|  |  |  |
| --- | --- | --- |
|  |  |  |
| МИНОБРНАУКИ РОССИИ | | |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | |

**Институт информационных технологий (ИТ)**

**Кафедра инструментального и прикладного программного обеспечения (ИиППО)**

**КУРСОВАЯ РАБОТА**

по дисциплине: Разработка серверных частей интернет-ресурсов

по профилю: Разработка и дизайн компьютерных игр и мультимедийных приложений

направления профессиональной подготовки: 09.03.04 «Программная инженерия»

Тема: Серверная часть интернет-ресурса «Интернет-магазин»

Студент: Шило Юрий Сергеевич

Группа: ИКБО-33-22

Работа представлена к защите (дата) / Шило Ю. С. /

(подпись и ф.и.о. студента)

Руководитель: Беляев Павел Вячеславович

Работа допущена к защите\_\_\_\_\_\_\_\_\_\_\_(дата)\_\_\_\_\_\_\_\_\_/ Беляев П. В. /

(подпись и ф.и.о. рук-ля)

Оценка по итогам защиты: \_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /

(подписи, дата, ф.и.о., должность, звание, уч. степень двух преподавателей, принявших защиту)

М. РТУ МИРЭА. 2024 г.

# **ГЛОССАРИЙ**

1. **Веб-сервис** — это программное приложение, предоставляющее функциональность или данные через интернет, обычно с помощью HTTP/HTTPS протоколов, с использованием стандартов, таких как SOAP или REST.
2. **Фронтенд (Frontend)** — это часть веб-приложения, которая отвечает за взаимодействие с пользователем, включая визуальную часть интерфейса, расположенную в браузере.
3. **Бэкенд (Backend)** — это серверная часть веб-приложения, которая обрабатывает запросы от фронтенда, выполняет бизнес-логику, работает с базой данных и управляет ресурсами.
4. **JPA (Java Persistence API)** — стандарт API в языке Java, который предоставляет способы работы с базами данных и позволяет сохранять объекты Java в реляционных базах данных.
5. **Фреймворк** — это набор библиотек и инструментов, который предоставляет основу для разработки приложений, облегчая создание, поддержку и расширение программного обеспечения.
6. **Java Spring** — это фреймворк для разработки корпоративных приложений на языке Java, предоставляющий инфраструктуру для построения масштабируемых, защищённых и легко тестируемых приложений.
7. **SGI-сервер** — сервер, предоставляющий возможность обрабатывать графику, видео и мультимедийные данные, обычно используется в задачах, требующих высокой вычислительной мощности.
8. **ER-диаграмма (Entity-Relationship Diagram)** — это графическое представление сущностей и их взаимосвязей в базе данных, используемое для проектирования и анализа структуры данных.

# **СПИСОК ПРИНЯТЫХ СОКРАЩЕНИЙ**

1. HTML (HyperText Markup Language) - язык гипертекстовой разметки.
2. CSS (Cascading Style Sheets) - каскадные таблицы стилей.
3. MVC (Model-View-Controller) - модель-вид-контроллер.
4. MVT (Model-View-Template) - модель-вид-шаблон.
5. ORM (Object-Relational Mapping) - технология, которая связывает объектно-ориентированные программы с реляционными базами данных.
6. DDD (Domain-Driven Design) - Предметно-ориентированное проектирование. Набор принципов и схем, направленных на создание оптимальных систем объектов
7. SQL (Structured Query Language) — язык структурированных запросов.

# **СПИСОК ИСПОЛЬЗУЕМЫХ НОРМАТИВНЫХ ДОКУМЕНТОВ**

1. ГОСТ 2.105-95. Общие требования к текстовым документам. – М.: ИСО, 1995.
2. ГОСТ 7.0.5-2008. Система стандартов по информации, библиотечному и издательскому делу. Термины и определения. – М.: Стандартинформ, 2008.
3. ГОСТ 7.32-2017. Система стандартов по информации, библиотечному и издательскому делу. Документы научно-технические и отраслевые. Оформление курсовых работ, дипломных проектов и других документов. – М.: Стандартинформ, 2017.
4. ГОСТ 7.32-2017. Межгосударственный стандарт. Система стандартов по информации, библиотечному и издательскому делу. Отчет о научно-исследовательской работе. Структура и правила оформления (введен в действие Приказом Росстандарта от 24.10.2017 N 1494-ст)
5. ГОСТ Р 57193-2016. Национальный стандарт Российской Федерации. Системная и программная инженерия. Процессы жизненного цикла систем (введен в действие Приказом Федерального агентства по техническому регулированию и метрологии от 31 октября 2016 г. N 1538-ст)
6. ГОСТ 7.1-2003. Система стандартов по информации, библиотечному и издательскому делу. Библиографическая запись

# **СОДЕРЖАНИЕ**

[**ГЛОССАРИЙ** 2](#_Toc184754070)

[**СПИСОК ПРИНЯТЫХ СОКРАЩЕНИЙ** 3](#_Toc184754071)

[**СПИСОК ИСПОЛЬЗУЕМЫХ НОРМАТИВНЫХ ДОКУМЕНТОВ** 4](#_Toc184754072)

[**СОДЕРЖАНИЕ** 5](#_Toc184754073)

[**ВВЕДЕНИЕ** 7](#_Toc184754074)

[**ОСНОВНАЯ ЧАСТЬ** 8](#_Toc184754075)

[**1.** **Анализ предметной области** 8](#_Toc184754076)

[**1.1.** **Основной функционал готовых программных решений** 8](#_Toc184754077)

[**1.2.** **Требования к серверной части интернет-ресурса** 8](#_Toc184754078)

[**1.3.** **Сравнительный анализ существующих решений** 9](#_Toc184754079)

[**1.4.** **Сравнительный анализ существующих решений** 11](#_Toc184754080)

[**1.5.** **Вывод** 11](#_Toc184754081)

[**2.** **Выбор и обоснование технологий** 12](#_Toc184754082)

[**2.1.** **Анализ возможных технологий** 12](#_Toc184754083)

[**2.2.** **Обоснование и выбор технологий** 14](#_Toc184754084)

[**2.3.** **Вывод** 15](#_Toc184754085)

[**3.** **Разработка архитектуры на основе MVC паттерна** 17](#_Toc184754086)

[**3.1.** **Применение паттерна MVC** 18](#_Toc184754087)

[**3.2.** **Архитектура приложения** 19](#_Toc184754088)

[**3.3.** **Диаграмма архитектуры** 20](#_Toc184754089)

[**3.4.** **Реализация ключевых функций на основе MVC** 21](#_Toc184754090)

[**3.5.** **Вывод** 22](#_Toc184754091)

[**4.** **Разработка серверной части интернет-ресурса** 23](#_Toc184754092)

[**4.1.** **Реализация моделей** 23](#_Toc184754093)

[**4.2.** **Реализация контроллеров** 23](#_Toc184754094)

[**4.3.** **Получившееся приложение** 24](#_Toc184754095)

[**4.4.** **Тестирование** 34](#_Toc184754096)

[**4.5.** **Вывод** 35](#_Toc184754097)

[**ЗАКЛЮЧЕНИЕ** 36](#_Toc184754098)

[**СПИСОК ЛИТЕРАТУРЫ** 37](#_Toc184754099)

[**ПРИЛОЖЕНИЕ** 38](#_Toc184754100)

# **ВВЕДЕНИЕ**

Современные технологии значительно изменили способы ведения бизнеса, и интернет-магазины стали неотъемлемой частью коммерческой сферы. В последние десятилетие развитие интернет-технологий привело к созданию множество онлайн платформ для продажи товаров и услуг, а правильно функционирование таких магазинов не возможно без надежной и масштабируемой серверной части. Бэкенд интернет-магазина играет ключевую роль в обеспечении стабильности, безопасности, быстродействия и удобства взаимодействия с пользователем.

Данная курсовая работа посвящена разработке и анализу архитектуры бэкенд-части интернет-магазина, а также созданию функционального веб-приложения, которое будет обеспечивать надежную работу платформы и простое взаимодействие с пользователем.

**Цель:** заключается в разработке серверной части интернет-ресурса, предназначенного для функционирования интернет-магазина

**Актуальность:** обусловлена ростом популярности интернет-коммерции и необходимостью создания удобных, эффективных и безопасных платформ для онлайн-продаж.

**Объект исследования:** это серверная часть интернет-магазина, включающая в себя базу данных.

**Предмет исследования:** архитектура бэкенд-части интернет-магазина, а также методы обеспечения безопасности, производительности и масштабируемости системы.

**Предполагаемый результат:** создание функционального веб-приложения интернет-магазин, включающий в себя интерфейсы для добавления товаров, отображения чужих товаров, редактирования своего профиля и просмотра чужих.

# **ОСНОВНАЯ ЧАСТЬ**

1. **Анализ предметной области**

В данной главе проводится анализ существующих решений в области интернет-магазинов, рассматриваются их ключевые функциональные возможности, выявляются требования, предъявляемые к таким системам, а также определяются преимущества и недостатки, которые могут быть использованы для разработки конкурентного программного продукта.

* 1. **Основной функционал готовых программных решений**

Готовые программные решения для интернет-магазинов представляют широкий спектр функциональных решений, которые позволяют быстро создавать и легко работать с площадкой для купли-продажи товаров.

Независимо от программного продукта основные функциональные возможности включают в себя:

Управление каталогом товаров:

* Добавление, редактирование и удаление товара;
* Управления изображениями, описанием и характеристиками товара.

Пользовательский интерфейс:

* Регистрация и авторизация пользователя;
* Управление своим профилем;
* Личный кабинет пользователя для просмотра заказов и сохранения товара;
* Поддержка нескольких ролей (администратор, клиент).

Инструменты для маркетинга и продвижения:

* Управление популярностью товара;
* Поддержка рейтинга товара.

Безопасность:

* Управление правами доступа (роли пользователей);
* Защита от CSRF-атак.
  1. **Требования к серверной части интернет-ресурса**

На практике система интернет-магазина должны соответствовать следующим требованиям:

* быть простыми в использовании для логистов и сотрудников складов, с интуитивно понятным интерфейсом и минимальными требованиями к обучению;
* поддерживать функции разделения пользователей на основе их роли для работы в веб-приложении;
* предоставлять возможность пользователю добавлять свои товары и просматривать чужие;
* предоставлять возможность изменять пользовательские данные (фото профиля, имя и другую информацию);
* обеспечивать быструю и надежную обработку запросов пользователя.
  1. **Сравнительный анализ существующих решений**

Сравним несколько популярных решений с точки зрения их функциональных возможностей, преимуществ и недостатков (см. Таблица 1). Для анализа были выбраны следующие интернет-магазины:

1. Amazon – крупнейший интернет-магазин в мире. Он предлагает широкий ассортимент товаров, включая электронику книги, одежду, товары для дома и многое другое. Amazon работает на глобальном рынке, предоставляя услуги доставки во множество стран;
2. AliExpress – глобальная интернет платформа для розничной торговли. Данная платформа ориентирована на международную аудиторию, предлагая миллионы товаров от китайских производителей и продавцом по очень низким ценам;
3. OZON – один из крупнейших интернет-магазинов России. Компания предлагает широкий ассортимент товаров, включающий книги, электронику, бытовую технику, одежду и даже продукты питания. В отличии от международных гигантов, OZON активно развивает собственные логистические мощности, предоставляя гибкие способы доставки и оплаты.

*Таблица 1 - Сравнительный анализ существующих решений*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Интернет-магазин** | **Ассортимент** | **Удобство использования** | **Отзывы и рейтинг** | **Платформы** |
| ***AMAZON*** | Огромный выбор товаров: электроника, одежда, книги и др. | Удобный интерфейс, расширенный поиск, персонализация. | Полезная система отзывов и рейтингов. | Веб-сайт, приложения для iOS и Android. |
| ***AliExpress*** | Преобладание недорогих товаров от китайских производителей. | Простая навигация, но иногда сложно найти качественные товары. | Рейтинги товаров не всегда объективны. | Веб-сайт, приложения для iOS и Android. |
| ***OZON*** | Широкий ассортимент, в том числе локальные товары. | Удобный поиск, персонализированные рекомендации. | Удобная система отзывов, часто пишут реальные покупатели. | Веб-сайт, приложения для iOS и Android. |
| ***Разработанный проект*** | Выбор стикеров и наклеек | Удобный поиск по названию стикера или никнейму пользователя, простой и понятный интерфейс. | Сортировка товаров по популярности за разный промежуток времени. | Веб-сайт |

* 1. **Сравнительный анализ существующих решений**

Основное отличие разрабатываемой системы заключается в её функционале, ориентированном на логистов-инженеров. Сервис сочетает удобства ручного ввода, быстрого изменения и фильтрации данных. Она предоставляет:

* гибкий и интуитивно понятный интерфейс для пользователя, позволяющий просматривать товары других пользователей;
* доступность системы как для небольших интернет-магазинов, так и для крупных компаний благодаря её масштабируемости;
* быстродействие системы.
  1. **Вывод**

Анализ существующих решений в области интернет-магазинов, таких как Amazon, AliExpress и Ozon, показал, что каждая из платформ имеет свои сильные стороны, но также и ограничения. Amazon выделяется высоким качеством сервиса и скоростью доставки, однако требует платной подписки для премиум-услуг. AliExpress привлекает низкими ценами и разнообразием товаров, но страдает от долгих сроков доставки и возможных проблем с качеством товаров. Ozon, в свою очередь, ориентирован на российский рынок и отличается удобством локальной доставки и разнообразием акций, но может быть менее конкурентоспособным по ценам на международной арене.

Анализ выявил несколько ключевых направлений для улучшения и развития интернет-магазинов:

* оптимизация интерфейса и удобство использования, с акцентом на улучшение навигации;
* гибкость в обработке данных;
* обогащение ассортимента.

1. **Выбор и обоснование технологий**

В данной главе проводится анализ технологий и паттернов проектирования, подходящих для разработки интернет-магазина. Обоснование выбора технологий основано на их функциональных возможностях, совместимости и эффективности для реализации поставленных целей.

* 1. **Анализ возможных технологий**

При разработке интернет-магазина важно учитывать такие аспекты, как безопасность, производительность, масштабируемость и удобство разработки. Важно также использовать такие инструменты, которые обеспечат высокую доступность и поддержку расширений в будущем.

На основе анализа предметной области и требований к системе выделены следующие ключевые компоненты, которые необходимо реализовать:

* фронтенд, интуитивно понятный интерфейс для инженеров-логистов;
* бэкенд, серверная часть для обработки запросов и выдачи ответов;
* база данных, хранение информации о посылках;
* архитектура, схема взаимодействия пользователя и базы данных.

**Фронтенд**:

* HTML и CSS используются для создания структуры и стилизации интерфейса, эти технологии универсальны, хорошо поддерживаются всеми браузерами и обеспечивают быструю разработку;
* JavaScript добавляет интерактивность и динамичность в интерфейс;
* Bootstrap — Фреймворк, использующий HTML, CSS и JavaScript для быстрого создания адаптивных и стильных веб-страниц. Он поможет ускорить процесс разработки и сделать интерфейс магазина удобным и красивым на различных устройствах.

**Бэкенд**:

* Java: Это один из самых популярных языков программирования, известный своей надежностью и производительностью. Java широко используется для разработки крупных приложений, обеспечивая хорошую поддержку многозадачности и многопоточности, что критично для интернет-магазинов с большим числом пользователей.
* Spring Framework (Java Spring): Это мощный и гибкий фреймворк для Java, который упрощает разработку и развертывание веб-приложений. Он включает такие компоненты, как Spring Boot (для быстрой настройки и запуска приложения) и Spring Security (для обеспечения безопасности), что делает его идеальным для создания сложных и масштабируемых приложений.

**База данных:**

* MySQL: реляционная база данных, которая используется для хранения структурированных данных. MySQL поддерживает сложные запросы, транзакции и хорошо подходит для типичных задач интернет-магазина, таких как хранение информации о товарах, пользователях, заказах и транзакциях. Она также обладает хорошей производительностью и поддерживает масштабирование, что делает ее подходящей для большинства интернет-магазинов.
* Redis: это in-memory база данных, которая используется для кэширования данных и временного хранения. Redis хранит данные в оперативной памяти, что значительно ускоряет обработку запросов и повышает производительность системы. В интернет-магазинах Redis можно использовать для кэширования информации о товарах, сессиях пользователей, корзинах покупок и других часто запрашиваемых данных, что уменьшает нагрузку на основную реляционную базу данных и ускоряет время отклика.
* PostgreSQL: это мощная реляционная СУБД, известная своей высокой надежностью и поддержкой сложных запросов. PostgreSQL также поддерживает расширенные типы данных, индексы и транзакции, что делает его подходящим для работы с большими объемами данных. Он также предоставляет более гибкие возможности для масштабирования по сравнению с MySQL и может быть полезен для крупных интернет-магазинов, которым требуется высокая надежность и обработка сложных данных.

**Архитектура:**

* MVC (Model-View-Controller): это паттерн проектирования, который разделяет приложение на три основные части:
  + Model (Модель): отвечает за бизнес-логику и взаимодействие с базой данных.
  + View (Представление): отображает данные пользователю и предоставляет интерфейс.
  + Controller (Контроллер): управляет взаимодействием между моделью и представлением, обрабатывает пользовательские запросы и изменяет данные в модели.
* MVT (Model-View-Template): этот паттерн схож с MVC, но отличается тем, что контроллер в MVT заменен на серверную логику. Также MVT включает в себя шаблоны (templates), которые отвечают за представление данных, а также встроенную поддержку маршрутизации и ORM (Object-Relational Mapping). В рамках MVT взаимодействие с базой данных и маршрутизация запросов осуществляется через серверную логику, что упрощает разработку.
* DDD (Domain-Driven Design): это паттерн проектирования, ориентированный на моделирование бизнес-логики, который помогает создавать сложные системы, разделяя их на области (domains). В DDD особое внимание уделяется взаимодействию между разработчиками и бизнес-экспертами для правильного понимания требований и эффективного управления проектом.
  1. **Обоснование и выбор технологий**

Для реализации веб-приложения интернет-магазин были выбраны следующие технологии:

1. **Java** — это высокопроизводительный язык, который используется для разработки многозадачных, надежных и безопасных приложений. Она предоставляет богатый набор библиотек и фреймворков, что позволяет решать различные задачи, от работы с базами данных до интеграции с внешними сервисами.
2. **Java Spring** предоставляет большой набор инструментов и фреймворков для работы с различными аспектами веб-разработки, такими как безопасность, работа с базой данных, обработка запросов и многое другое. Использование Spring позволяет быстро разрабатывать и масштабировать приложения. Spring Boot, в частности, значительно ускоряет процесс разработки благодаря автоматической настройке и упрощению конфигурации.
3. **HTML/CSS**: эти технологии необходимы для создания основного макета и стилизации веб-страниц. HTML обеспечивает структуру страницы, а CSS — визуальное оформление.
4. **JavaScript**: используется для добавления интерактивности на страницы. В интернет-магазине JavaScript помогает создать динамичные элементы интерфейса, такие как карусели, фильтры и корзины.
5. **Bootstrap**: с помощью Bootstrap можно быстро и качественно создать адаптивный интерфейс, который будет одинаково хорошо выглядеть на различных устройствах. Этот фреймворк также содержит готовые компоненты (кнопки, формы, модальные окна и т. д.), которые ускоряют процесс разработки.
6. **MySQL**: это одна из самых популярных и широко используемых реляционных баз данных. MySQL имеет хорошую поддержку транзакций и возможность масштабирования, что важно для интернет-магазина, в котором необходимо эффективно хранить и обрабатывать большие объемы данных о товарах, пользователях и заказах. MySQL также хорошо интегрируется с Java и Spring, что упрощает работу с базой данных.
   1. **Вывод**

Выбор технологий для разработки интернет-магазина основывается на их функциональных возможностях, совместимости и эффективности. Java и Spring Framework обеспечивают надежность, производительность и масштабируемость серверной части приложения, а HTML, CSS и JavaScript, в сочетании с Bootstrap, дают возможность создавать удобный и адаптивный интерфейс. MySQL идеально подходит для хранения данных, обеспечивая быстродействие и поддержку транзакций.

Таким образом, выбранный стек технологий (Java, Spring, HTML, CSS, JavaScript, Bootstrap, MySQL) является оптимальным для разработки современного и эффективного интернет-магазина, обеспечивая как высокую производительность, так и хорошее взаимодействие между клиентом и сервером.

1. **Разработка архитектуры на основе MVC паттерна**

В данном разделе рассматривается применение паттерна проектирования MVC для разработки архитектуры системы интернет-магазина. Паттерн MVC обеспечивает структурированность кода, разделение ответственности между компонентами системы и упрощает процесс её разработки и поддержки.

Для построения архитектуры веб-приложения определяются основные бизнес-правила и требования, которые формируют структуру системы. На рисунке 1 представлена диаграмма вариантов использования, демонстрирующая ключевые сценарии взаимодействия пользователей с системой интернет-магазина.

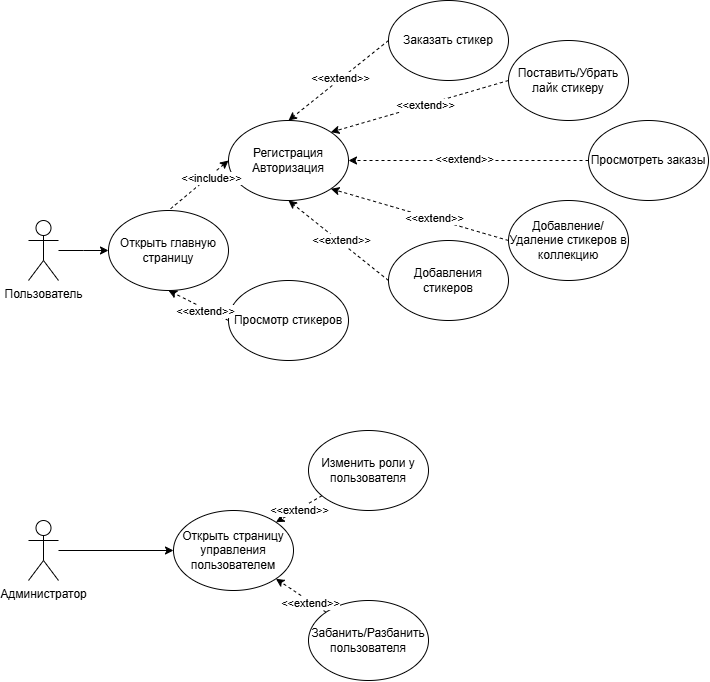


Рисунок 1 – Диаграмма вариантов использования интернет-магазина для пользователя

На рисунке 2 представлена диаграмма вариантов использования, демонстрирующая ключевые сценарии взаимодействия администратора с пользователями, зарегистрированными в интернет-магазине.

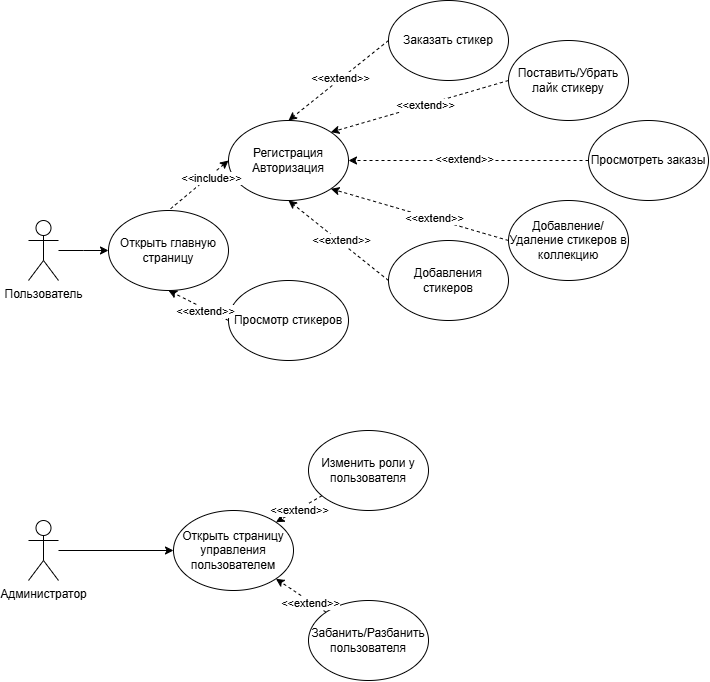


Рисунок 2 – Диаграмма вариантов использования интернет-магазина для администратора

* 1. **Применение паттерна MVC**

Паттерн MVC разделяет архитектуру приложения на три основные части, представленные на рисунке 3.

1. **Модель**: отвечает за работу с данными, включая взаимодействие с базой данных (MySQL), обработку бизнес-логики и управление состоянием. Например, в интернет-магазине модель будет содержать классы для управления данными о товарах, заказах и пользователях.
2. **Интерфейс**: обеспечивает отображение информации пользователю с помощью HTML и CSS, создавая интерфейсы для просмотра товаров, оформления заказов, редактирования данных и других взаимодействий.
3. **Контроллер**: обрабатывает пользовательские запросы, связывая модель и интерфейс. Контроллер отвечает за такие действия, как добавление пользовательских стикеров, отображения пользовательских стикеров, и добавление товаров в заказ, обновления данных пользователя.

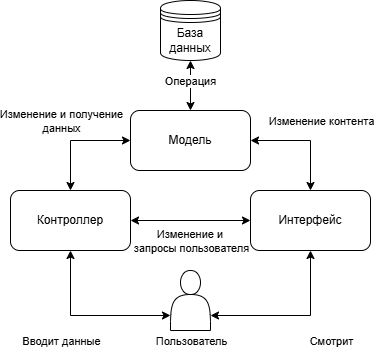


Рисунок 3 – Схема работы MVC-паттерна

* 1. **Архитектура приложения**

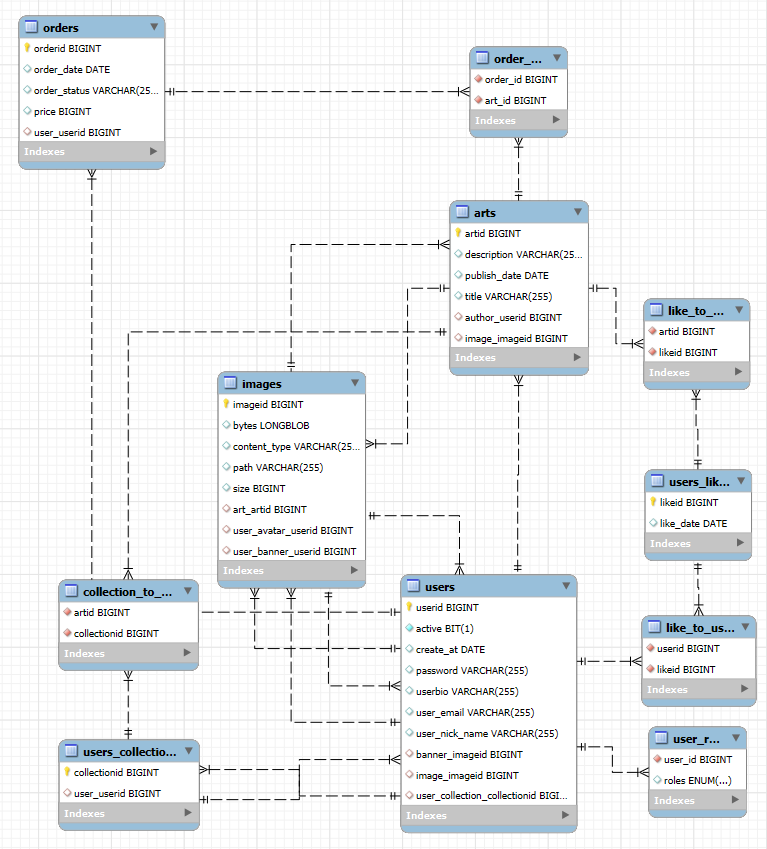
Рассмотрим основные компоненты системы интернет-магазина:

1. Файлы представлений (HTML/CSS): формируют пользовательский интерфейс. Эти файлы отвечают за отображение товаров, корзины, оформления заказов и других элементов, взаимодействующих с пользователем.
2. Классы моделей: управляют взаимодействием с базой данных (MySQL). Взаимодействие осуществляется через ORM или нативные SQL-запросы для обработки данных о товарах, заказах, пользователях и платежах.
3. Классы контроллеров: реализуют бизнес-логику, управляя процессами обработки заказов, обновления статусов и отправки уведомлений.
4. Ниже указан примерный сценарий для лучшего понимания архитектуры:

В системе интернет-магазина взаимодействие компонентов происходит следующим образом: пользователь отправляет запрос через веб-интерфейс, например, чтобы оформить добавить новый стикер в коллекцию сайта. Контроллер, реализованный с помощью Spring Boot, принимает запрос и вызывает соответствующий метод модели. Модель взаимодействует с базой данных MySQL для извлечения или обновления данных о стикерах. Результат передается обратно в контроллер, который обрабатывает данные и направляет их в представление. Представление формирует ответ в виде веб-страницы и возвращает его пользователю. Такой подход обеспечивает высокую производительность, структурированность кода и удобство использования для пользователей интернет-магазина.

* 1. **Диаграмма архитектуры**

На рисунке 4-5 представлены ER-диаграмма базы данных и диаграмма классов для созданного в результате выполнения курсовой работы веб-приложения на тему интернет-магазин.



Рисноук 4 – ER-диаграмма базы данных в mySQL

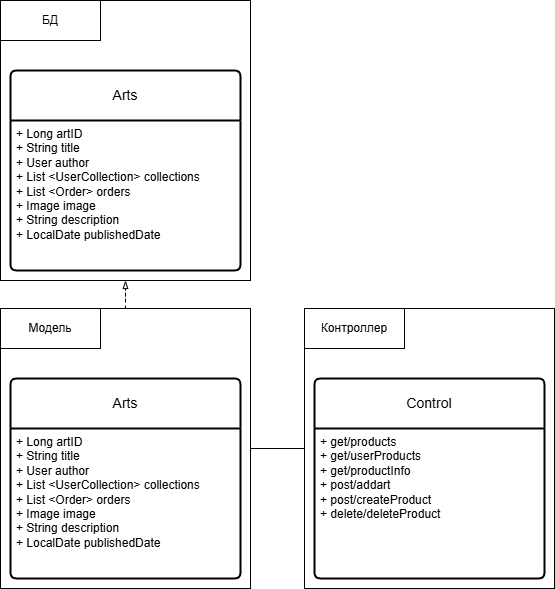


Рисунок 5 - Диаграмма классов для одной из моделей

* 1. **Реализация ключевых функций на основе MVC**
     1. **Добавление лайка:**
* Контроллер: обрабатывает запросы на добавление лайка для определенной записи.
* Модель: управляет таблицей лайков в MySQL, добавляя новый лайк для указанного объекта.
* Представление: отображает обновленную таблицу лайков, добавляя только что поставленный лайк к записи.
  + 1. **Срез по лайкам по датам:**
* Контроллер: обрабатывает запросы на показ лайков, поставленных в определенном диапазоне дат.
* Модель: выполняет запрос к базе данных MySQL, извлекая лайки, поставленные в заданном интервале.
* Представление: отображает таблицу лайков, отфильтрованную по датам, с результатами для выбранного диапазона.
  + 1. **Поиск записей по имени:**
* Контроллер: обрабатывает запросы на поиск записей по имени пользователя или другим параметрам.
* Модель: выполняет запрос к базе данных MySQL, чтобы найти записи, соответствующие введенному имени или поисковому запросу.
* Представление: отображает таблицу записей, соответствующих запросу, с результатами поиска по имени.
  + 1. **Изменение пользовательской информации:**
* Контроллер: обрабатывает запросы на изменение записей.
* Модель: выполняет запрос к базе данных MySQL, чтобы найти записи, соответствующие текущему пользователю, и изменяет их.
* Представление: отображает измененные данные.
  + 1. **Добавление новых стикеров:**
* Контроллер: обрабатывает запросы добавление стикера.
* Модель: выполняет запрос к базе данных MySQL, чтобы добавить новую запись.
* Представление: отображает добавленные стикеры.
  1. **Вывод**

В данной главе была рассмотрена разработанная архитектура системы интернет-магазина с применением паттерна MVC. Такое решение обеспечивает логическое разделение компонентов приложения на уровне представления, обработки данных и бизнес-логики, что способствует упрощению разработки, тестирования и поддержки системы.

1. **Разработка серверной части интернет-ресурса**

Разработка серверной части системы интернет-магазина основана на использовании языка программирования Java и фреймворка Java Spring с применением паттерна MVC. В данной главе подробно рассматриваются ключевые аспекты реализации серверной части: обработка запросов, взаимодействие с базой данных MySQL и управление бизнес-логикой. Также уделяется внимание тестированию системы и обеспечению её стабильной работы. Полный исходный код приложения представлен в приложении к отчету.

* 1. **Реализация моделей**

Все модели были реализованные с помощью Spring Data JPA. Spring Data JPA обеспечивает простое и удобное взаимодействие с таблицами базы данных, используя аннотации для определения сущностей и их связей. Также он автоматически создает скрипты для создания баз данных на языке MySQL. Пример реализации модели для коллекций пользователя предоставлена на рисунке 5. Реализацию же всех моделей предоставлена в листинге в пункте “Файлы реализации моделей” (см. Листинг 10-16).



Рисунок 6 – Реализация одной из моделей проекта

* 1. **Реализация контроллеров**

Контроллеры обрабатывают запросы и вызывают соответствующие методы моделей для выполнения операций. Пример контроллера для пользовательских коллекций в реализованном нами интернет-магазине показан на рисунке 6. Реализацию же всех контролеров предоставлен в листинге в пункте “Файлы реализации моделей” (см. Листинг 10-16).

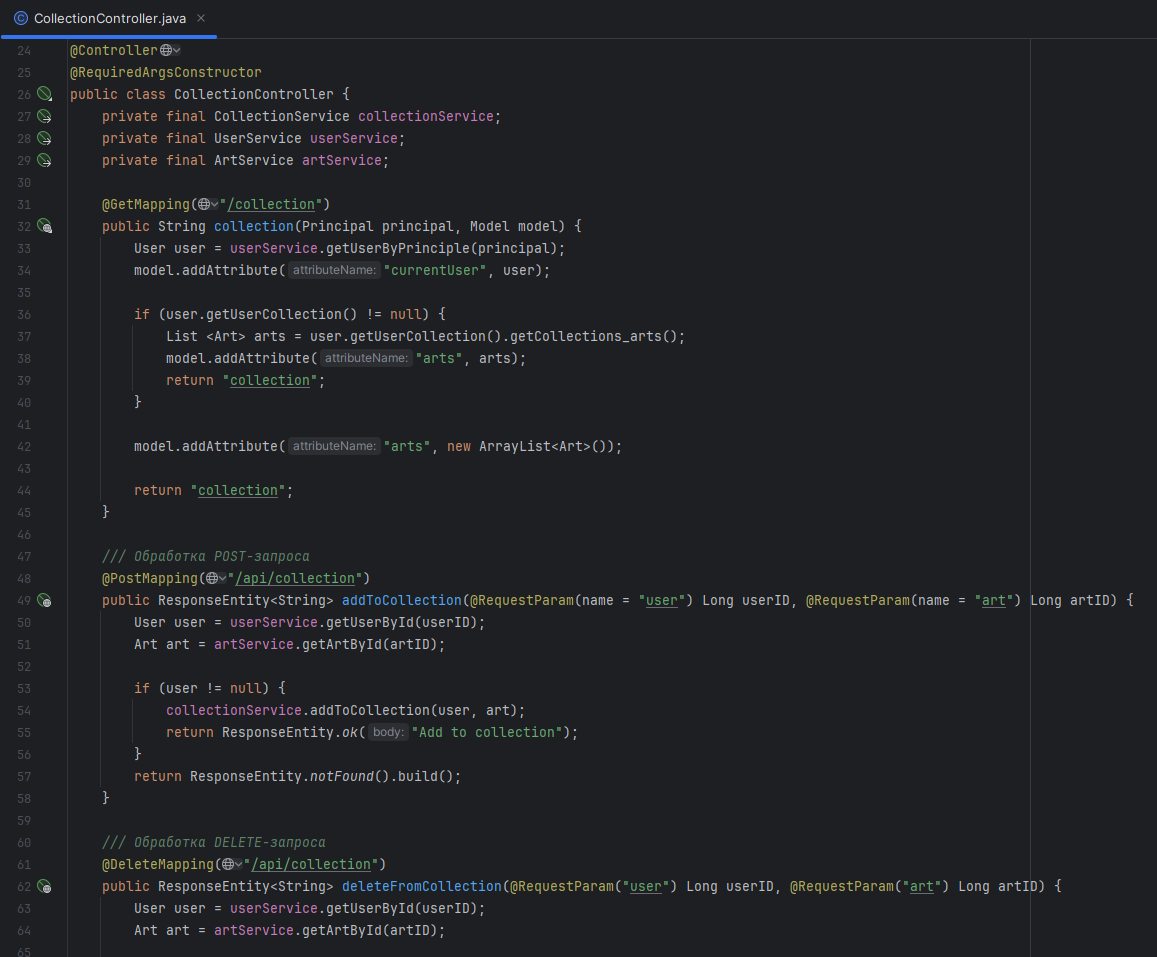


Рисунок 7 – Реализация одного из контролеров нашего проекта

* 1. **Получившееся приложение**

Далее рассмотрим опыт использования веб-приложения.

На рисунках 8-26 будет показаны часть реализованных интерфейсов нашего интернет-магазина.

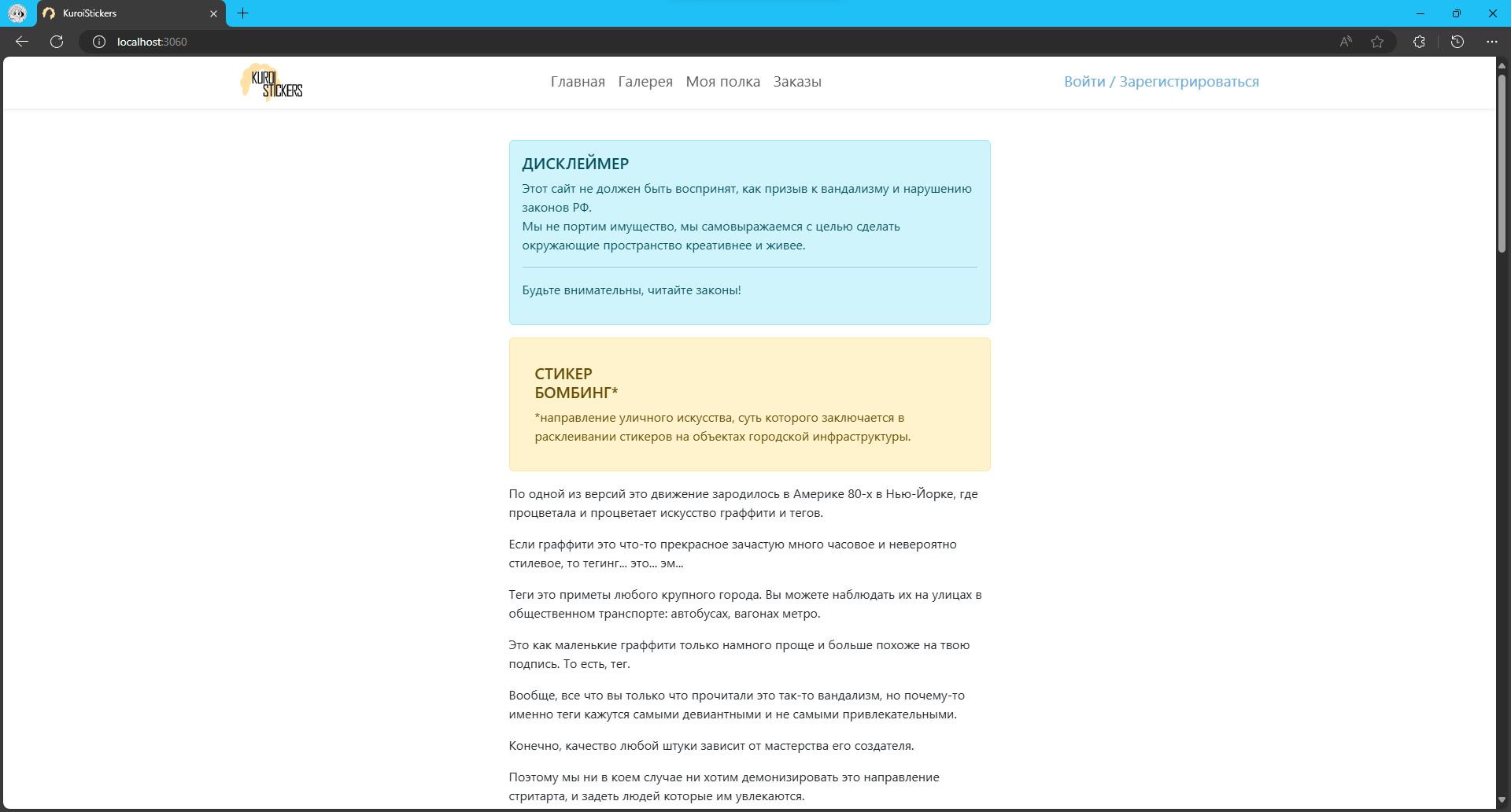
\

Рисунок 8 – Интерфейс главной страницы

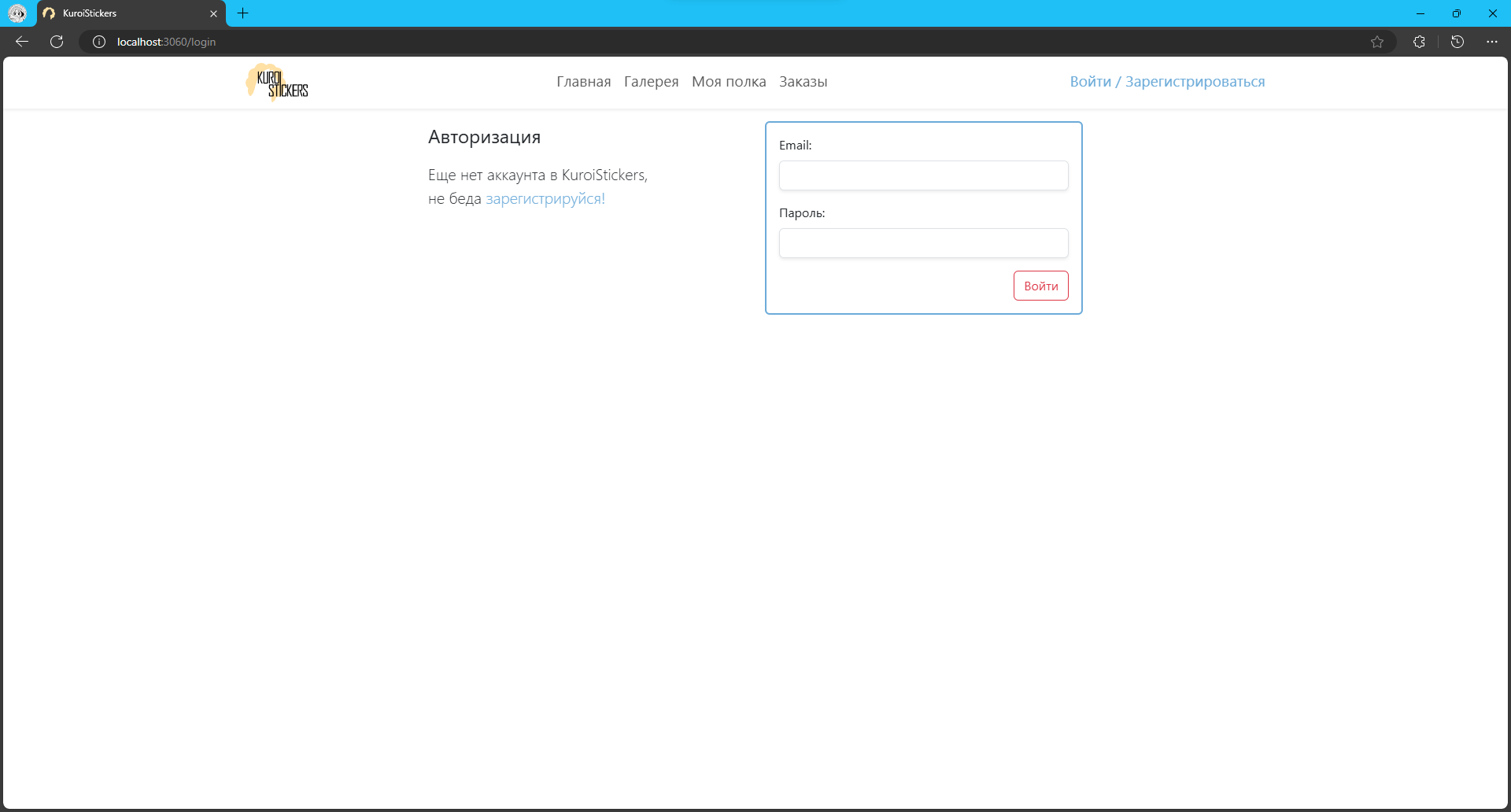


Рисунок 9 – Интерфейс страницы входа

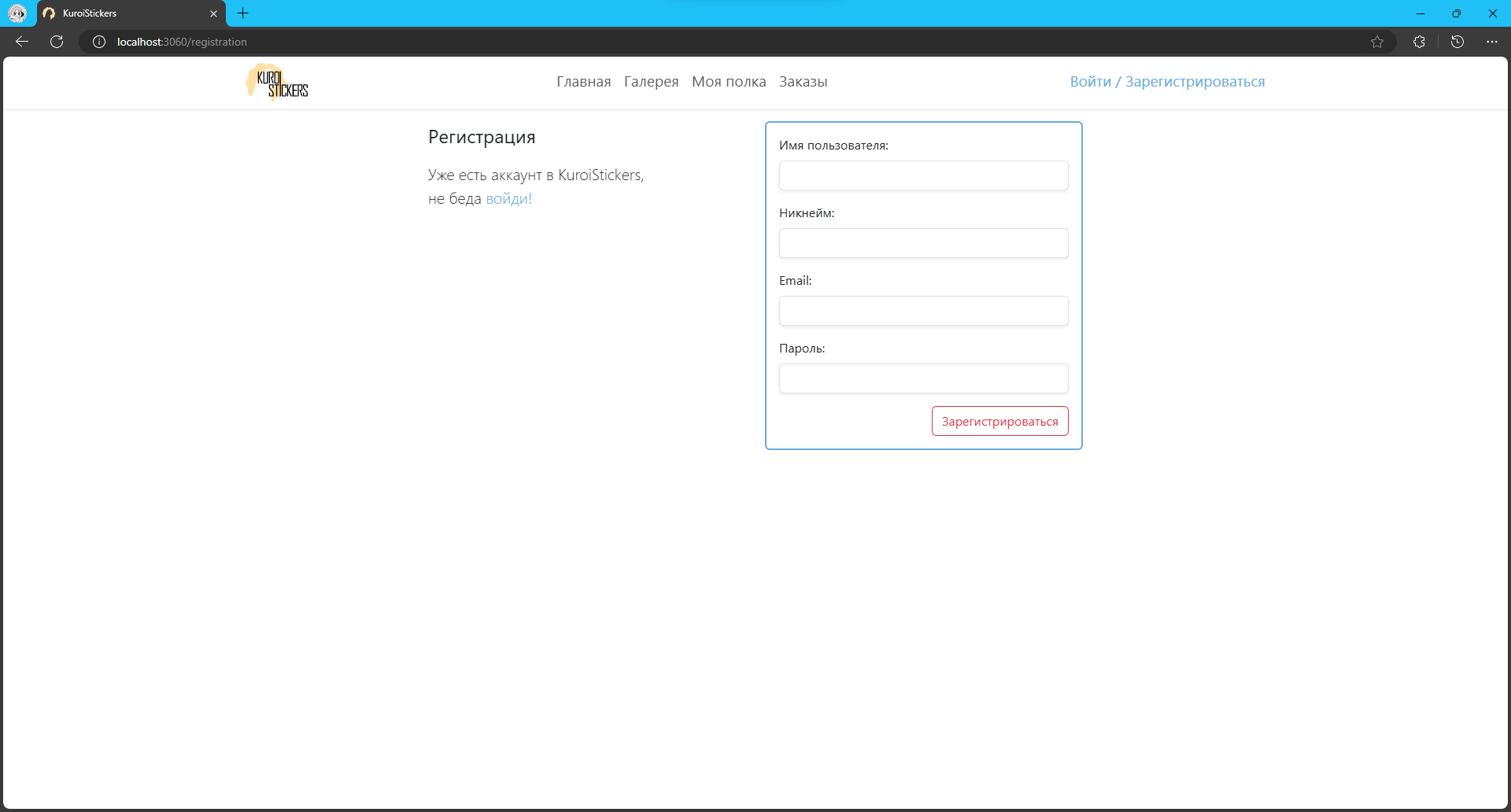


Рисунок 10 – Интерфейс страницы регистрации

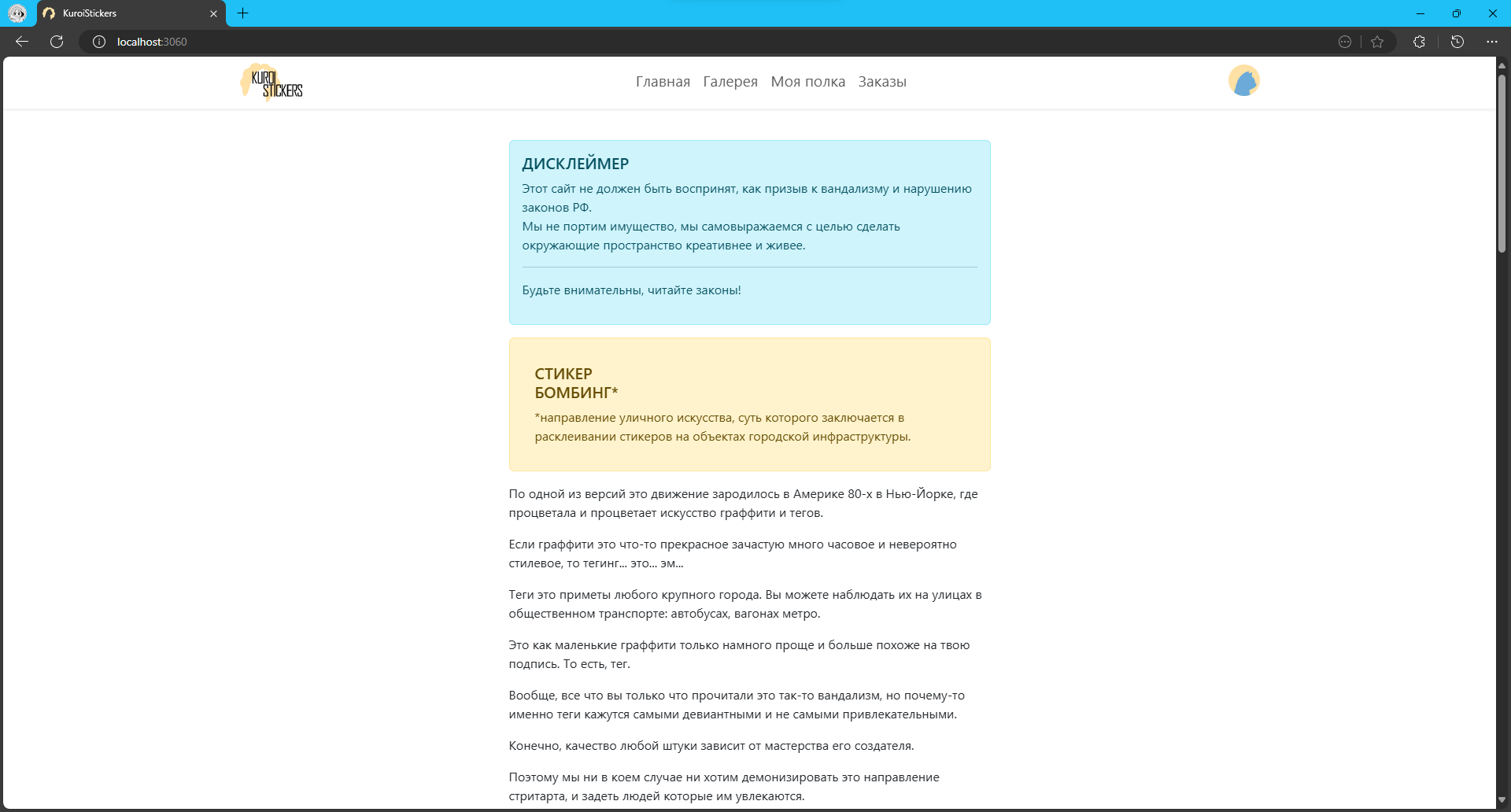


Рисунок 11 – Интерфейс главной страницы после входа

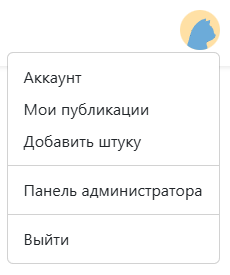


Рисунок 12 – Выпадающие меню для администратора

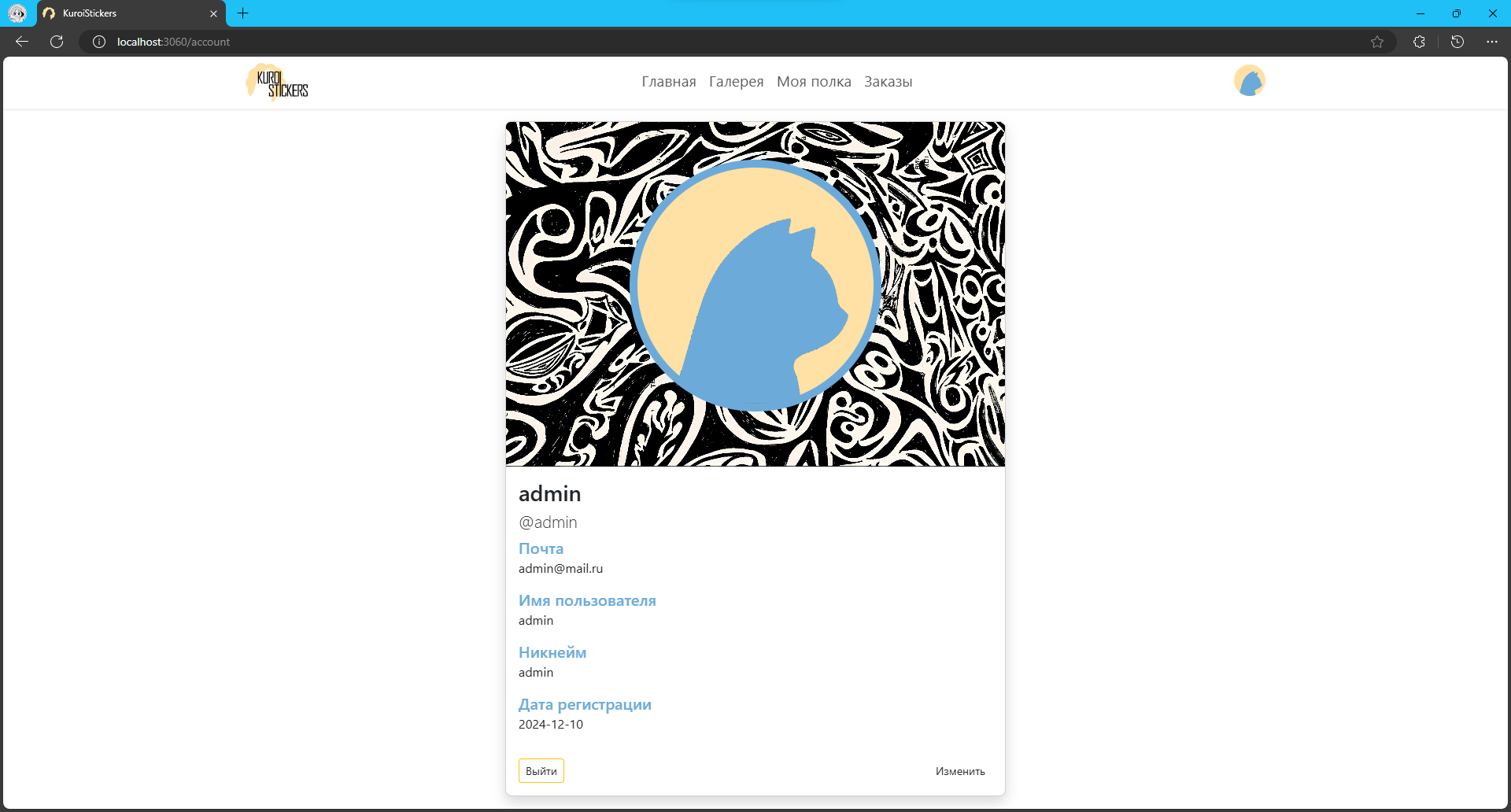


Рисунок 13 – Интерфейс профиля пользователя

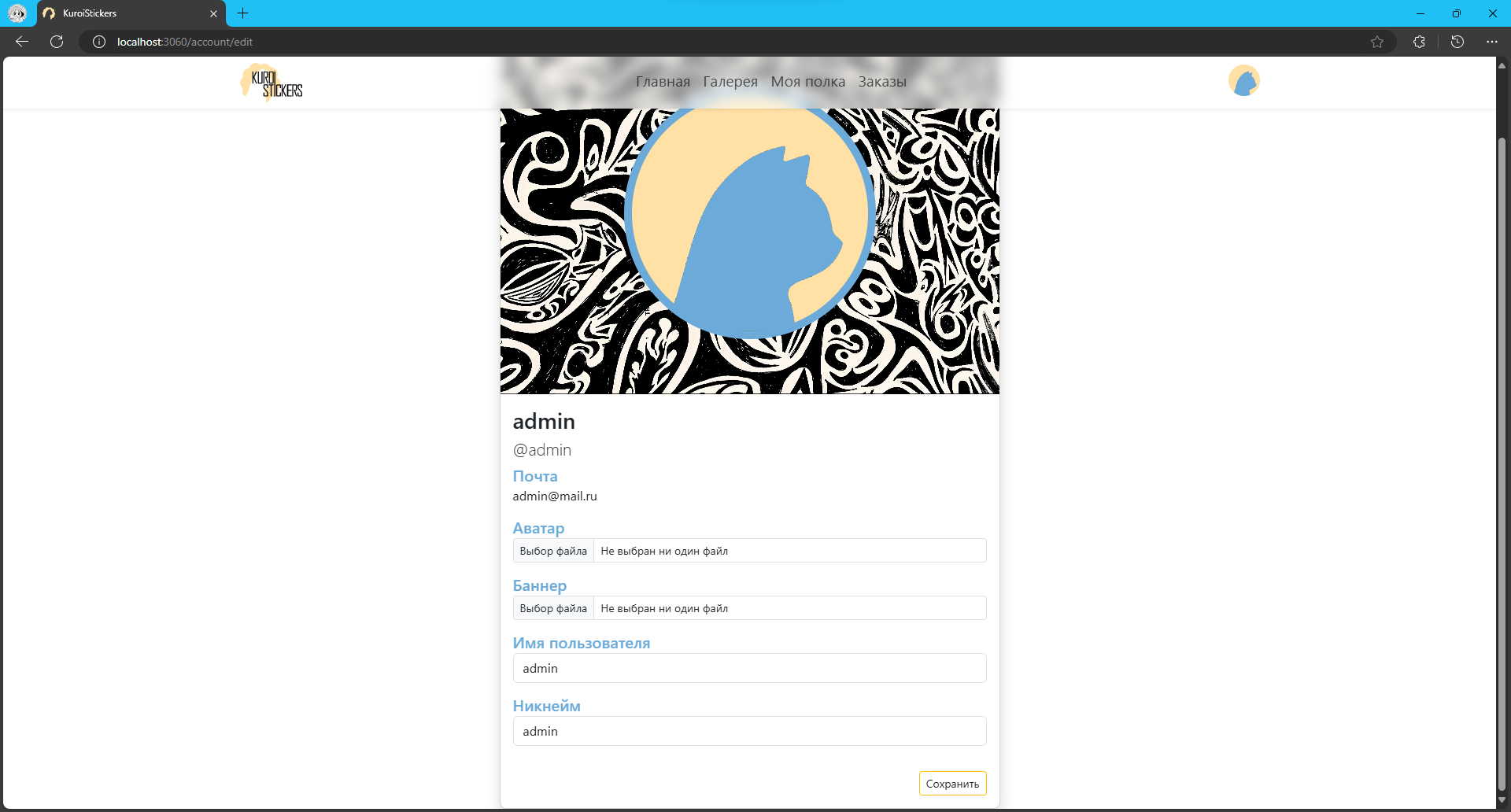


Рисунок 14 – Интерфейс для изменения информации о пользователе

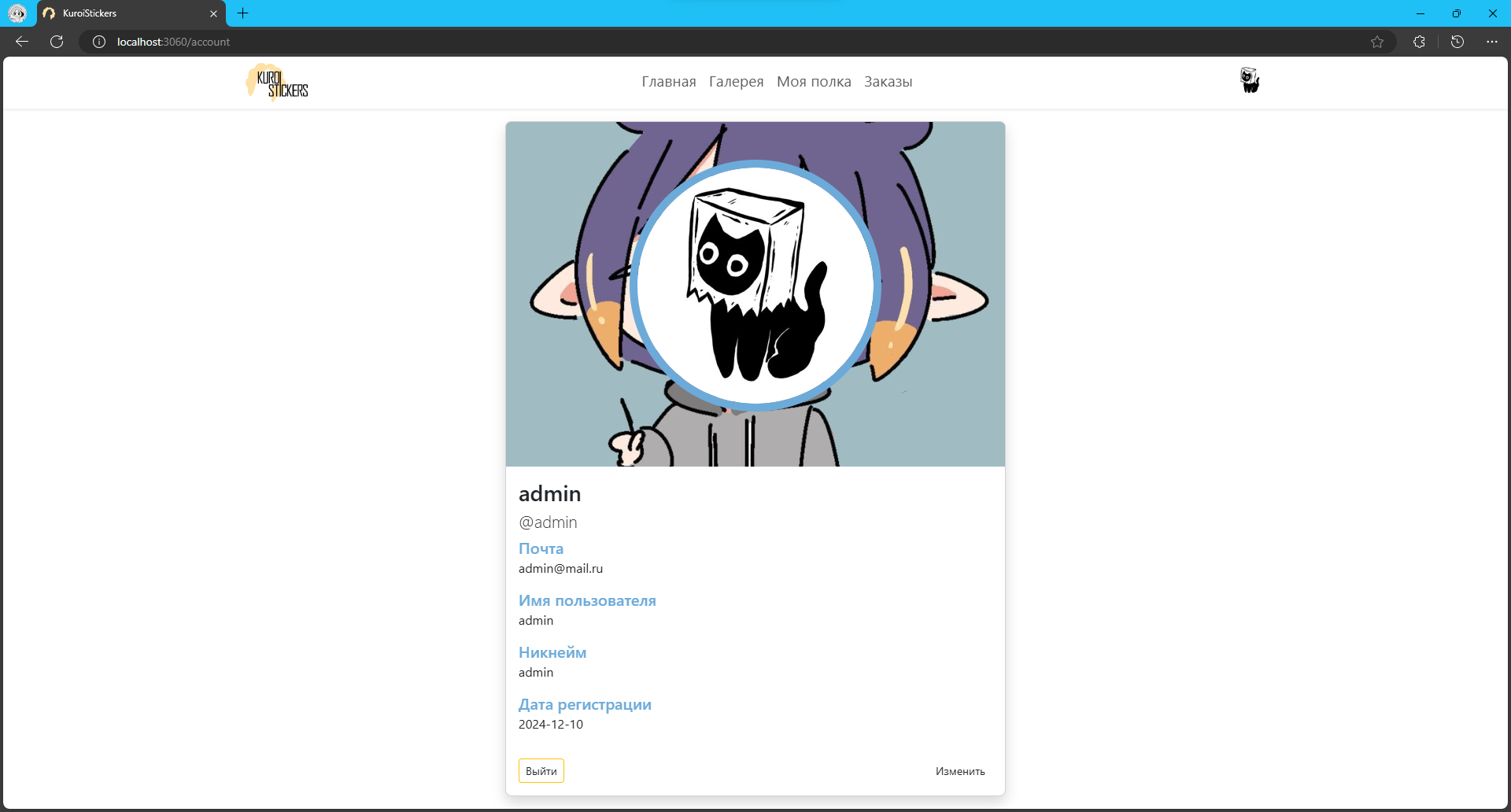


Рисунок 15 – Интерфейс профиля пользователя после изменения

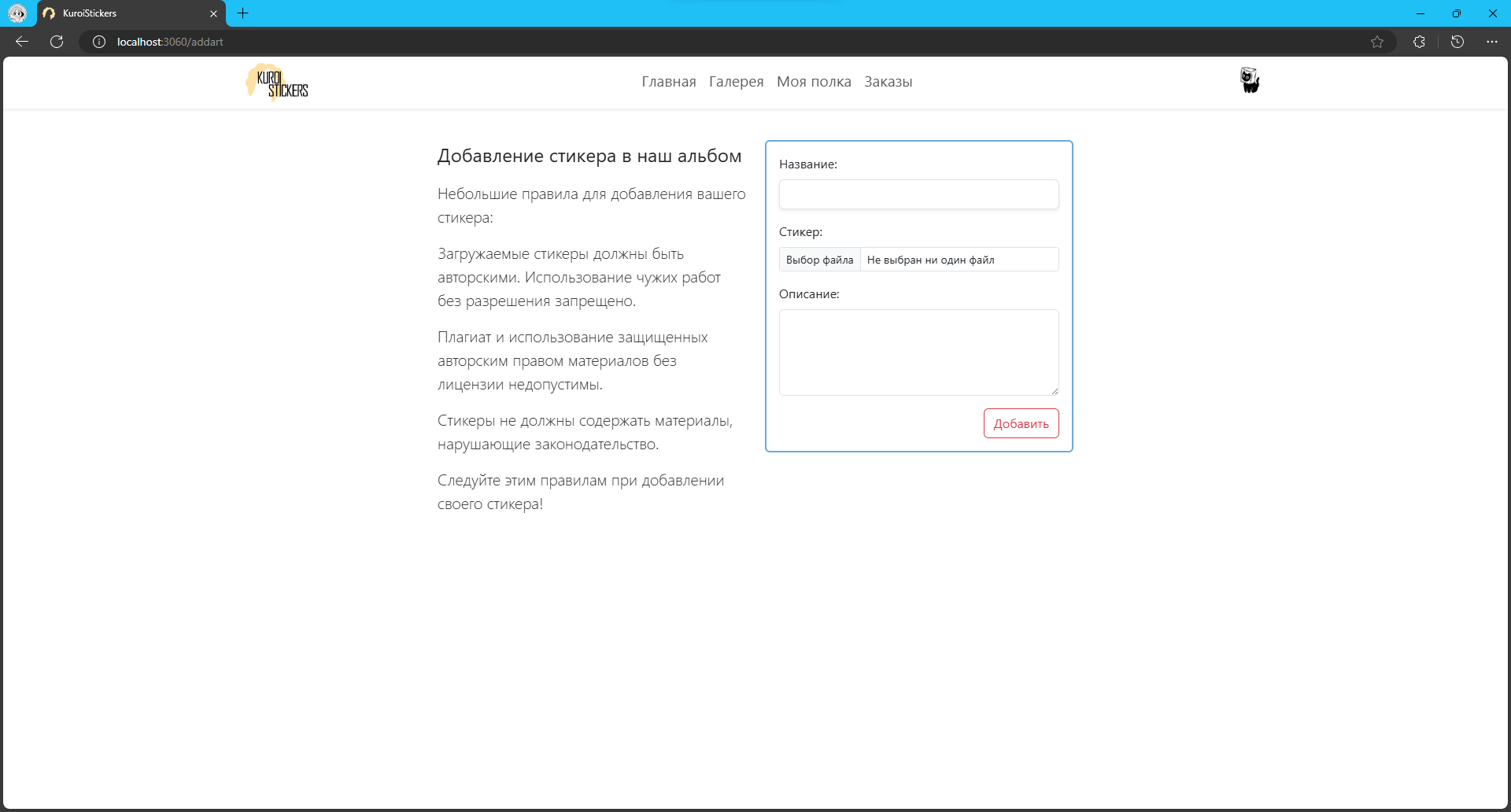


Рисунок 16 – Интерфейс для добавления стикера на сайт

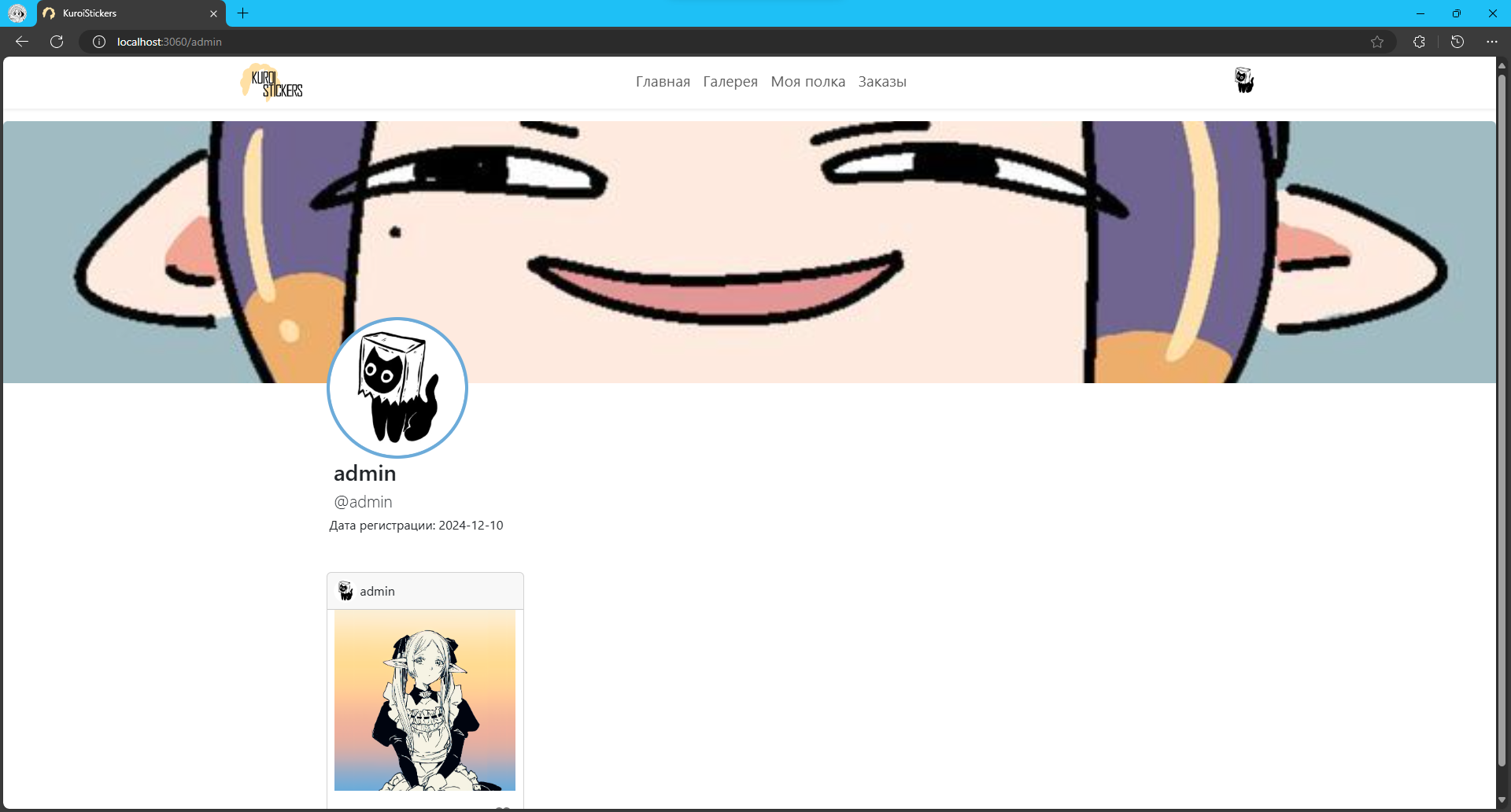


Рисунок 17 – Интерфейс для отображения стикеров добавленных пользователем

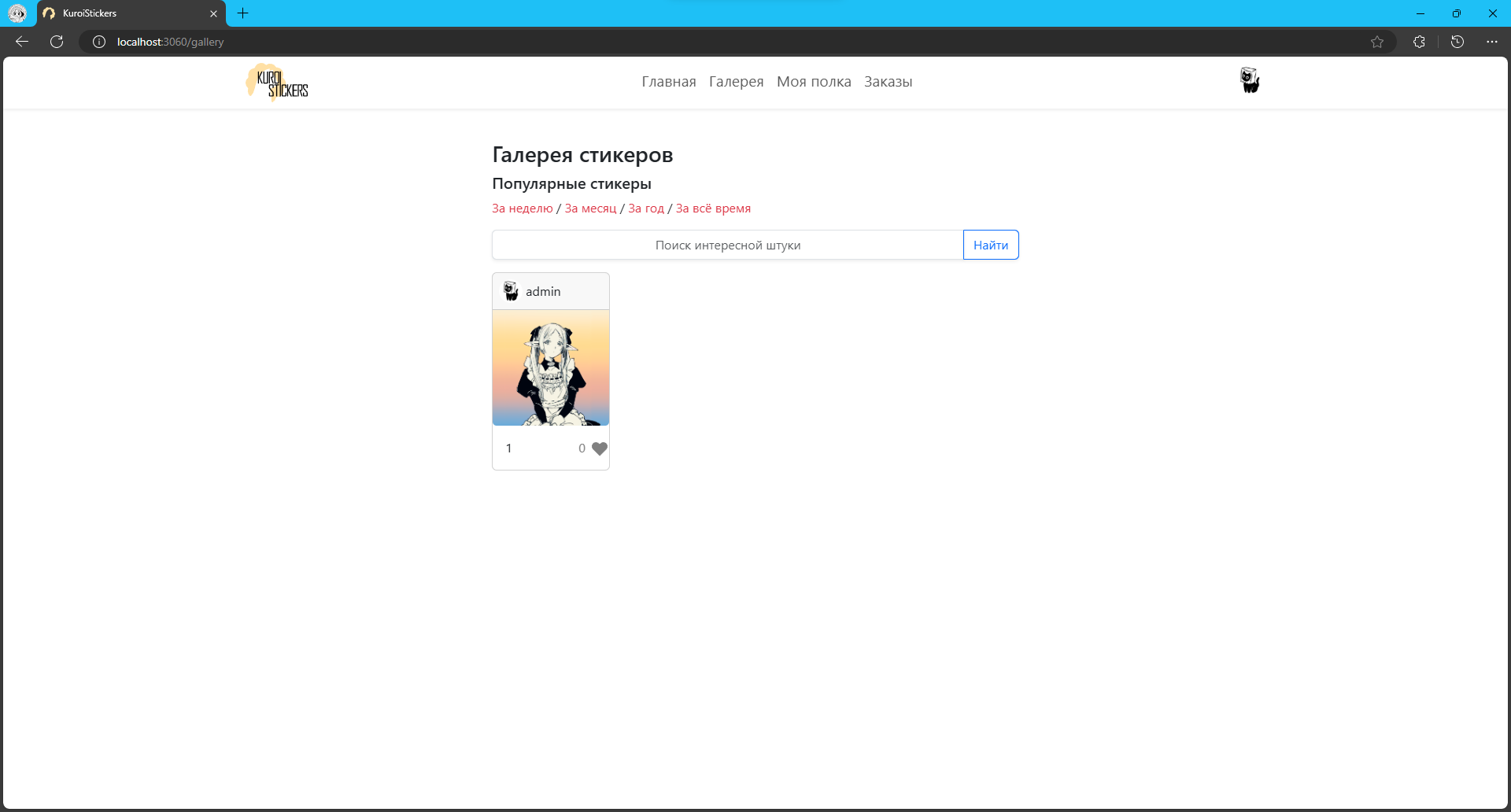


Рисунок 18 – Интерфейс галереи для отображения всех стикеров присутствующих на сайте

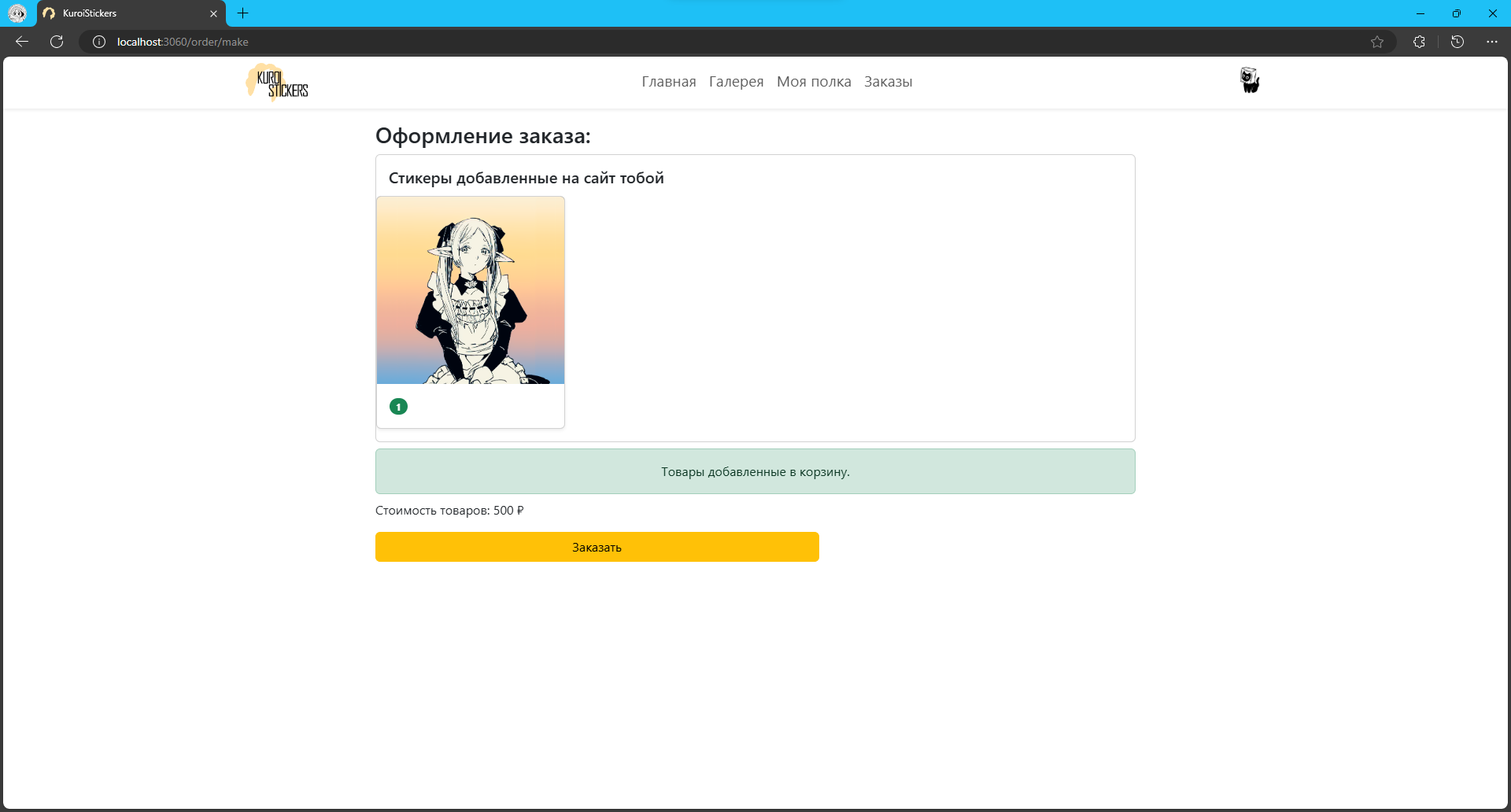


Рисунок 19 – Интерфейс для оформления заказа

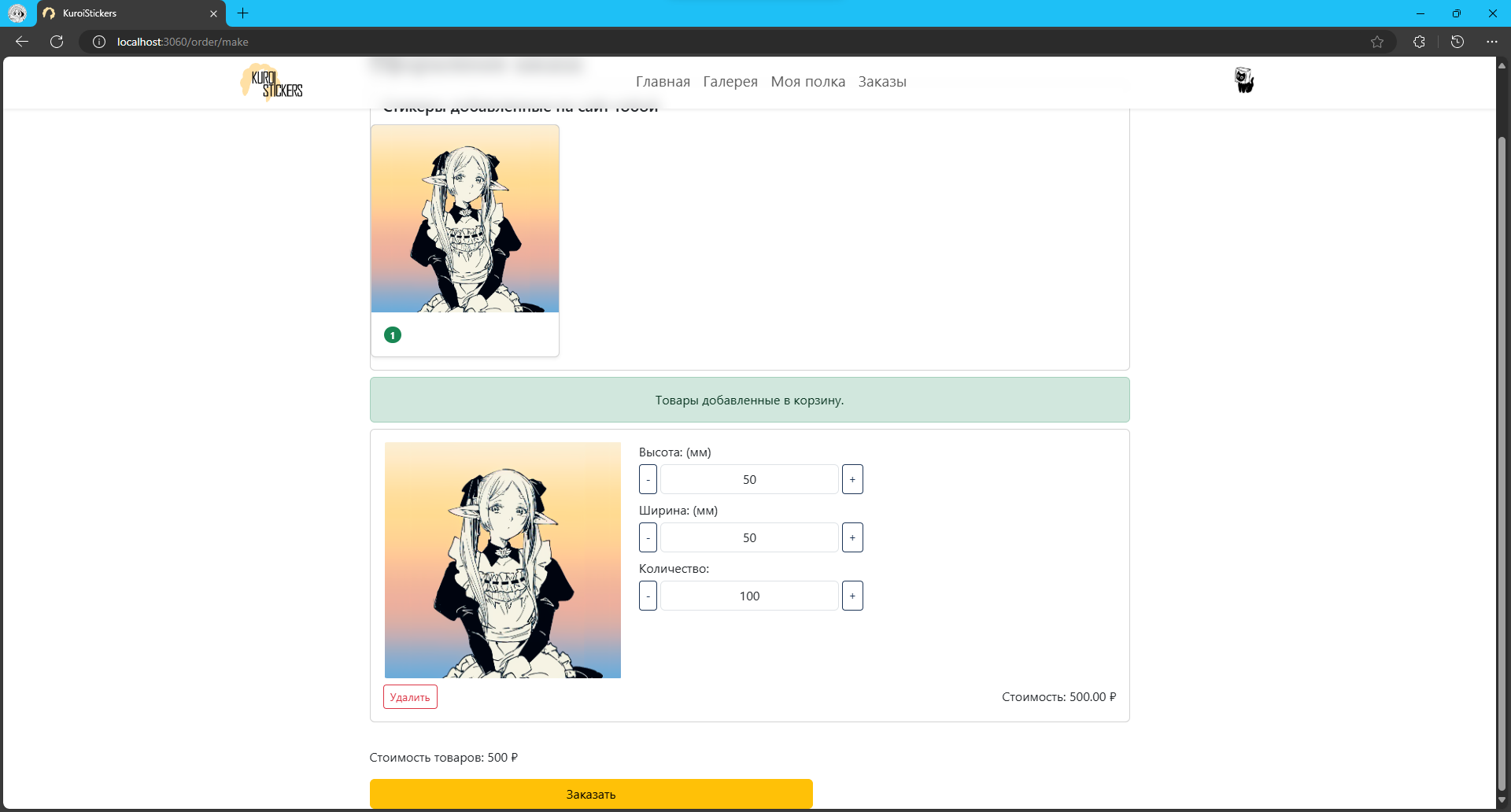


Рисунок 20 – Интерфейс для изменения количества и размера стикера

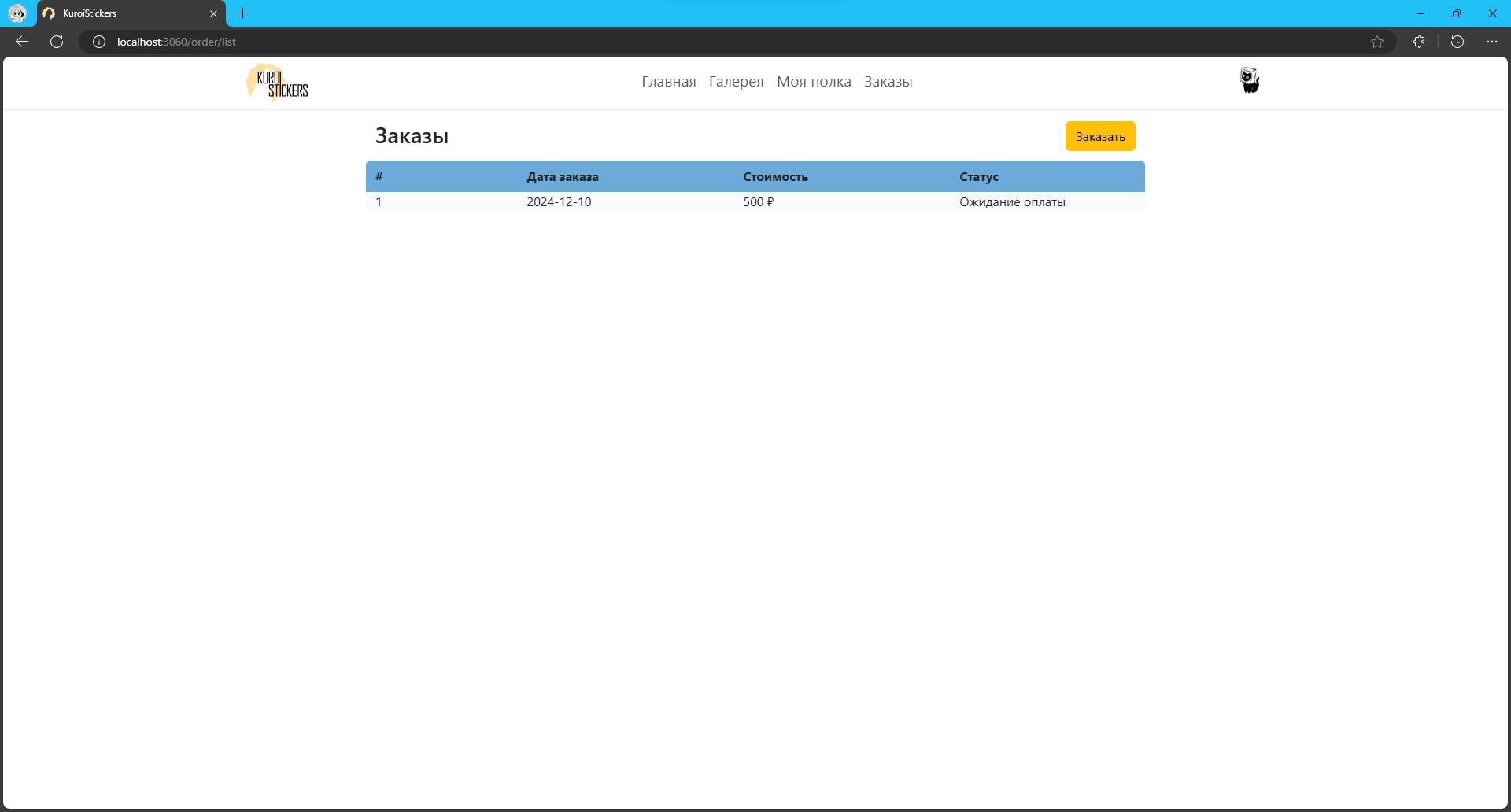


Рисунок 21 – Интерфейс, в котором отображаются все заказы пользователя

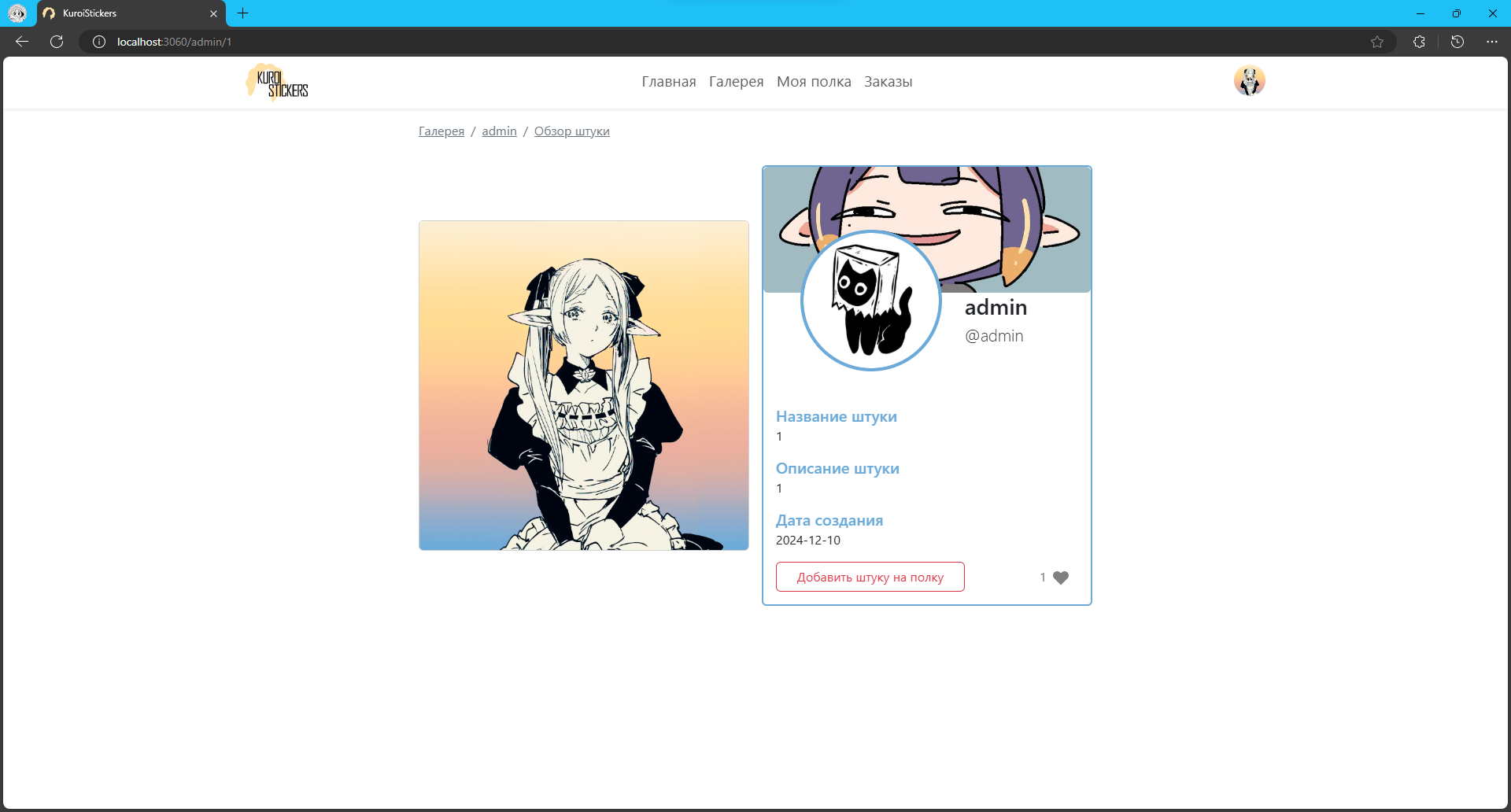


Рисунок 22 – Интерфейс для просмотра информации о стикере

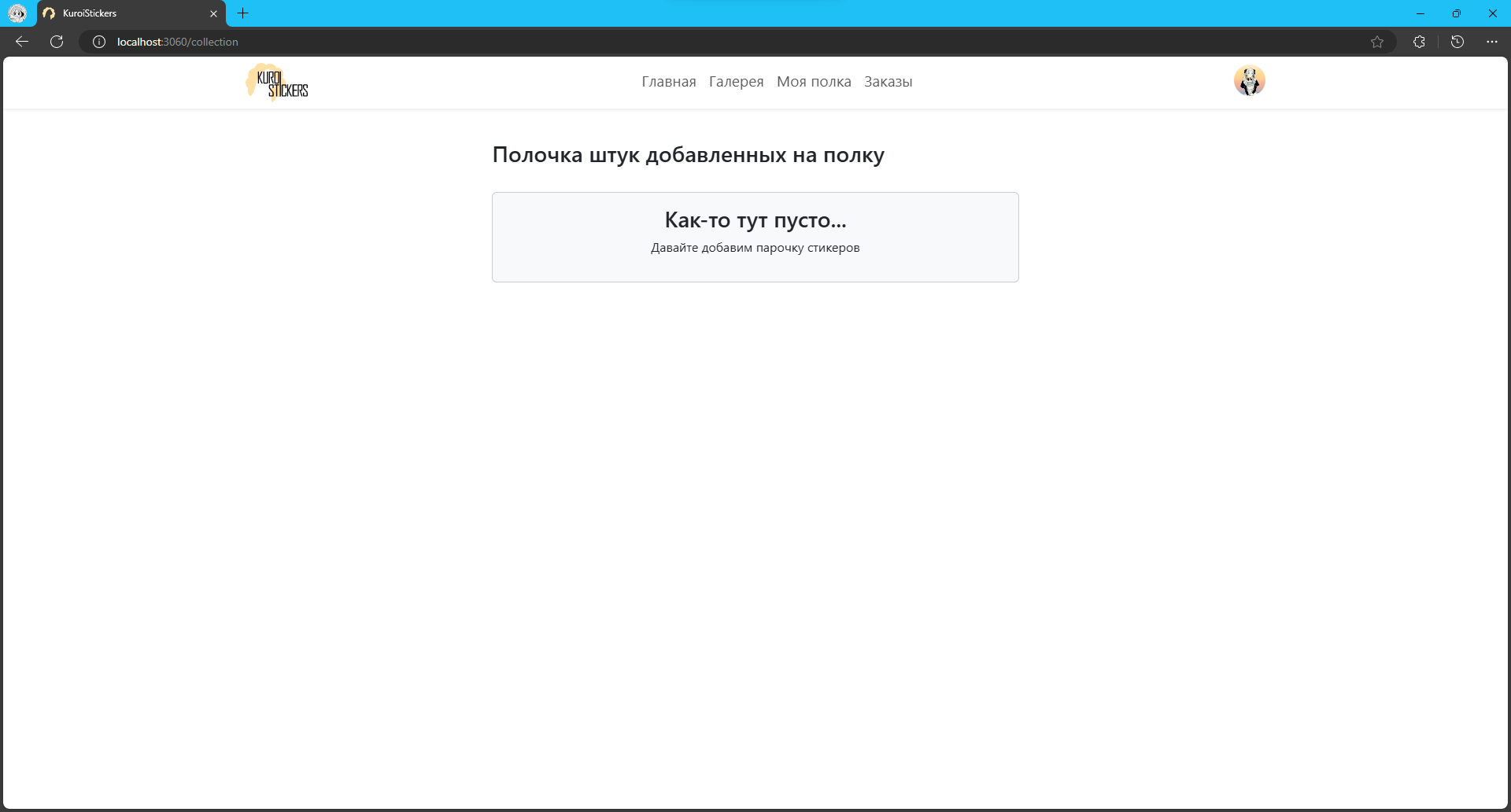


Рисунок 23 – Интерфейс, в котором отображаются все стикеры, которые пользователь добавил к себе на полку

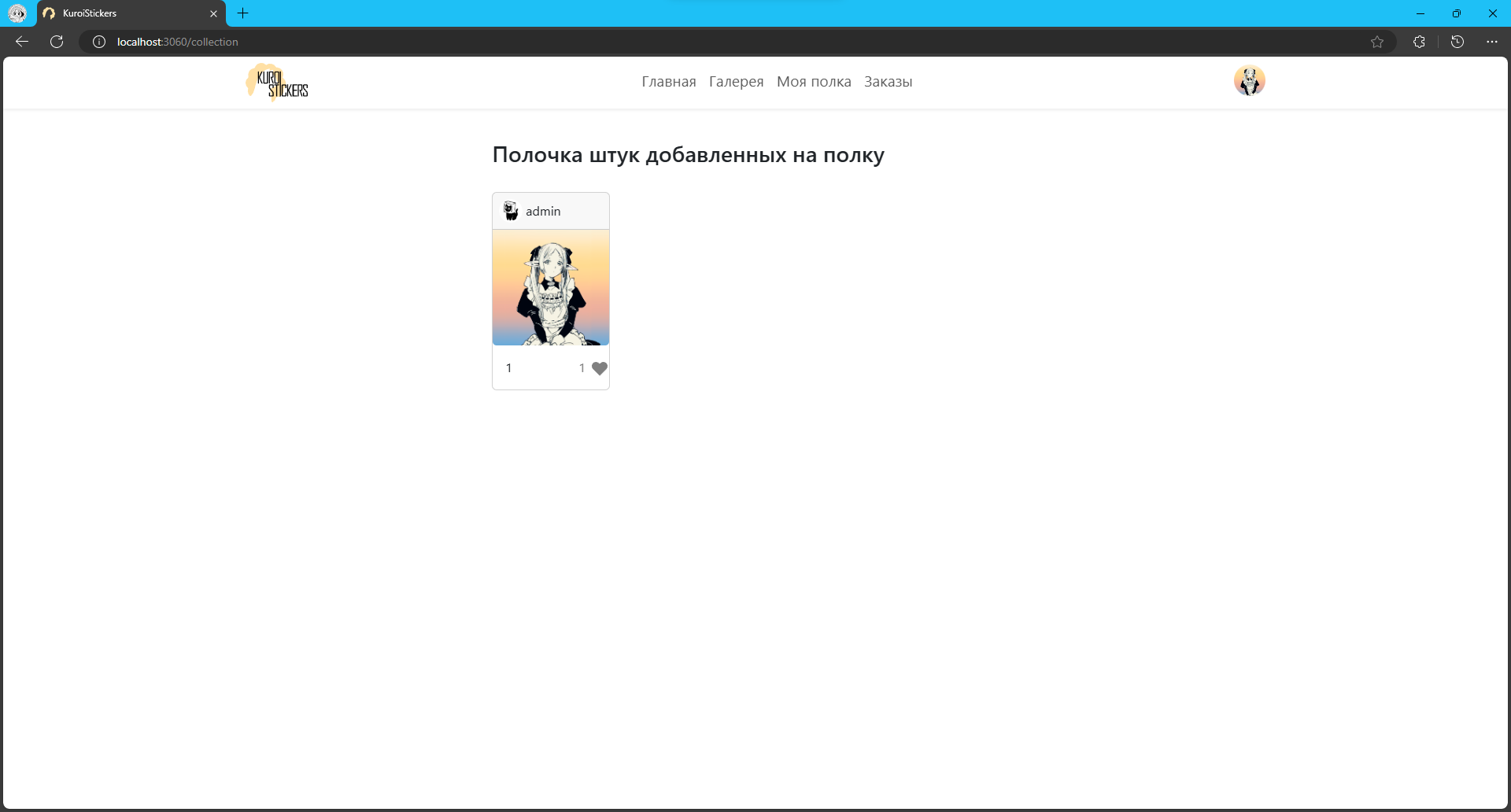


Рисунок 24 – Интерфейс полки после добавления на него стикера

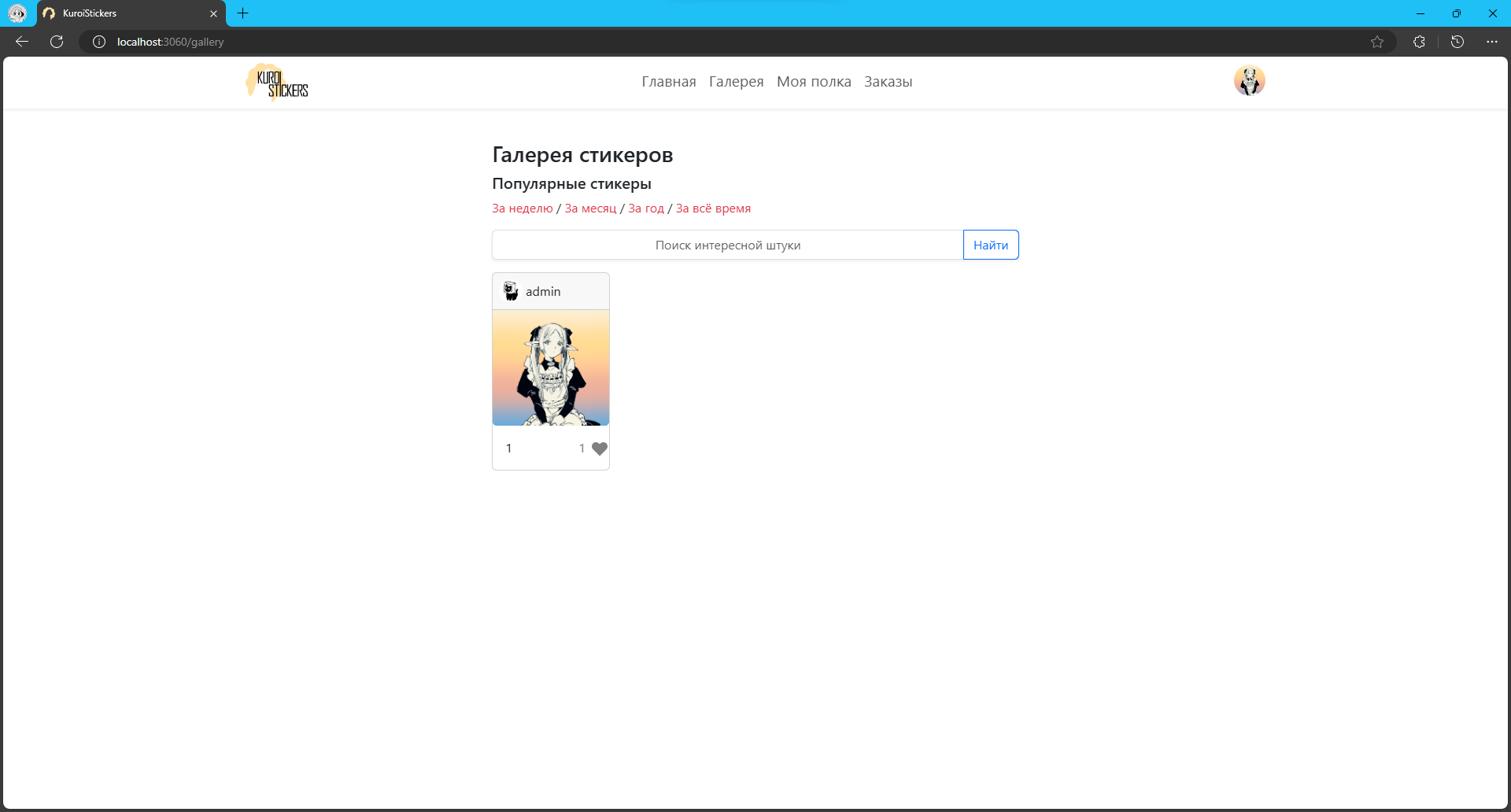


Рисунок 25 – Галерея глазами другого пользователя

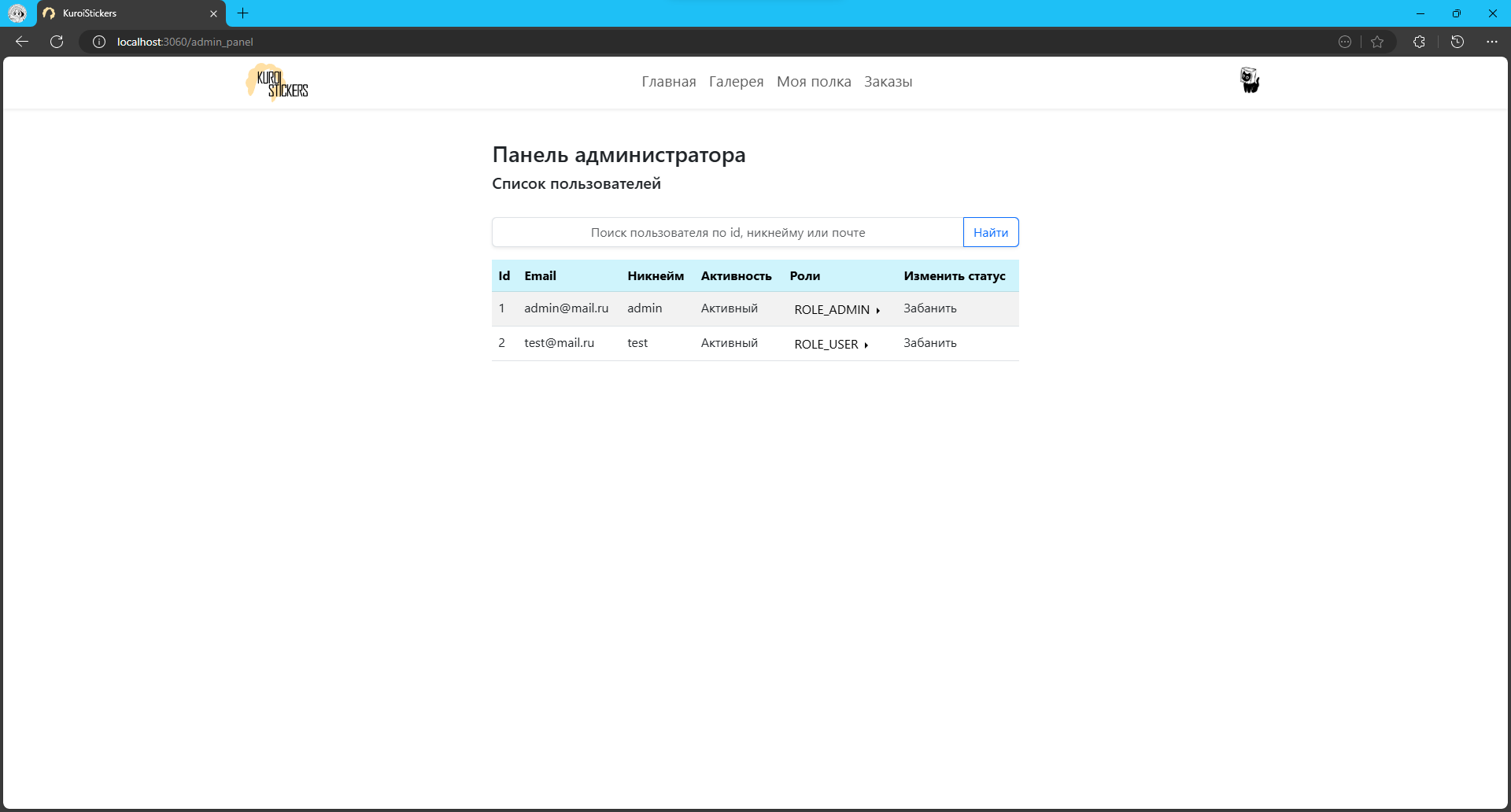


Рисунок 26 – Интерфейс с панелью администратора

* 1. **Тестирование**

Для проверки всех выше указанных элементов потребуется провести тестирование. Тестирование будет производится с помощью браузерного расширения Selenium IDE, которое позволяет создавать автоматизированные тесты.

Результаты тестирования различных модулей от добавления новых элементов до переходов между страницами показаны на рисунках 27-31.

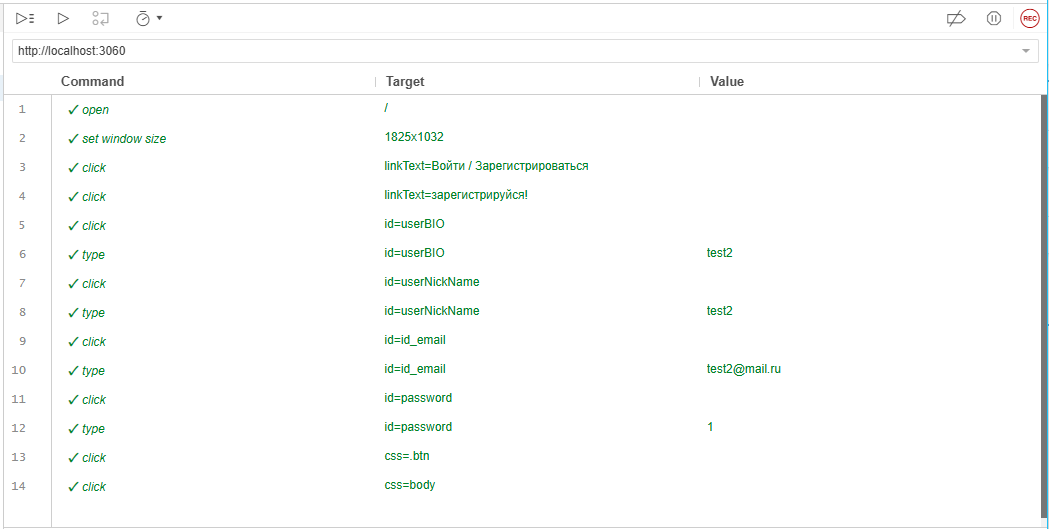


Рисунок 27 – Результат тестирования регистрации пользователя

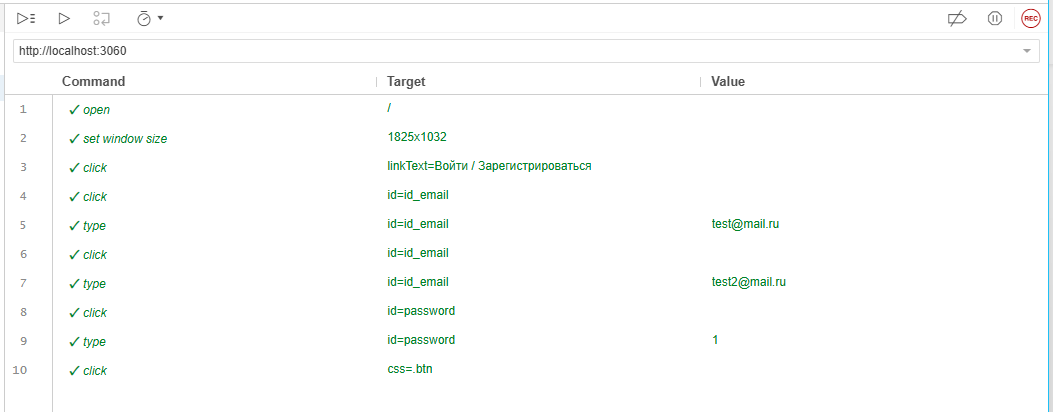


Рисунок 28 – Результат входа пользователя

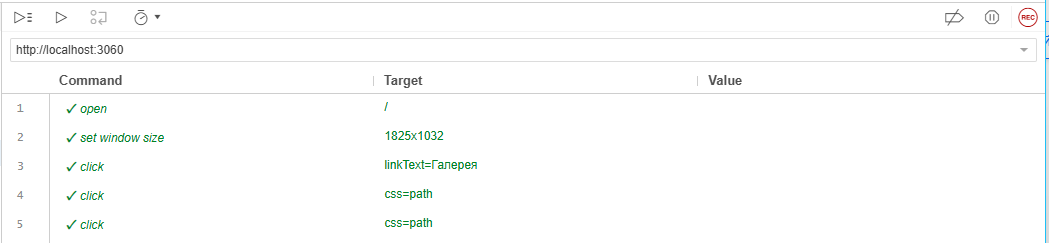


Рисунок 29 – Результат входа пользователя

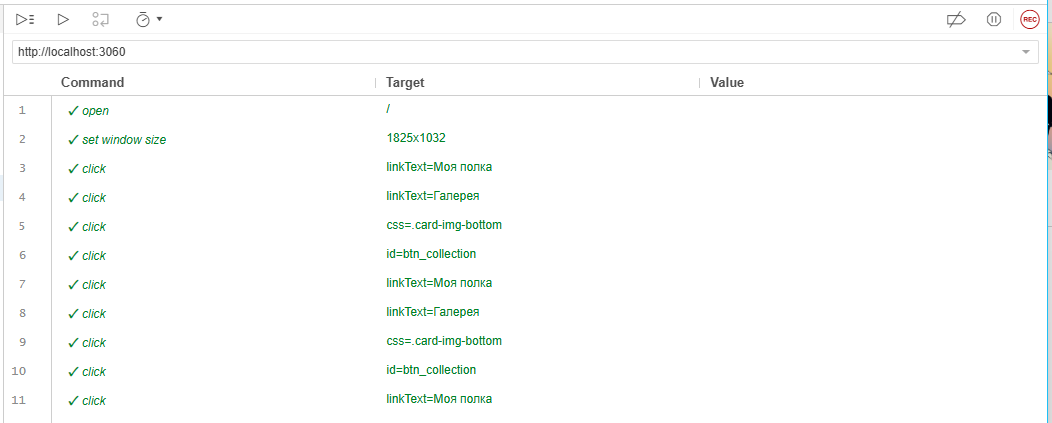


Рисунок 30 – Результат добавления и удаления стикера на полку

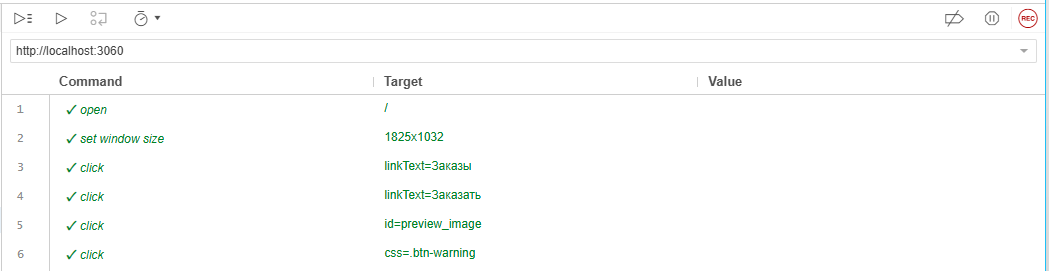


Рисунок 31 – Результат создания заказа

* 1. **Вывод**

В ходе разработки серверной части интернет-ресурса для интернет-магазина были успешно реализованы ключевые функциональные элементы, обеспечивающие корректное функционирование системы.

# **ЗАКЛЮЧЕНИЕ**

В ходе выполнения курсовой работы были достигнуты цели по разработке системы интернет-магазина. Данная система отвечает современным требованиям электронной коммерции. Основной целью разработки было создание веб-сервиса, который сможет значительно упростить работу с заказами и товарами благодаря удобной системе фильтрации, изменения, добавления и удаления данных.

Для достижения поставленной цели был проведен тщательный анализ предметной области, исследованы существующие программные решения, их функциональные возможности, а также их преимущества и недостатки. Это позволило определить ключевые требования к системе и разработать функционал, который выделяет её среди аналогов.

На основе результатов анализа был выбран архитектурный паттерн MVC, который обеспечил разделение логики приложения на модули и упростил процесс разработки и масштабирования. В ходе работы были освоены: MySQL, HTML/CSS, Spring MVC.

Серверная часть системы была реализована с использованием Java, Spring MVC и MySQL. База данных была спроектирована с учётом потребностей подобных интернет-магазинов, обеспечивая эффективное хранение данных о товарах, заказах и пользователях.

Тестирование подтвердило корректную работу всех модулей системы. Реализованный функционал позволяет:

* добавлять новые записи в базу данных;
* получать срезы по популярности стикера;
* изменять уже существующих пользователей;

GIT-репозиторий с файлами на проект курсовой работы: https://github.com/shok1i/Temp

# **СПИСОК ЛИТЕРАТУРЫ**

1. Тузовский, А. Ф. Проектирование и разработка web-приложений : учебное пособие для вузов / А. Ф. Тузовский. — Москва : Издательство Юрайт, 2021. — 218 с. — (Высшее образование). — ISBN 978-5-534-00515-8. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/469982 (дата обращения: 30.08.2021).
2. Хоффман Эндрю Х85 Безопасность веб-приложений. — СПб.: Питер, 2021. — 336 с.: ил. — (Серия «Бестселлеры O’Reilly»)
3. Мартин, Р. Чистая архитектура. Искусство разработки программного обеспечения / Р. Мартин. — СПб. : Питер, 2021. — 352 c.
4. Раджпут Д. Spring. Все паттерны проектирования.- СПб.: Питер, 2019. 7. Меджуи М., Уайлд Э., Митра Р., Амундсен М. Непрерывное развитие API. Правильные решения в изменчивом технологическом ландшафте. СПб.: Питер, 2020.
5. Диков А. В. Клиентские технологии веб-дизайна. HTML5 и CSS3 [Электронный ресурс]:учебное пособие.- Санкт-Петербург: Лань, 2019. 188 с. – Режим доступа: <https://e.lanbook.com/book/122174>
6. Шмидт, Р. Spring: инновации и возможности / Р. Шмидт. — СПб. : Питер, 2021. — 384 с. — ISBN 978-5-4461-1276-5. — Текст : электронный // Платформа Питер [сайт]. (дата обращения: 19.10.2024).
7. Бауманн, К. Spring 5. Полное руководство / К. Бауманн. — СПб. : Питер, 2019. — 512 с. — (Серия «Бестселлеры O'Reilly»). — ISBN 978-5-4461-0723-5. — Текст : электронный // Образовательная платформа Питер [сайт]. (дата обращения: 21.11.2024).
8. Бауманн, К. Spring 5 и Spring Boot / К. Бауманн. — М. : ДМК Пресс, 2020. — 416 с. — ISBN 978-5-6040151-7. — Текст : электронный // Образовательная платформа ДМК [сайт]. — URL: <https://www.dmkpress.ru> (дата обращения: 28.11.2024).