панель в полной мере соответствует требованиям заказчика. Интерфейс обеспечивает удобное управление списком маркетплейсов, правилами, а добавление нового маркетплейса (Aliexpress) подтвердило масштабируемость архитектуры. Все ключевые сценарии, заданные в бэклоге, выполнены с учётом бизнес-логики и обеспечивают необходимую гибкость настройки - от создания и редактирования правил до оперативного подключения дополнительных маркетплейсов.

Качество программного продукта подтверждено результатами многоэтапного тестирования. Интеграционные и модульные тесты выявили несколько критичных мест при взаимодействии микросервисов и парсере Scrapper: в начальной версии были проблемы с уведомлениями через промежуток времени. После доработок код стал чище и производительнее. Тем не менее были зафиксированы менее критичные дефекты: редкие таймауты при высокой нагрузке к API, долгий запуск микросервисов. Их влияние на работоспособность минимально, но требует внимания при дальнейшем масштабировании.

Для дальнейшего развития программного продукта и повышения его конкурентоспособности можно выделить несколько направлений усовершенствования.

Целесообразно реализовать систему ролевого доступа, которая позволит разграничивать права пользователей административной панели: например, выделить роли администратора, редактора и наблюдателя. Это повысит безопасность и управляемость в корпоративной среде, особенно при масштабировании количества пользователей.

Реализация поддержки импортирования и экспортирования правил в виде конфигурационных файлов (JSON/YAML), что упростит перенос и тиражирование настроек между различными инстансами системы.

Наконец, в качестве долгосрочного развития можно рассмотреть создание мобильного приложения для управления правилами и отслеживания статистики в удобном формате. Это повысит доступность системы и расширит её целевую аудиторию.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Spring Boot: Документация на русском [Электронный ресурс] / URL: https://docs.spring.io/spring-boot/docs/current/reference/html/ru/ (дата обращения: 26.05.2025)

2. Филатов, А. А. React и современная фронтенд-разработка : учеб. пособие / А. А. Филатов. – Санкт-Петербург : БХВ-Петербург, 2021. – 512 с.

3. React: Документация на русском [Электронный ресурс] / URL: https://ru.reactjs.org/ (дата обращения: 26.05.2025)

4. Грицай, Ю. В. Микросервисы: особенности разработки и эксплуатации / Ю. В. Грицай. – Москва : ДМК Пресс, 2020. – 350 с.

5. Richardson, C. Microservices Patterns: With examples in Java / C. Richardson. – Shelter Island, NY : Manning Publications, 2018. – 432 с.

6. TypeScript: Руководство на русском [Электронный ресурс] / URL: https://www.typescriptlang.org/ru/ (дата обращения: 26.05.2025)

7. Фролов, А. М., Иванова, Е. В. Разработка микросервисов на Java и Spring Boot / А. М. Фролов, Е. В. Иванова. – Москва : ЛитРес, 2022. – 352 с.

8. Docker: Документация на русском [Электронный ресурс] / URL: https://docs.docker.com/ru/ (дата обращения: 26.05.2025)

9. Apache Kafka: Введение на русском [Электронный ресурс] / URL: https://kafka.apache.org/intro (дата обращения: 26.05.2025)

10. PostgreSQL: Документация на русском [Электронный ресурс] / URL: https://postgrespro.ru/docs/postgresql/latest/ (дата обращения: 26.05.2025)