**Федеральное государственное бюджетное образовательное учреждение высшего образования**

**«РОССИЙСКАЯ АКАДЕМИЯ НАРОДНОГО ХОЗЯЙСТВА**

**И ГОСУДАРСТВЕННОЙ СЛУЖБЫ**

**при ПРЕЗИДЕНТЕ РОССИЙСКОЙ ФЕДЕРАЦИИ»**

**Поволжский институт управления имени П.А. Столыпина**

Факультет государственного и муниципального управления

Направление подготовки 09.03.03 Прикладная информатика

Образовательная программа Прикладная информатика в менеджменте

Кафедра прикладной информатики и моделирования

**ОТЧЕТ О ВЫПОЛНЕННОЙ РАБОТЕ**

**по дисциплине: «Программная инженерия» на тему:**

**«Разработка сетевого серверного приложения с функциями обращения к базе данных»**

**Автор работы:**

студент 3 курса 131 группы

очной формы обучения

Каппес Григорий Алексеевич

**Научный руководитель:**

Пластун Инна Львовна

Саратов 2025 г.

**Содержание**

С.

[**Введение** 3](#_Toc215073123)

[**Глава 1. Титульная страница** 4](#_Toc215073124)

[**Глава 2. Витрина товаров и система ролей (Гость)** 6](#_Toc215073125)

[**Глава 3. Корзина покупок (Пользователь)** 7](#_Toc215073126)

[**Глава 4. Платежная система** 8](#_Toc215073127)

[**Глава 5. Аналитика продаж (Менеджер)** 10](#_Toc215073128)

[**Глава 6. Панель управления (Администратор)** 13](#_Toc215073129)

[**Глава 7. Панель продавца** 15](#_Toc215073130)

[**Глава 8. Выборка и сортировка данных из БД** 16](#_Toc215073131)

[**Глава 9. Дизайн и интерфейс** 18](#_Toc215073132)

[**Глава 10. Безопасность** 23](#_Toc215073133)

[**Заключение** 26](#_Toc215073134)

[**Библиографический список** 27](#_Toc215073135)

**Введение**

Данный отчет описывает процесс разработки и функциональные возможности сетевого серверного приложения «Интернет-магазин электроники TechTown».

Проект был реализован с использованием современного стека технологий. Backend-часть приложения была разработана на языке Python с применением высокопроизводительного фреймворка FastAPI. Это позволило создать эффективное API, обеспечивающее быструю обработку запросов, автоматическую генерацию документации и асинхронные операции для работы с базой данных.

Для frontend-части использовались технологии HTML, CSS и JavaScript, которые взаимодействуют с backend через REST API. База данных используется для хранения информации о пользователях, товарах, заказах и т.д.

Основной целью проекта была реализация полнофункционального интернет-магазина с системой ролей, корзиной товаров, интеграцией платежной системы, панелью управления для персонала и механизмами для построения отчетов.

**Глава 1. Титульная страница**

Данный раздел описывает титульную страницу магазина, где представлена информация о магазине.

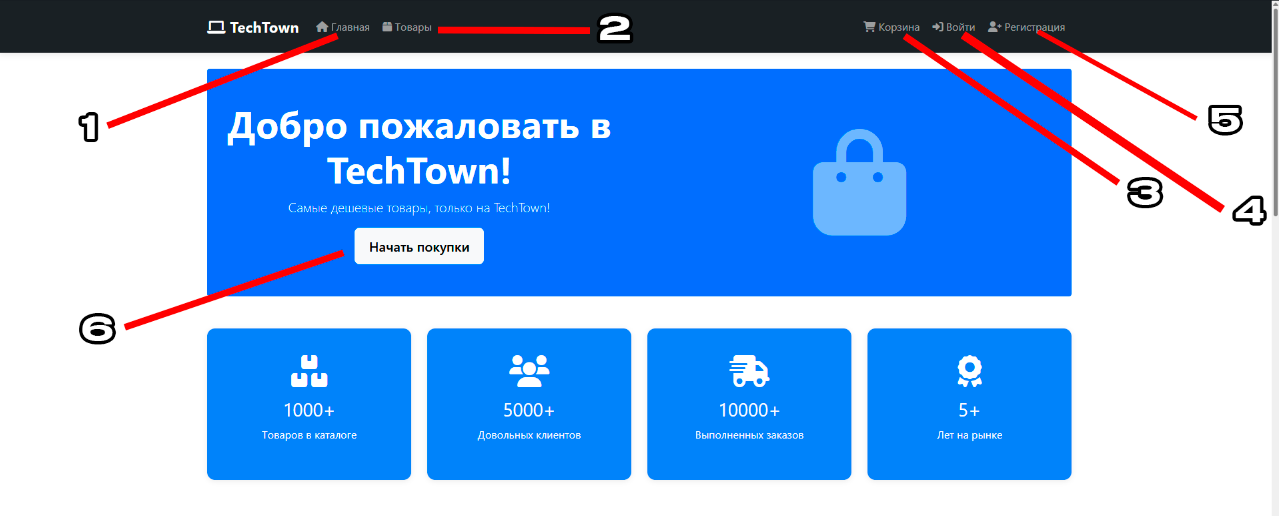


Рис.1. Главная страница

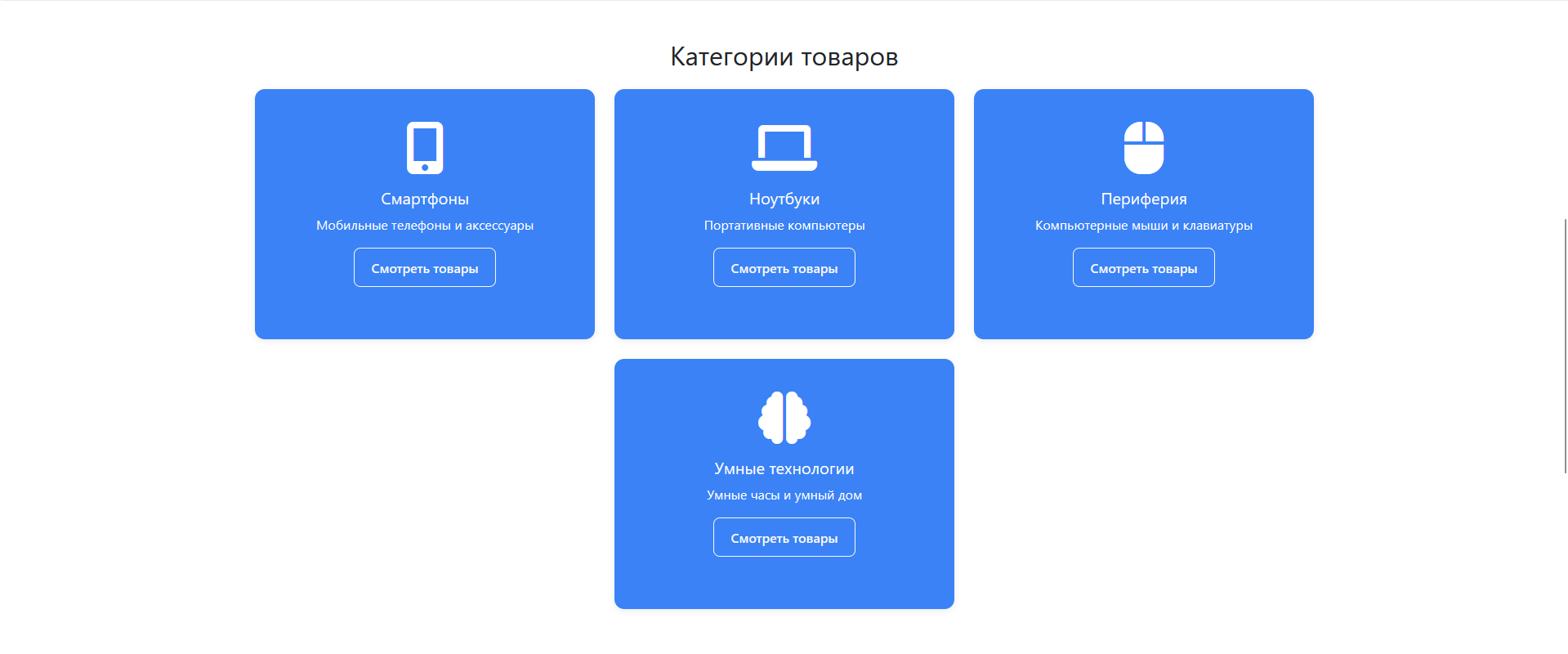


Рис.2. Главная страница (категории товаров)

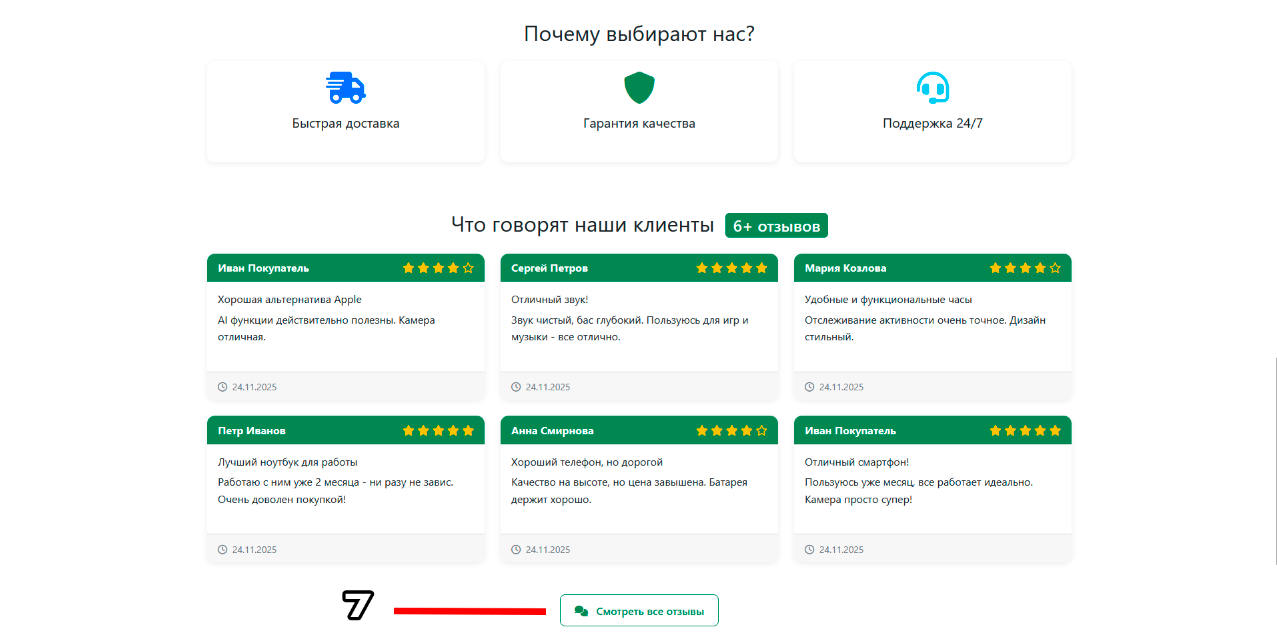


Рис.3. Главная страницы (отзывы)

Главная страница интернет-магазина "TechTown". На ней представлен логотип, кнопка главного меню (1), кнопка перехода на страницу каталога (2), кнопка перехода на страницу корзины (3), кнопка перехода на страницу авторизации (4), кнопка перехода на страницу регистрации (5), кнопка перехода на страницу каталога (6), категории товаров, и блок отзывов (7).

Листинг основных модулей:

* @app.get("/"), def read\_root – основной эндпоинт главной страницы.
* Index.html – визуализация главной страницы.
* Base.html – дизайн страниц и навигация.

**Глава 2. Витрина товаров и система ролей (Гость)**

Данный раздел описывает страницу каталога товаров, доступную для неавторизованного пользователя (роль "Гость").

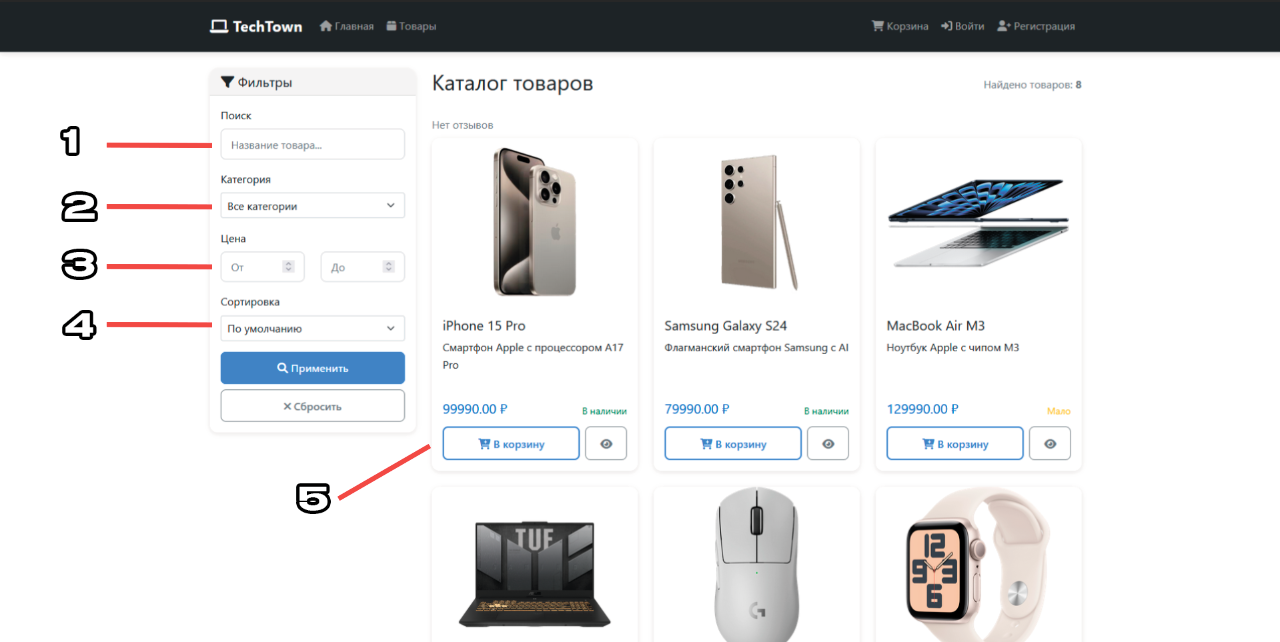


Рис.4. Каталог товаров

Страница отображает все товары в виде сетки. Гость может просматривать товары, использовать фильтры по названию товара (1), категориям (2) и цене (3), а также сортировку (по умолчанию, названию, цене возрастающей и убывающей, по популярности) (4). Функции добавления в корзину (5) и покупки недоступны – при нажатии кнопок отображается призыв к авторизации.

Листинг основных модулей:

* @app.post("/api/products/{product\_id}/update-popularity") – эндпоинт обновление популярности товара (для фильтрации).
* @app.get("/products/",response\_class=HTMLResponse), def products\_page – основной эндпоинт страницы каталога.
* @router.post("/add-to-cart/") в products.py – эндпоинт добавления товара в корзину.
* @router.get("/") в products.py – эндпоинт списка товаров
* Products.html – визуализация страницы каталога.
* Base.html – дизайн страниц и навигация.

**Глава 3. Корзина покупок (Пользователь)**

Данный раздел описывает функционал авторизованного пользователя, включая работу с корзиной.

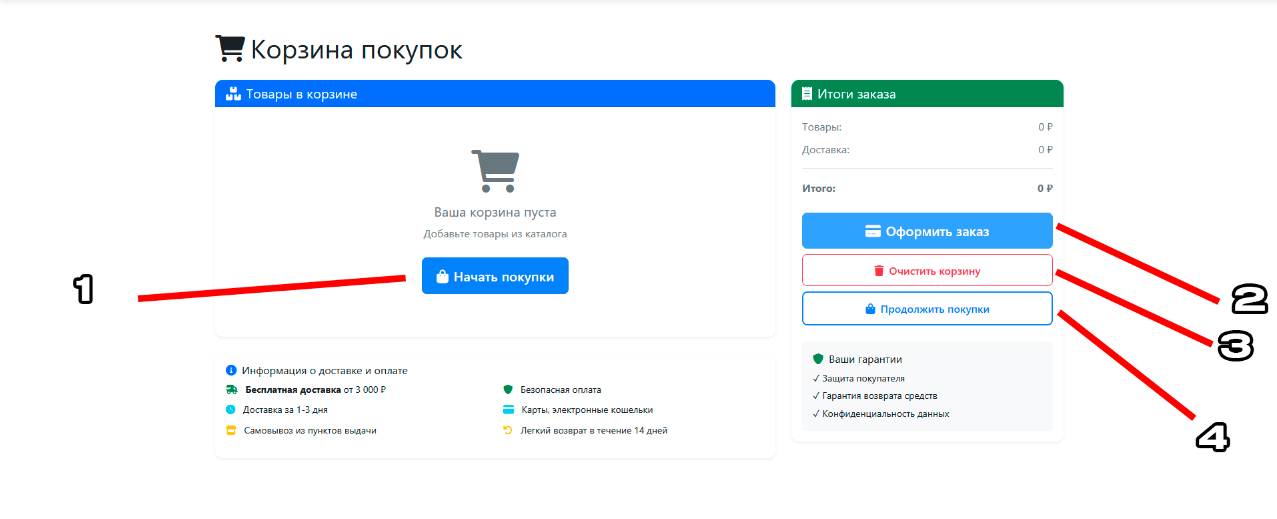


Рис.5. Корзина покупок



Рис.6. Корзина покупок с товарами

После авторизации пользователь получает доступ к корзине. На странице отображаются добавленные товары, их количество, общая сумма. Пользователь может изменить количество (5), удалить товар, перейти к оформлению заказа (2), начать покупки при отсутствии товаров в корзине(1), очистить корзину (3) или продолжить покупки (4).

Листинг основных модулей:

* @app.get("/cart/", response\_class=HTMLResponse) – основной эндпоинт корзины.
* Cart.html – визуализация страницы корзины.
* Base.html – дизайн страниц и навигация.

**Глава 4. Платежная система**

Данный раздел описывает процесс оформления заказа и интеграцию с платежной системой.

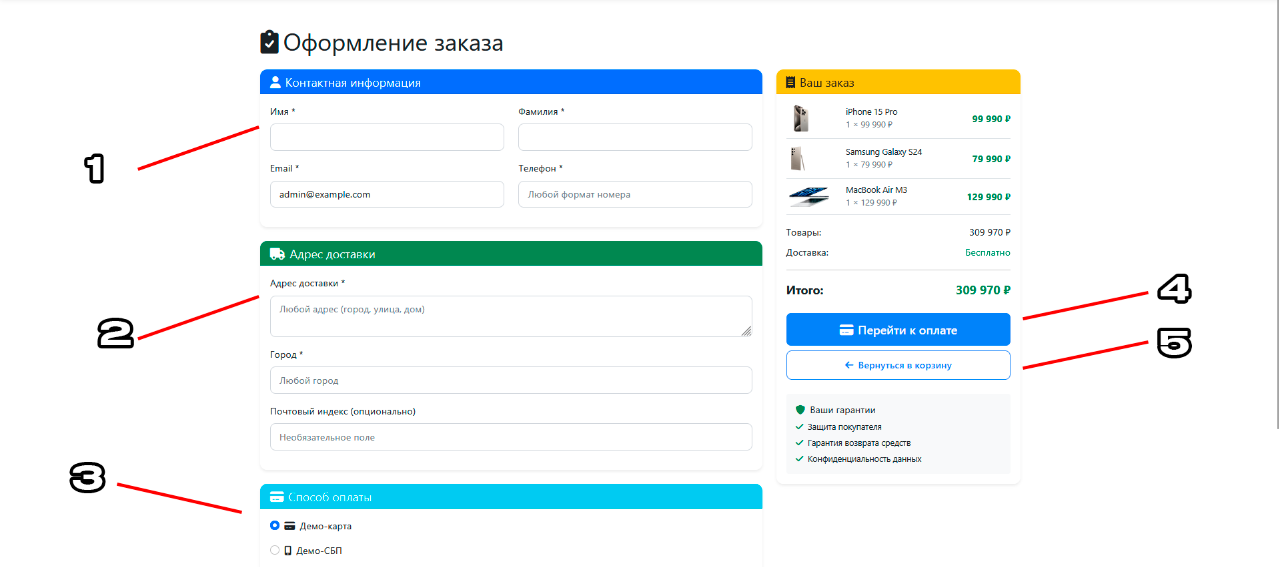


Рис.7. Оформление заказа

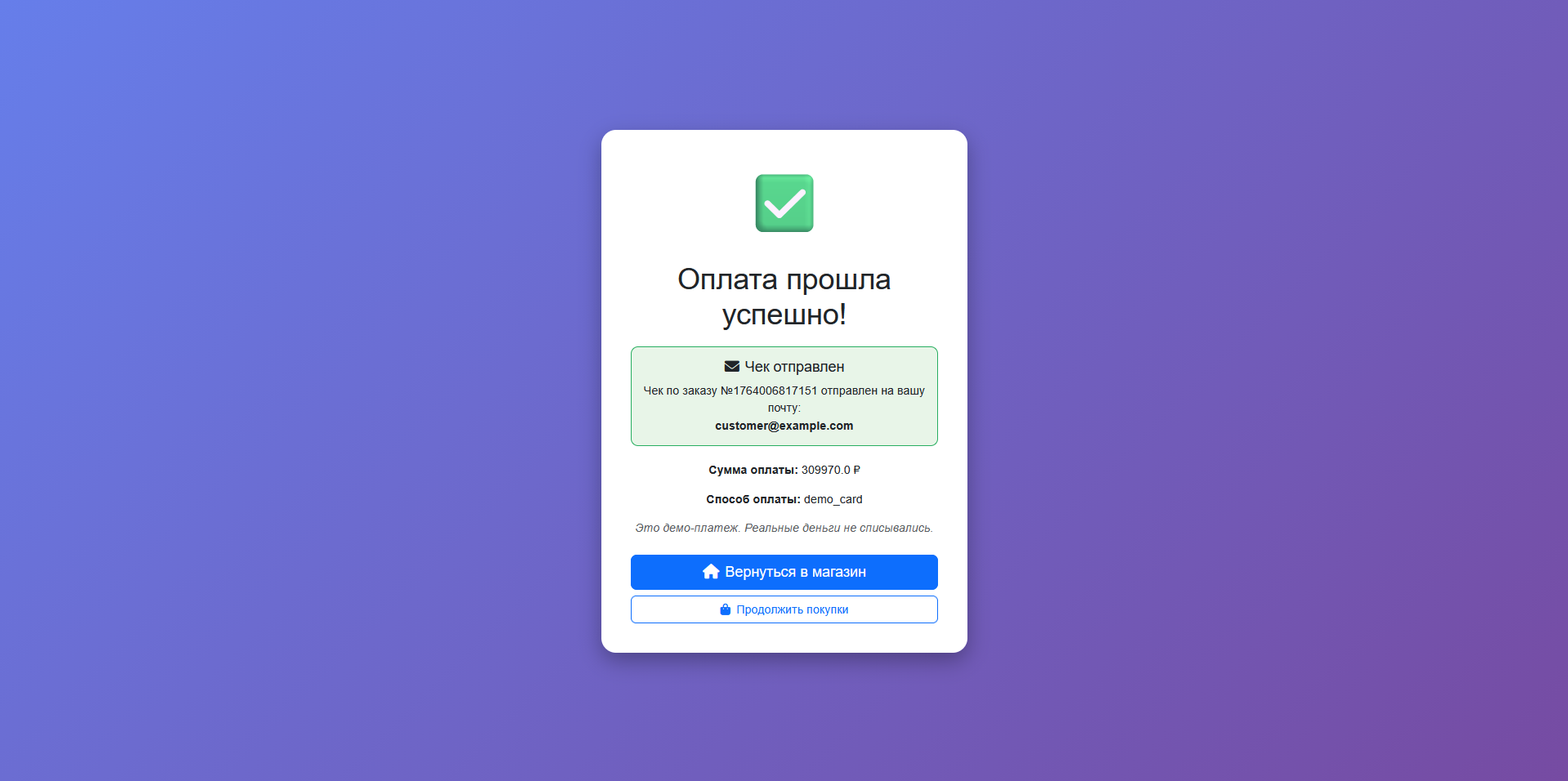


Рис.8. Успешная оплата

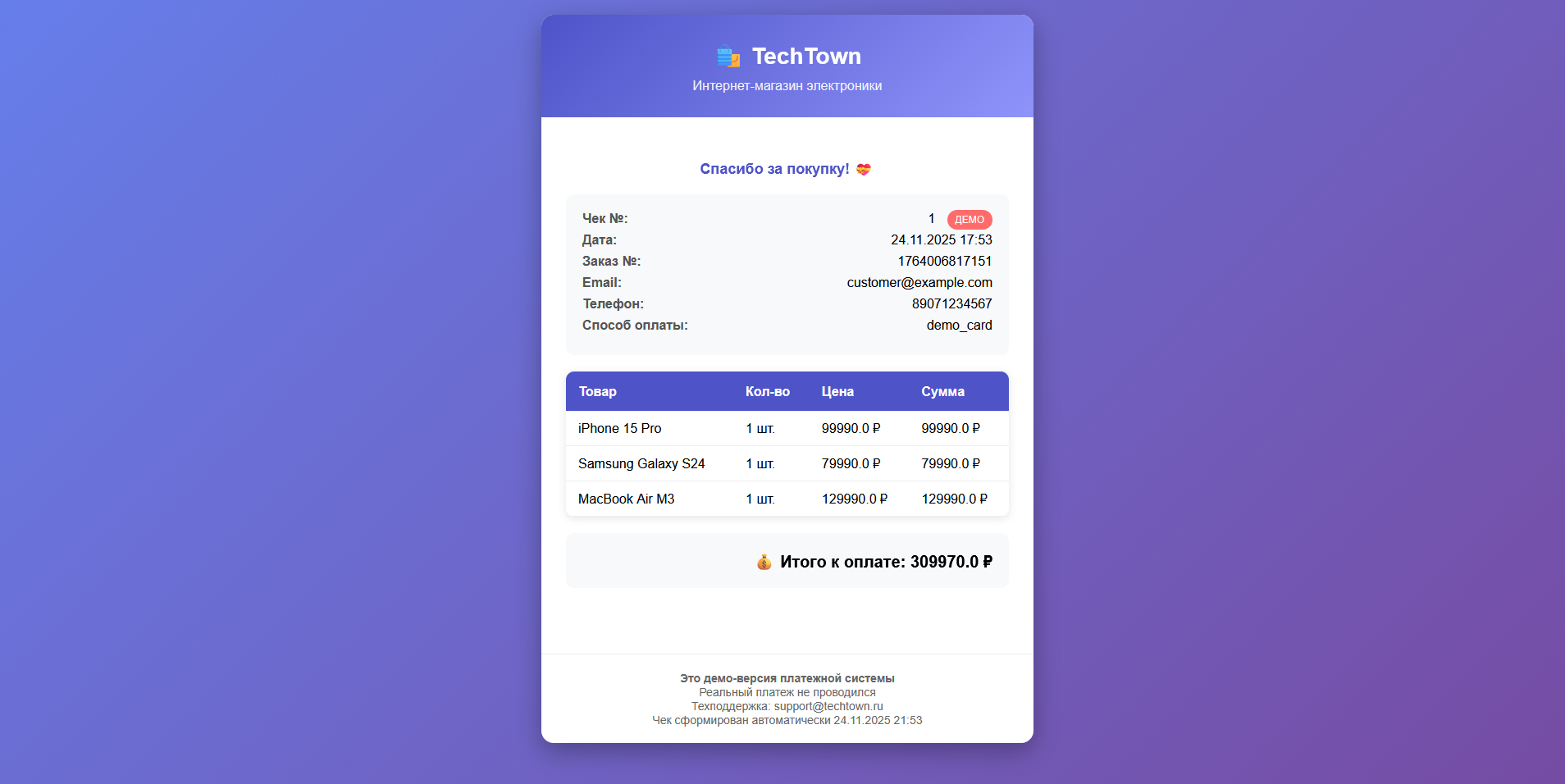


Рис.9. Чек об оплате

Многошаговая страница оформления заказа. Пользователь вводит контактные данные (1), данные для доставки (2), выбирает способ оплаты (3), также может перейти к оплате (4) или вернуться к корзине (5). После пользователь переходит на страницу успешной оплаты и получает чек об оплате.

Листинг основных модулей:

* Payments.py – основной файл платежной системы.
* Checkout.py – файл проверки платежной системы
* @router.get("/checkout/", response\_class=HTMLResponse) – основной эндпоинт проверки платежной системы.
* @router.post("/create", response\_model=PaymentResponse) – эндпоинт создания платежа.
* @router.get("/success/{payment\_id}",response\_class=HTMLResponse) – эндпоинт успешной оплаты.
* @router.get("/{payment\_id}") – основной эндпоинт платежной системы.

**Глава 5. Аналитика продаж (Менеджер)**

Данный раздел описывает интерфейс пользователя с ролью "Менеджер" для аналитики продаж и отчетов.

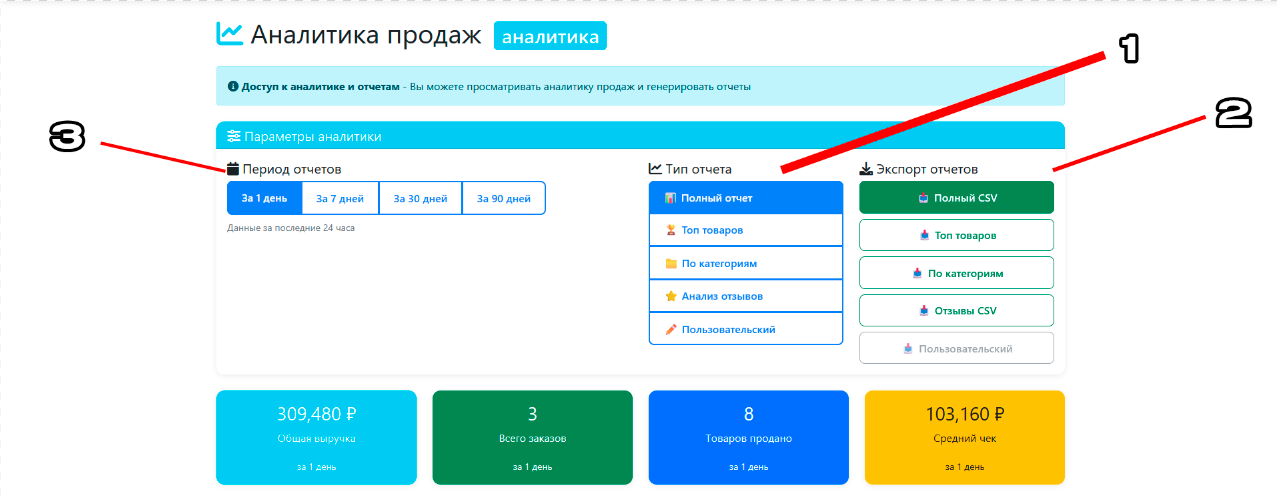


Рис.10. Аналитика продаж

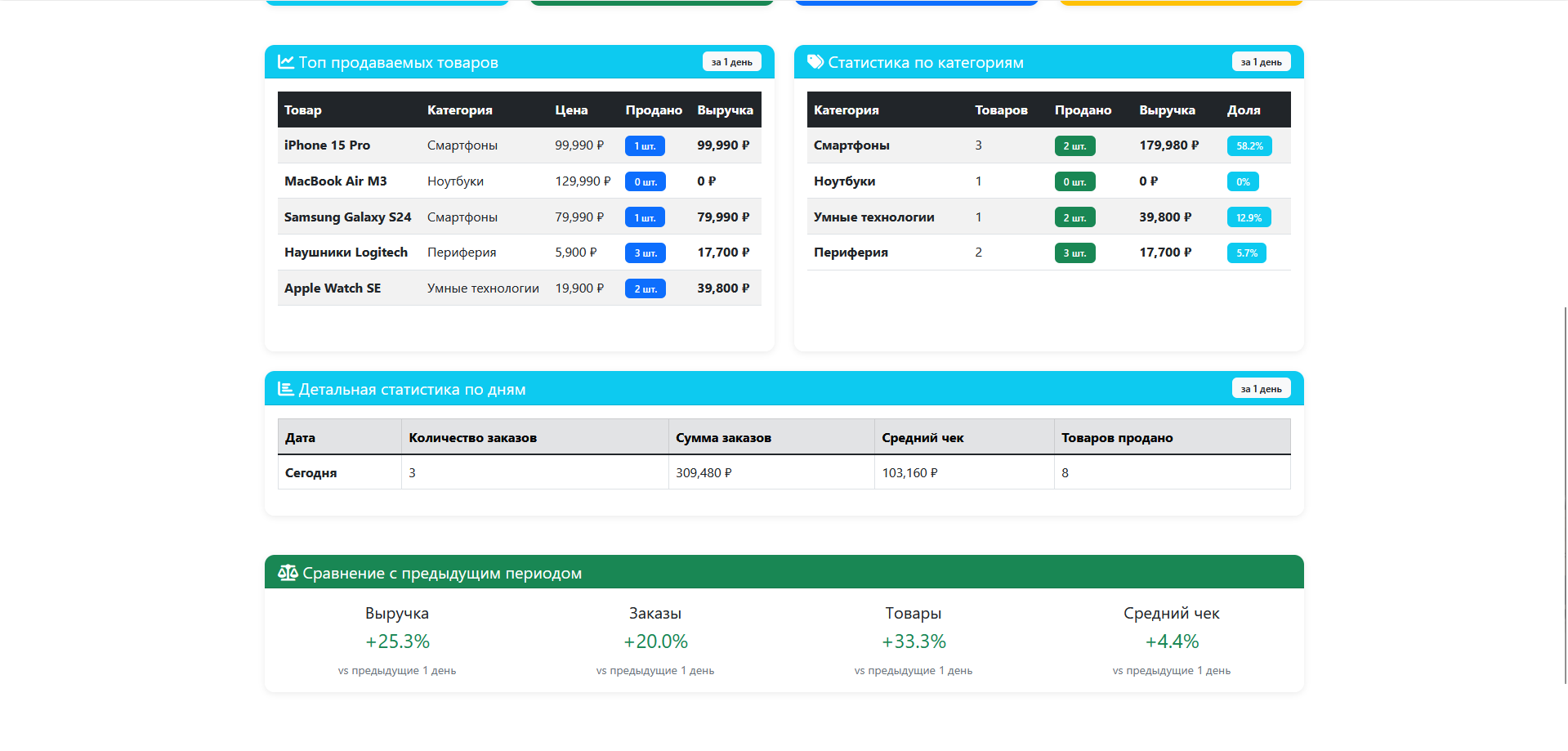


Рис.11. Полная статистика

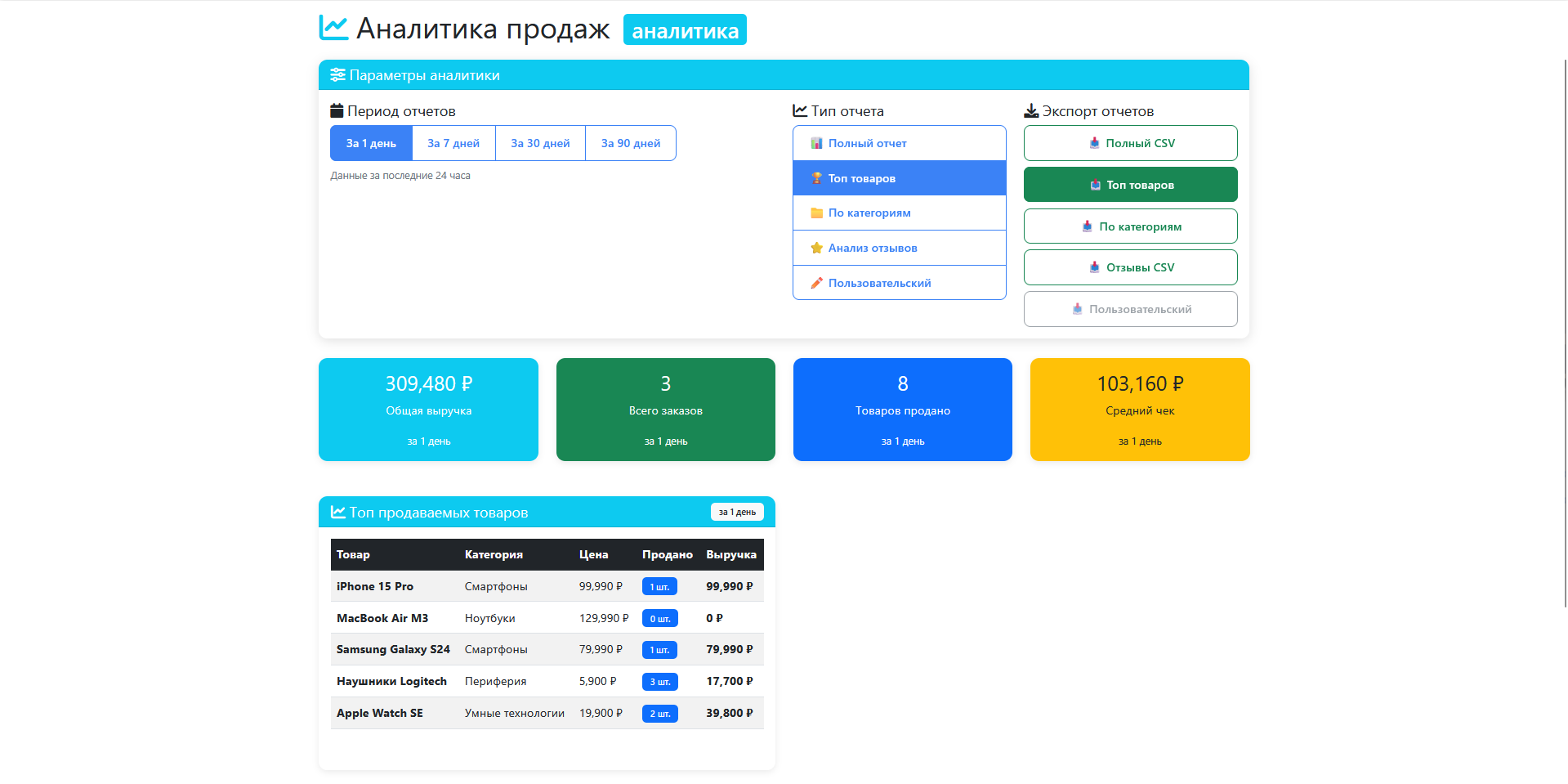


Рис.12. Топ товаров

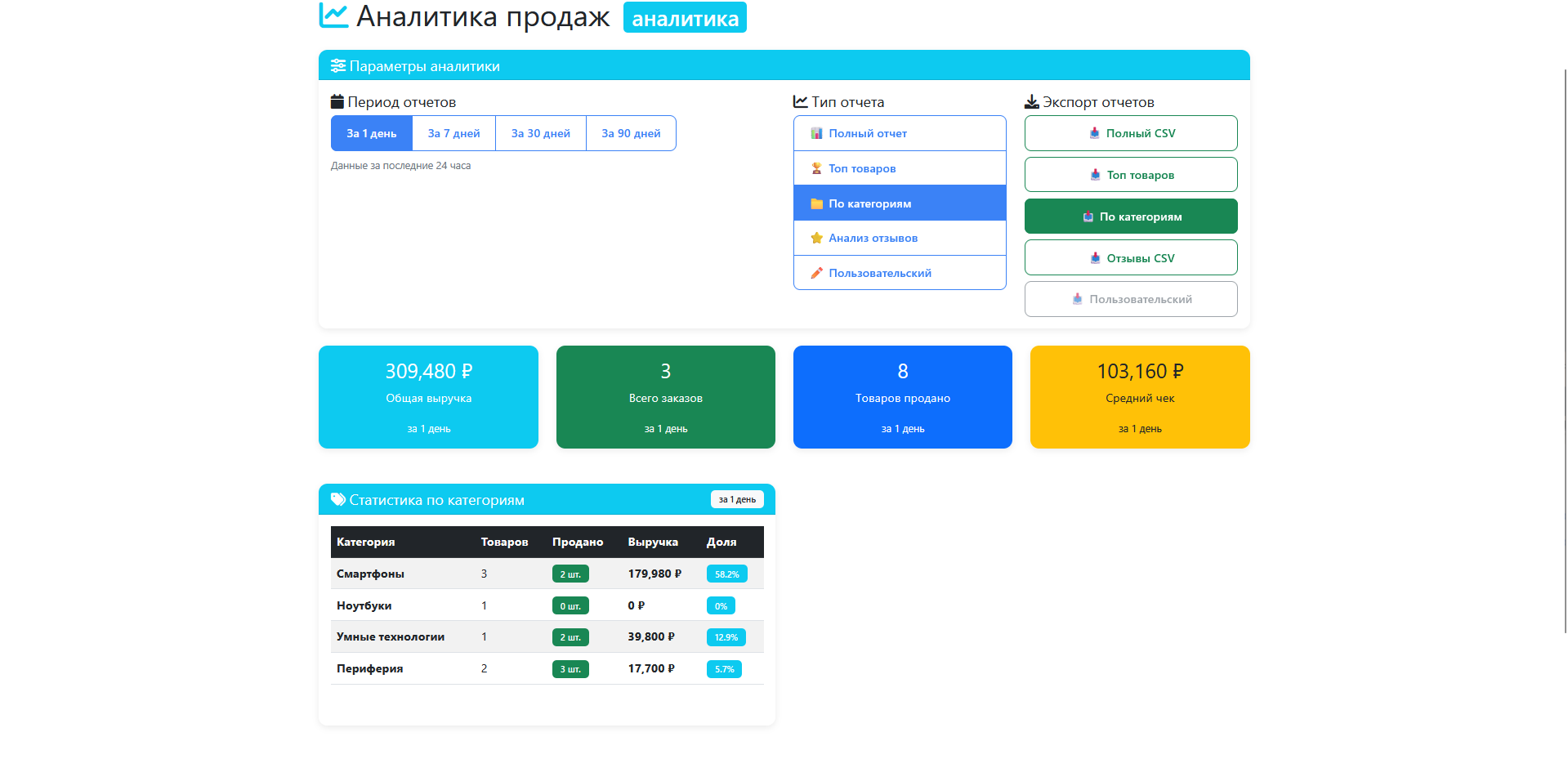


Рис.13. Статистика по категориям

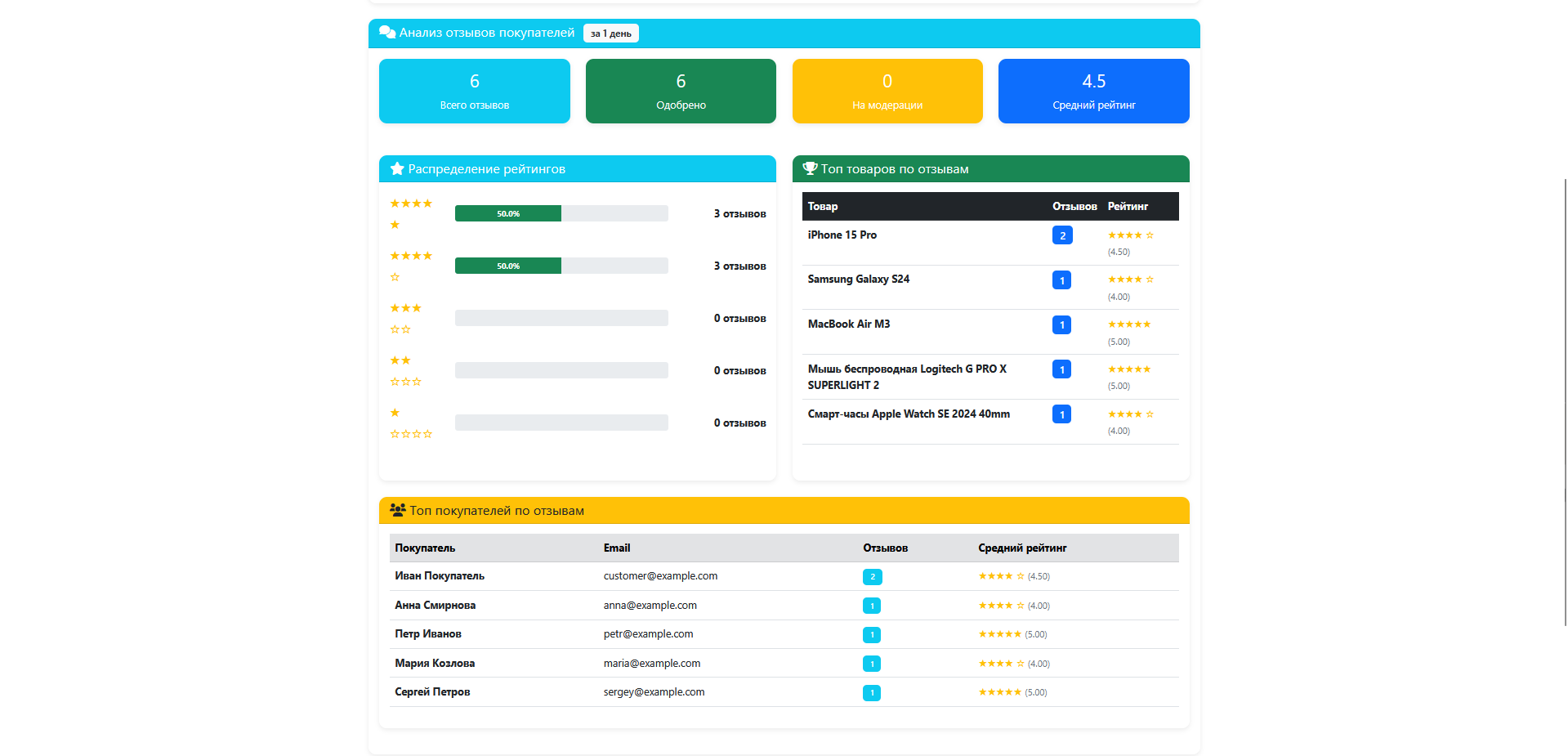


Рис.14. Статистика по отзывам

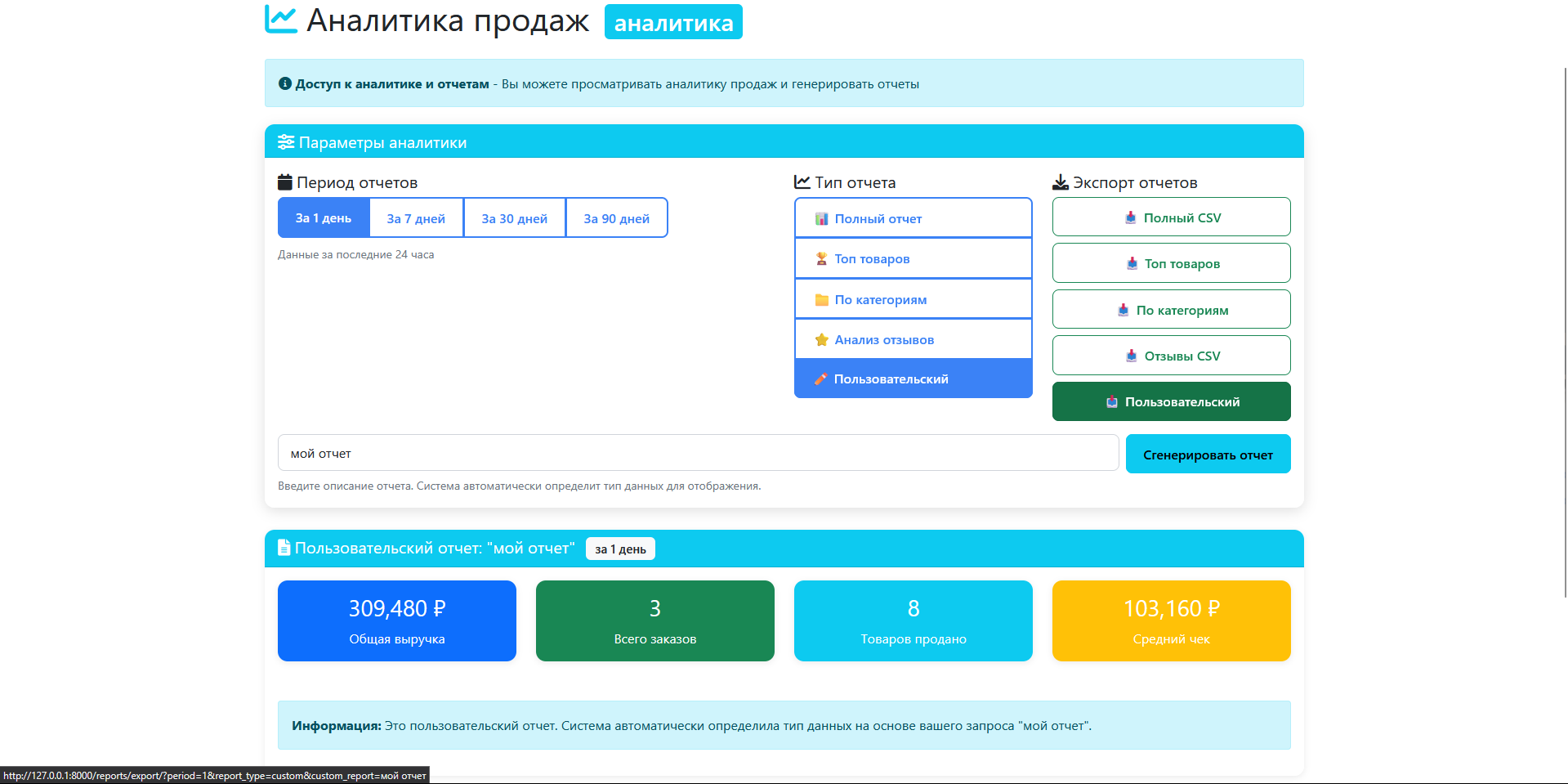


Рис.15. Пользовательский отчет

Аналитика продаж менеджера. Включает в себя возможность выбора периода статистики и отчета (3), типа отчета (1), установки отчета (2). Также можно посмотреть полную статистику, топ товаров, статистики по категориям, статистики по отзывам и составить пользовательский отчет.

Листинг основных модулей:

* reports.py – основной файл аналитики продаж.
* def require\_admin\_or\_manager – функция проверки менеджер или админ пользователь.
* @router.get("/", response\_class=HTMLResponse) – основной эндпоинт аналитики продаж.

**Глава 6. Панель управления (Администратор)**

Данный раздел описывает расширенную панель управления с полным контролем над системой.

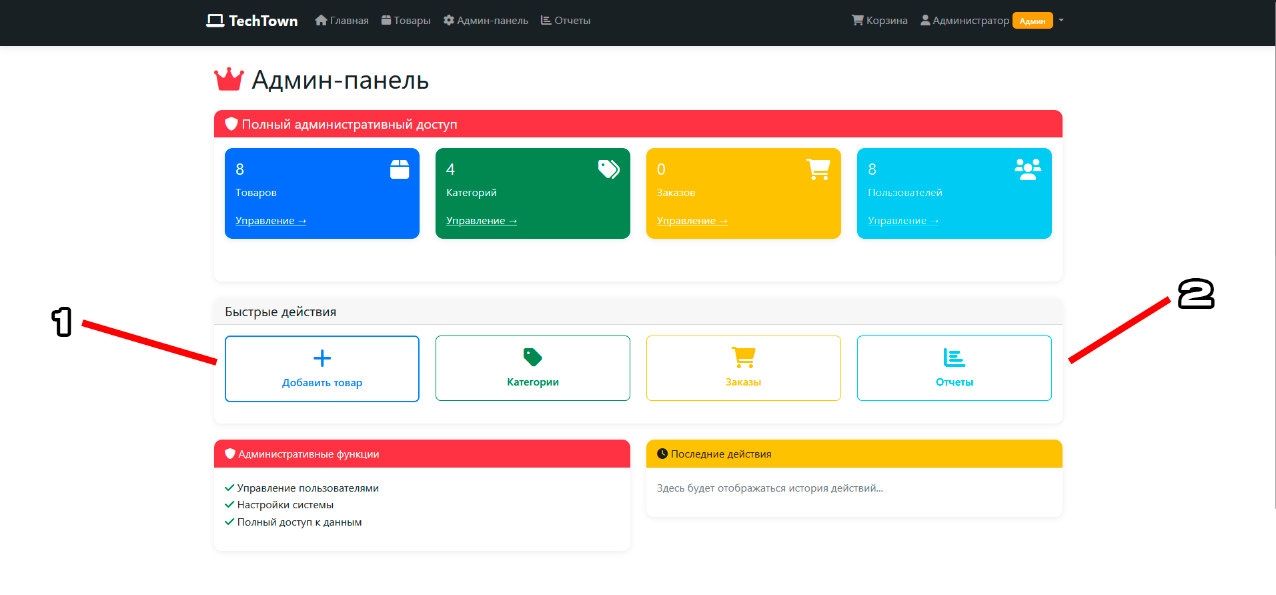


Рис.16. Панель управления

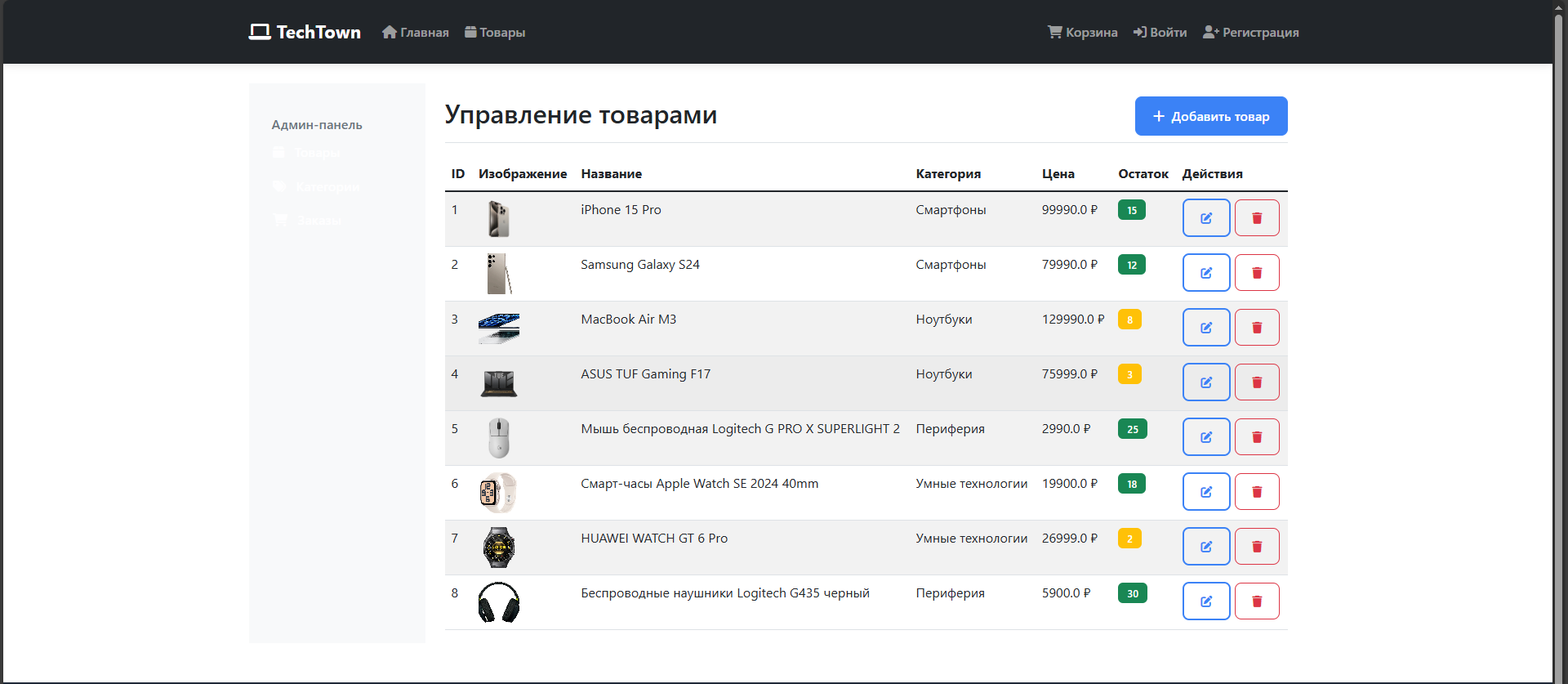


Рис.17. Добавление нового товара в каталог

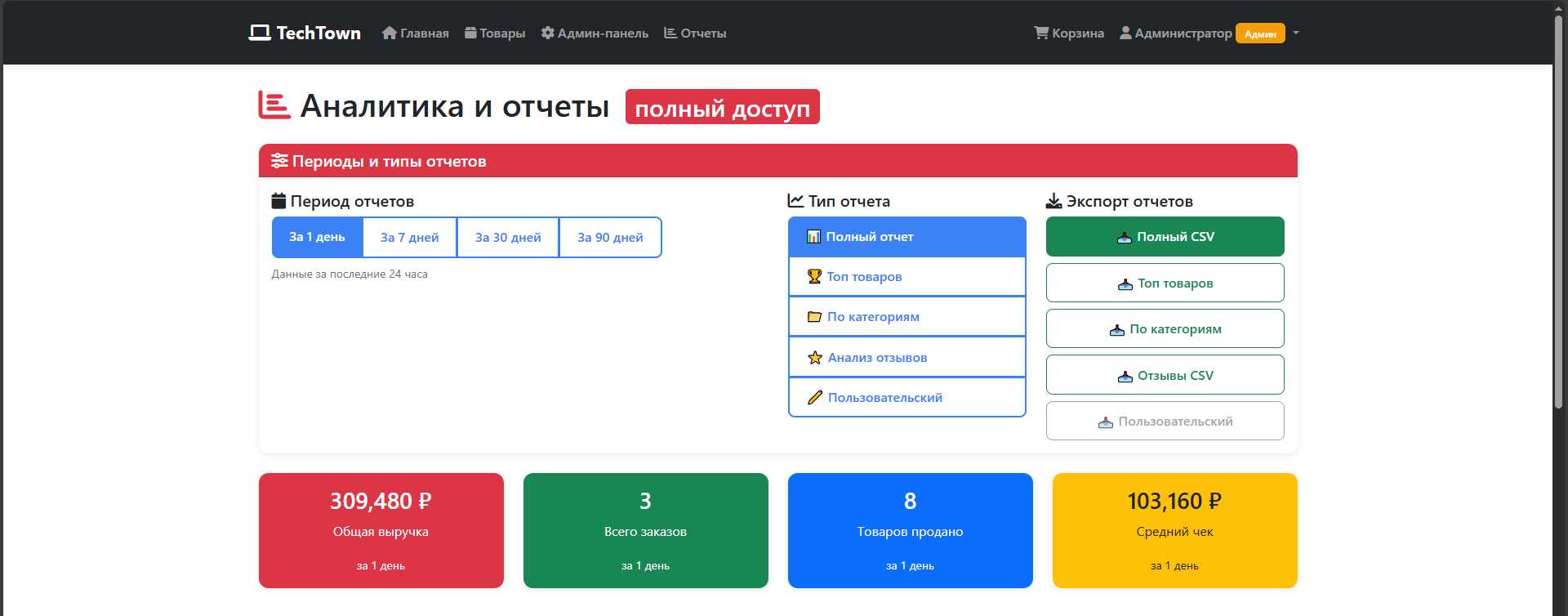


Рис.18. Аналитика и отчеты

Панель управления предоставляет полный доступ. Администратор может добавлять новый товар в каталог (1), а также просмотр отчетов (2)

Листинг основных модулей:

* def check\_admin\_access(current\_user: models.Customer) – проверка доступа для администратора.
* @app.get("/admin", response\_class=HTMLResponse) – основной эндпоинт администратора.
* Admin.py – основной файл для работы администратора.
* admin\_dashboard.html – основной html файл админ-панели.

**Глава 7. Панель продавца**

Данный раздел описывает панель продавца с контролем над товарами.

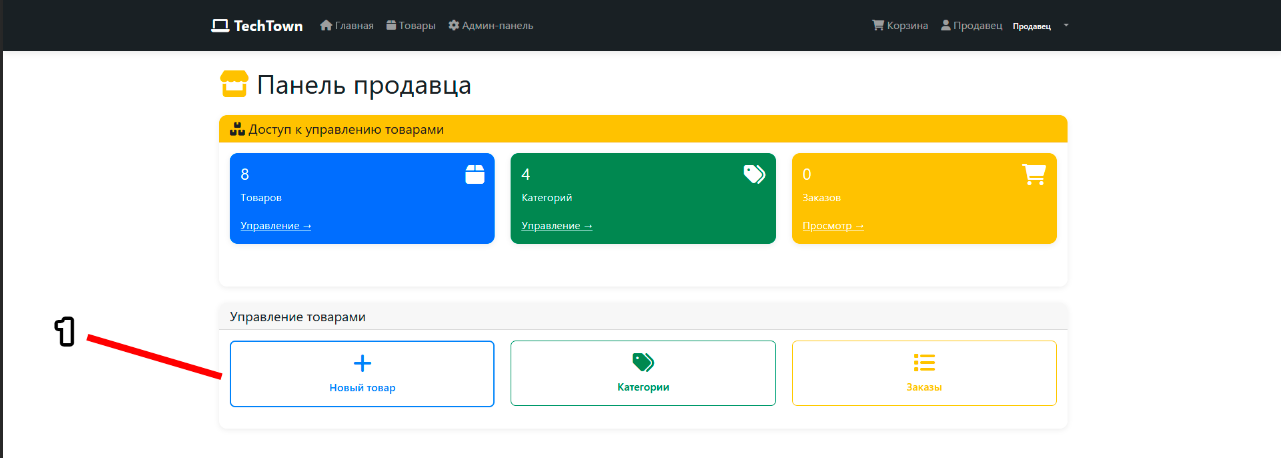


Рис.19. Панель продавца

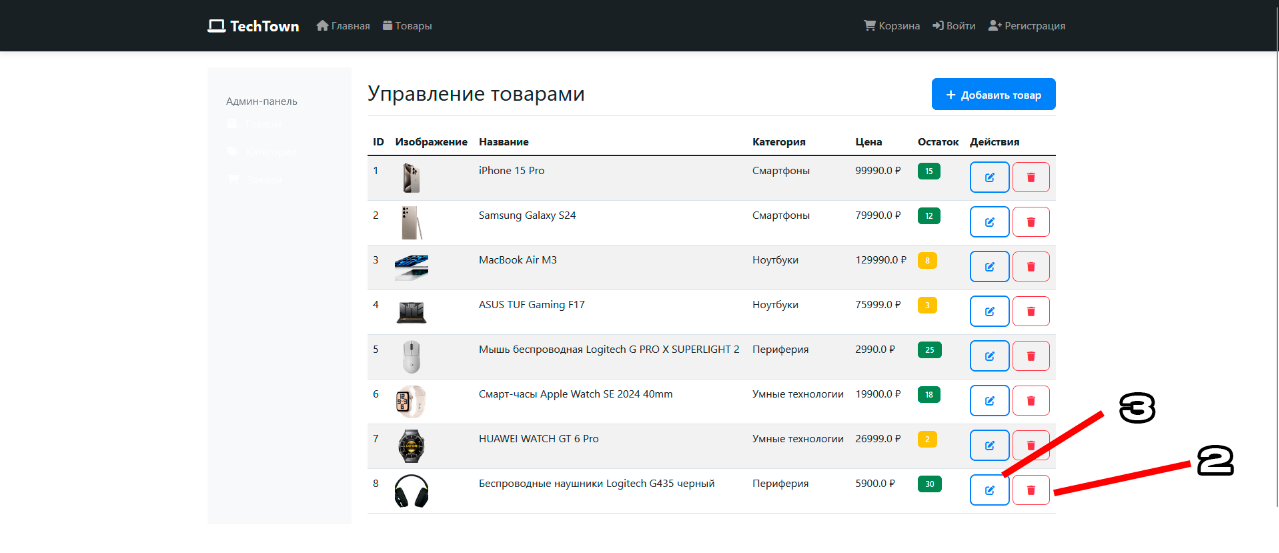


Рис.20. Добавление нового товара в каталог

Панель продавца предоставляет доступ к управлению товарами. Продавец может добавлять новый товар в каталог (1), а также изменять (3) и удалять (2) добавленный.

Листинг основных модулей:

* @app.get("/admin", response\_class=HTMLResponse), current\_user.role == "seller" – основной эндпоинт продавца.
* Auth.py – основной файл для работы авторизации.
* Seller.py – основной файл для работы продавца.
* admin\_products.html – основной html файл добавления товаров.

**Глава 8. Выборка и сортировка данных из БД**

Данный раздел описывает работу с базой данных.

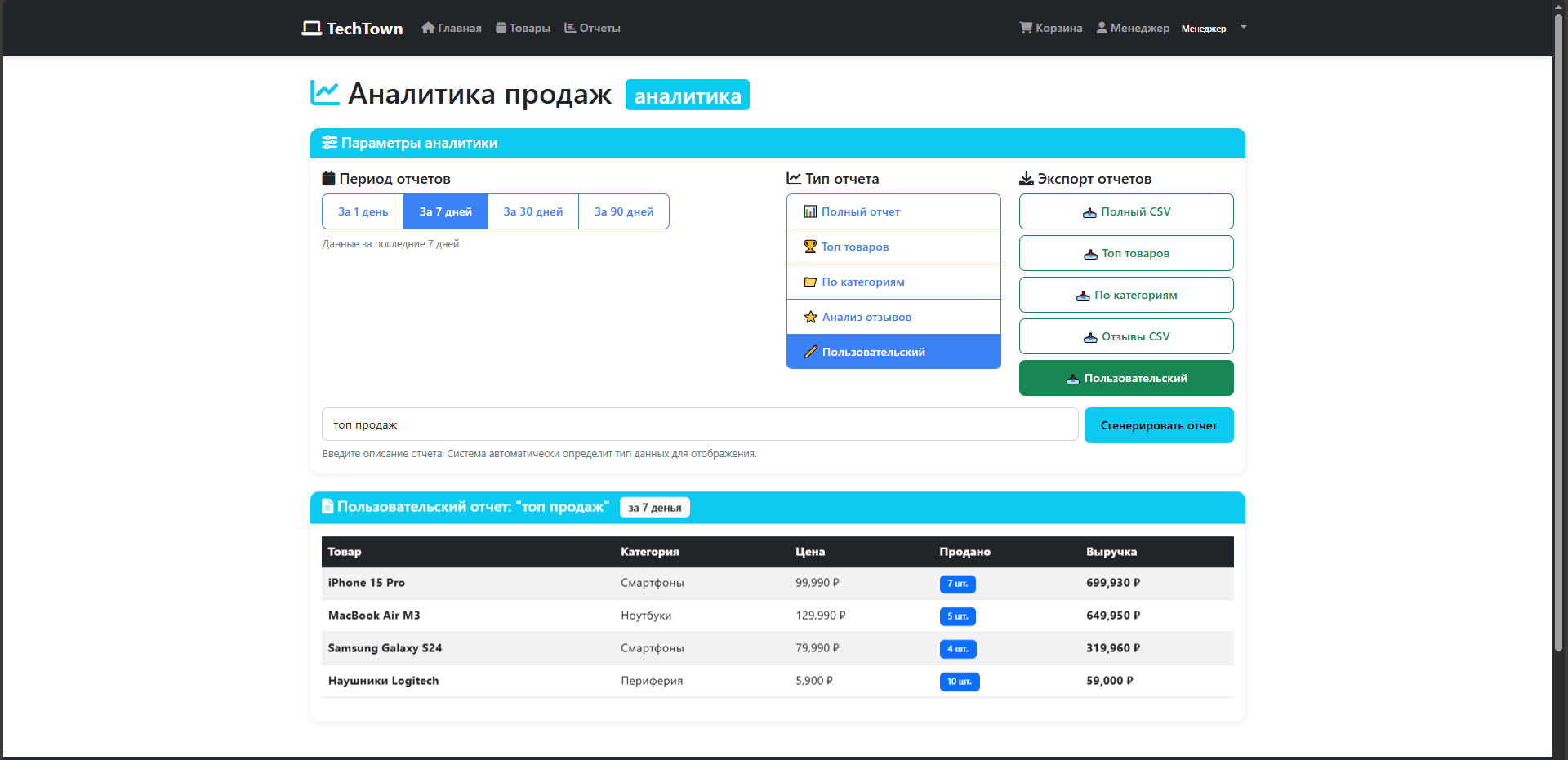


Рис.21. Выборка по топ продажам

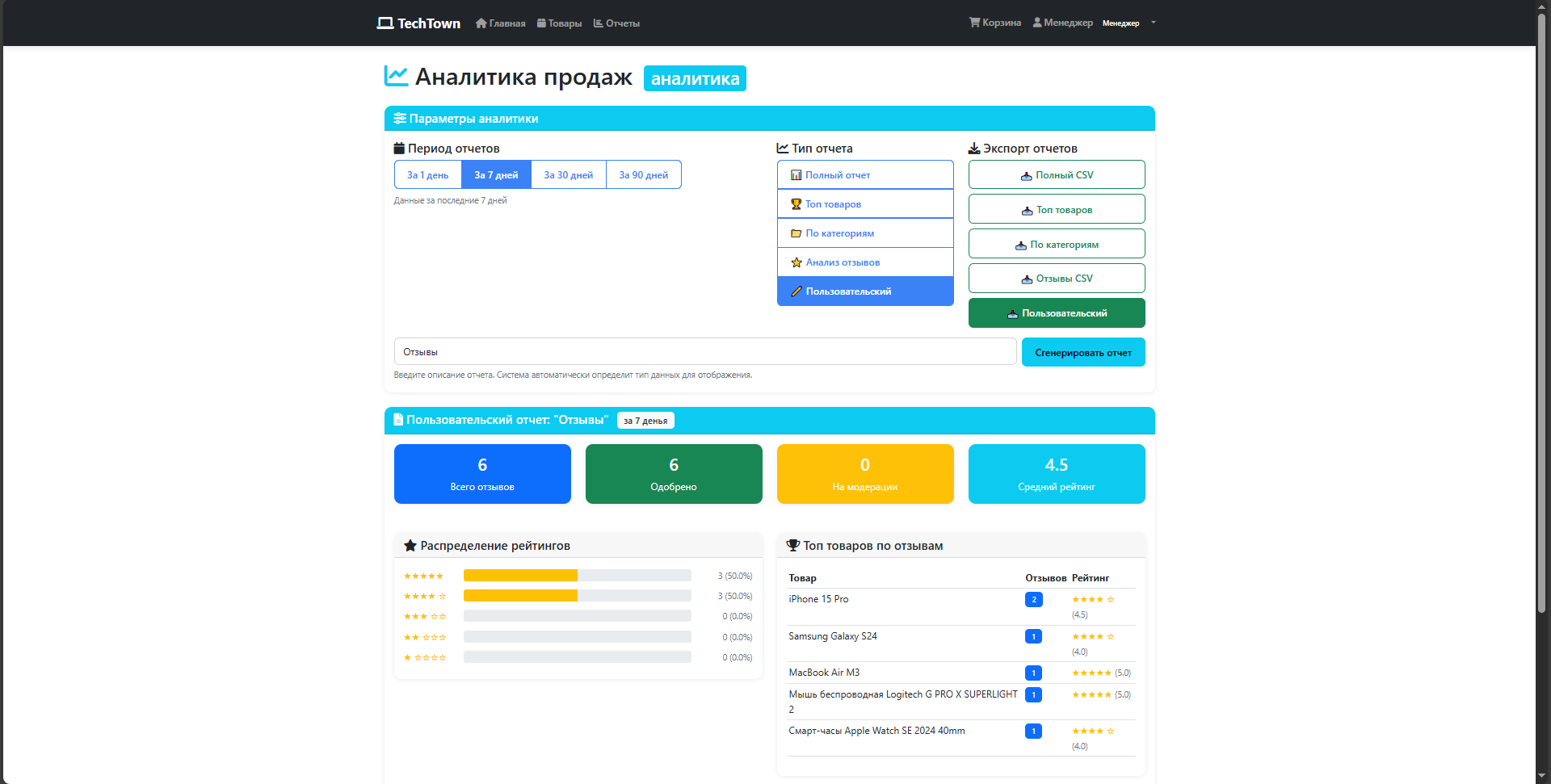


Рис.22. Выборка по отзывам

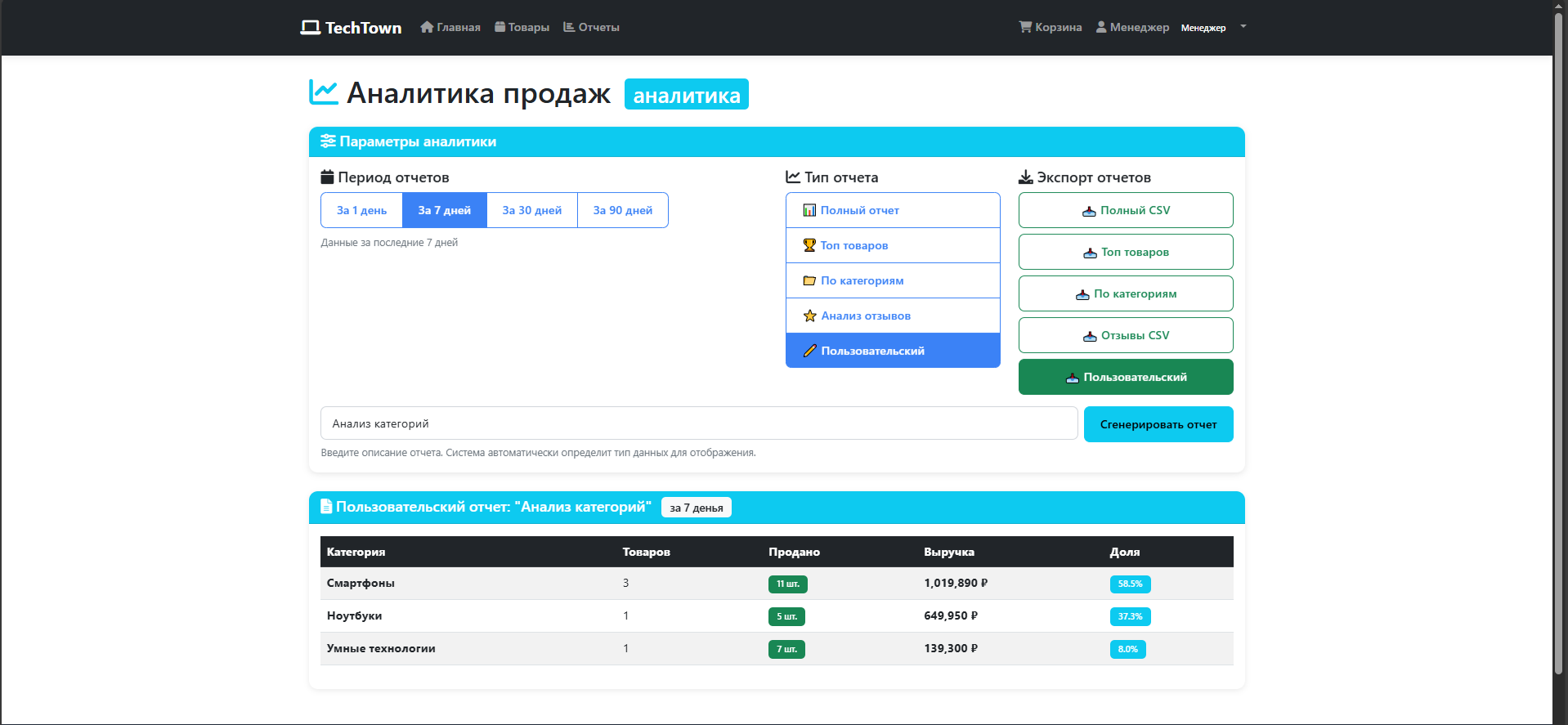


Рис.23. Выборка по категориям

Пример работы выборки и сортировки. Менеджер или администратор может отсортировать статистику по желаемым параметрам: по топ продажам, по отзывам, по категориям.

Листинг основных модулей:

* def get\_reviews\_statistics(db: Session, period: int) – основная функция для работы выборки.
* Reports.py – основной файл для работы выборки.
* reports.html – основной html файл отчетов.

**Глава 9. Дизайн и интерфейс**

Данный раздел описывает единый визуальный дизайн, применяемый на всех страницах приложения для разных ролей.

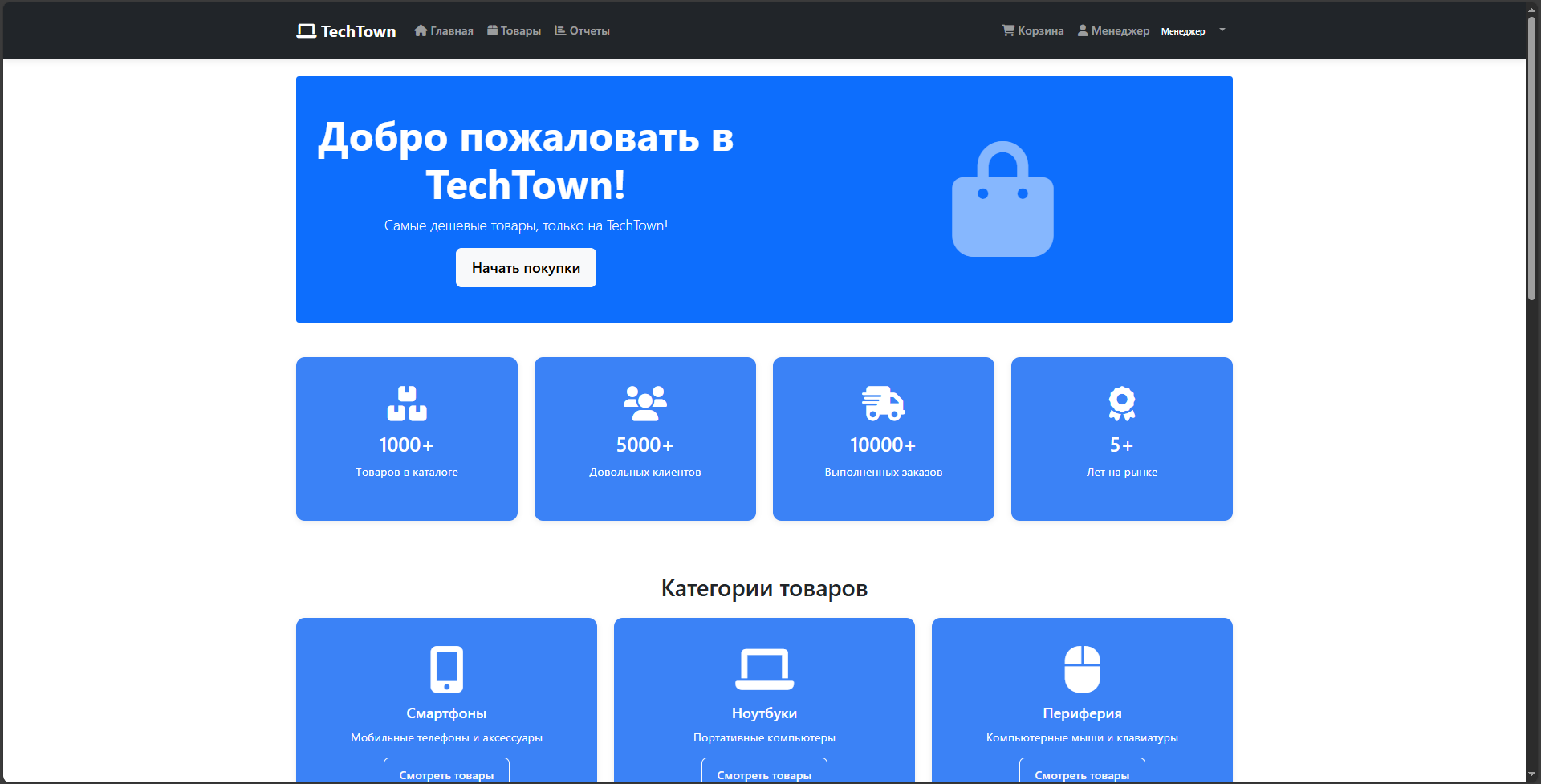


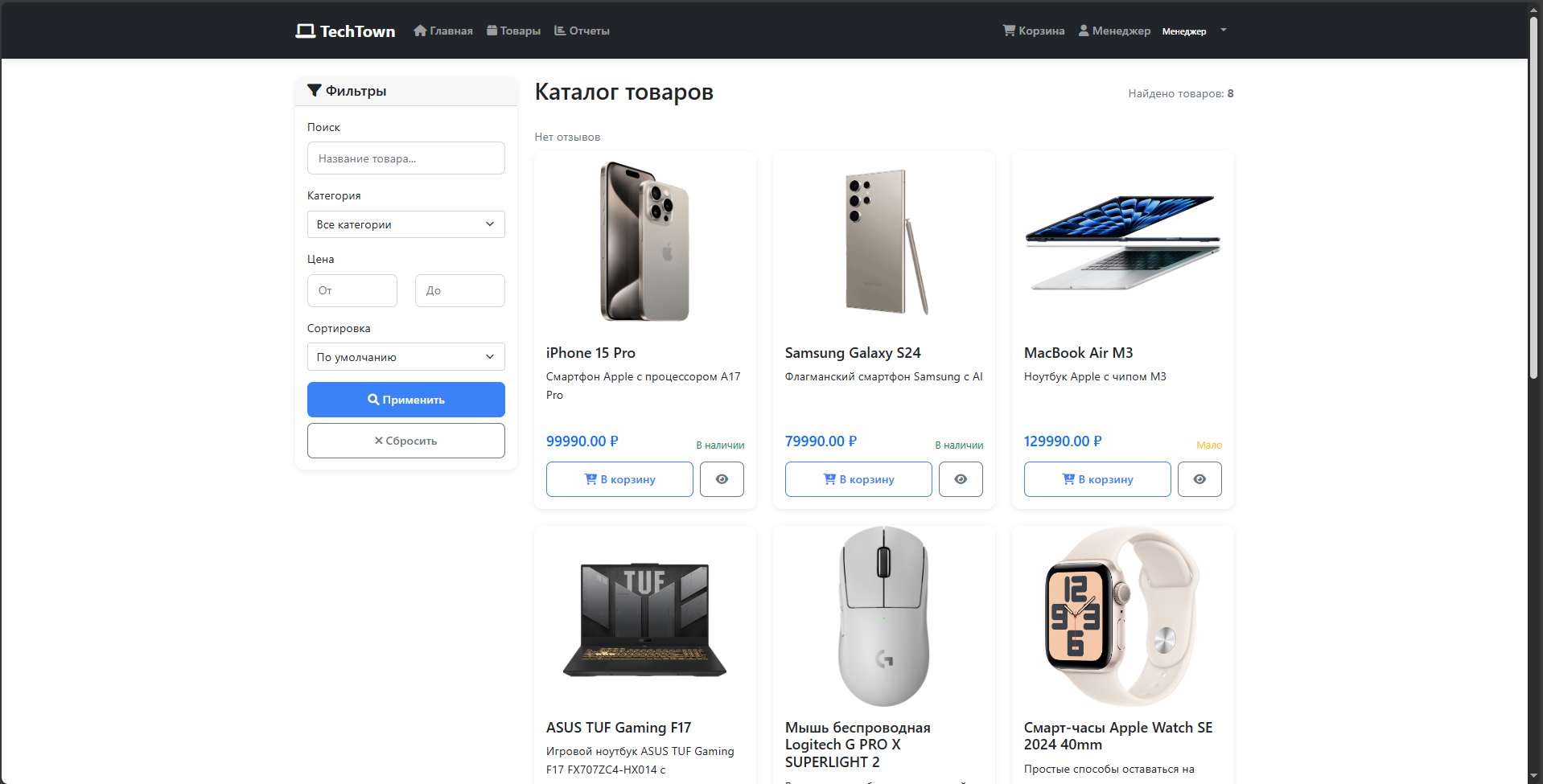
Рис.24. Главная страница

Рис.25. Каталог товаров

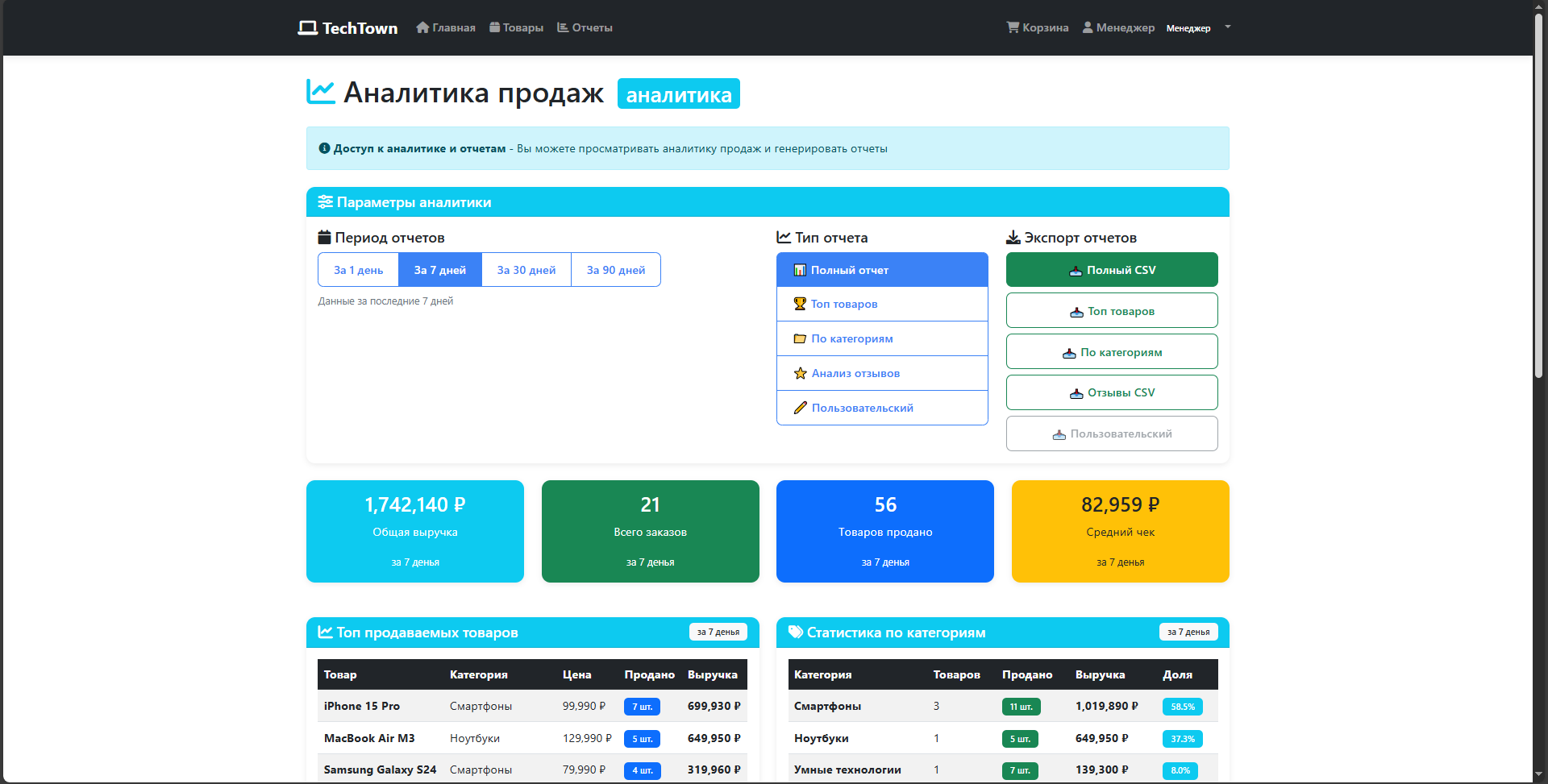


Рис.26. Аналитика продаж (менеджер)

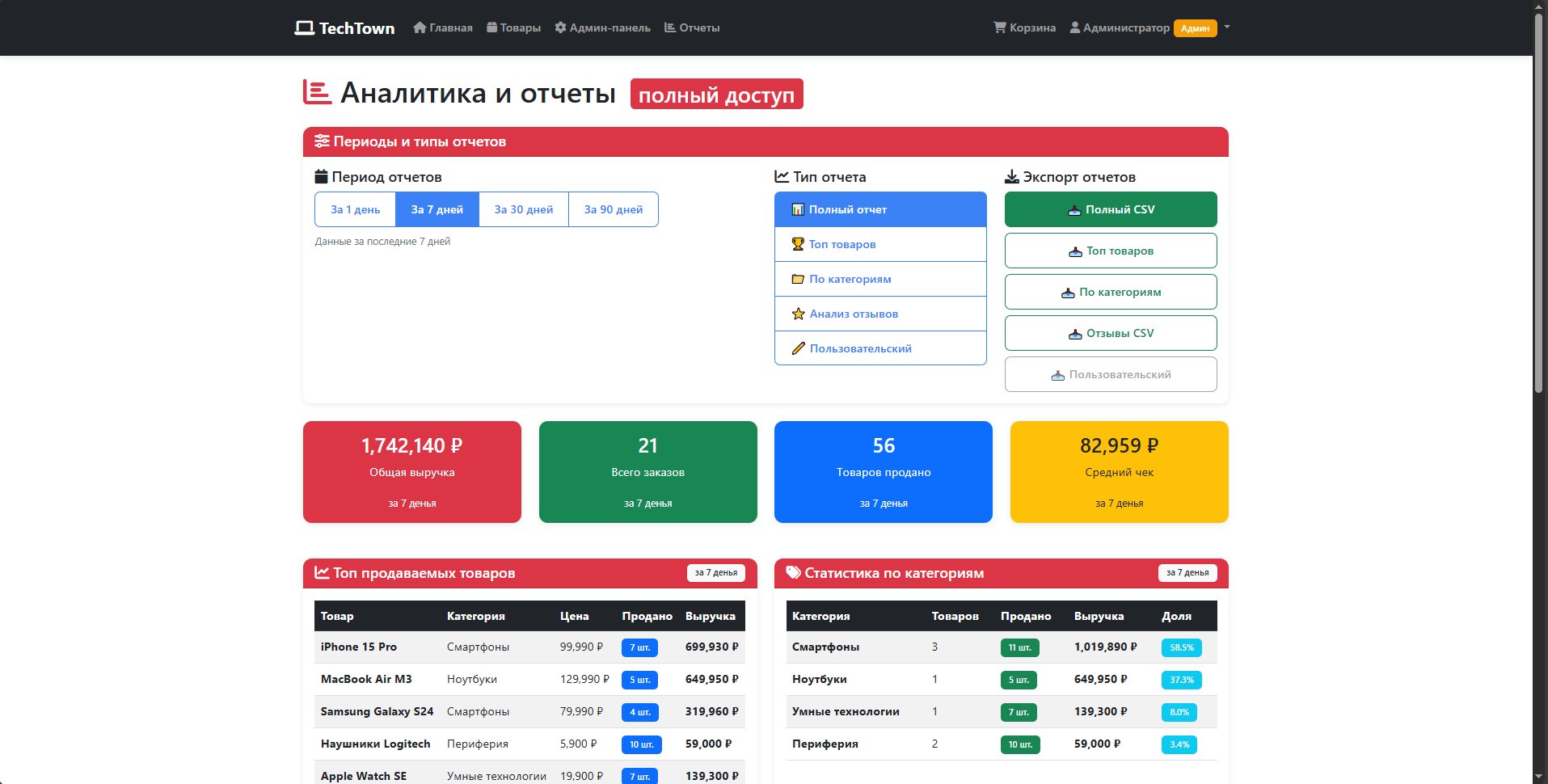


Рис.27. Аналитика и отчеты (администратор)

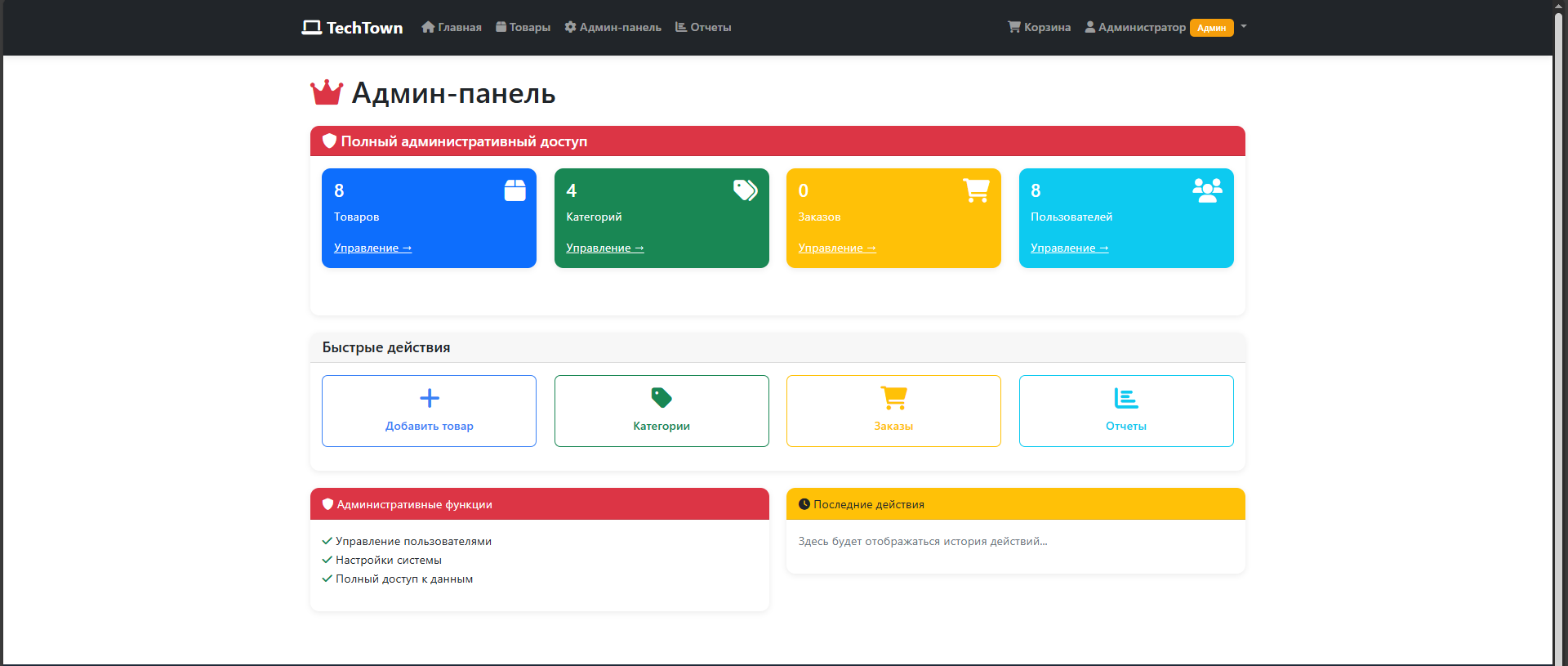


Рис.28. Админ-панель

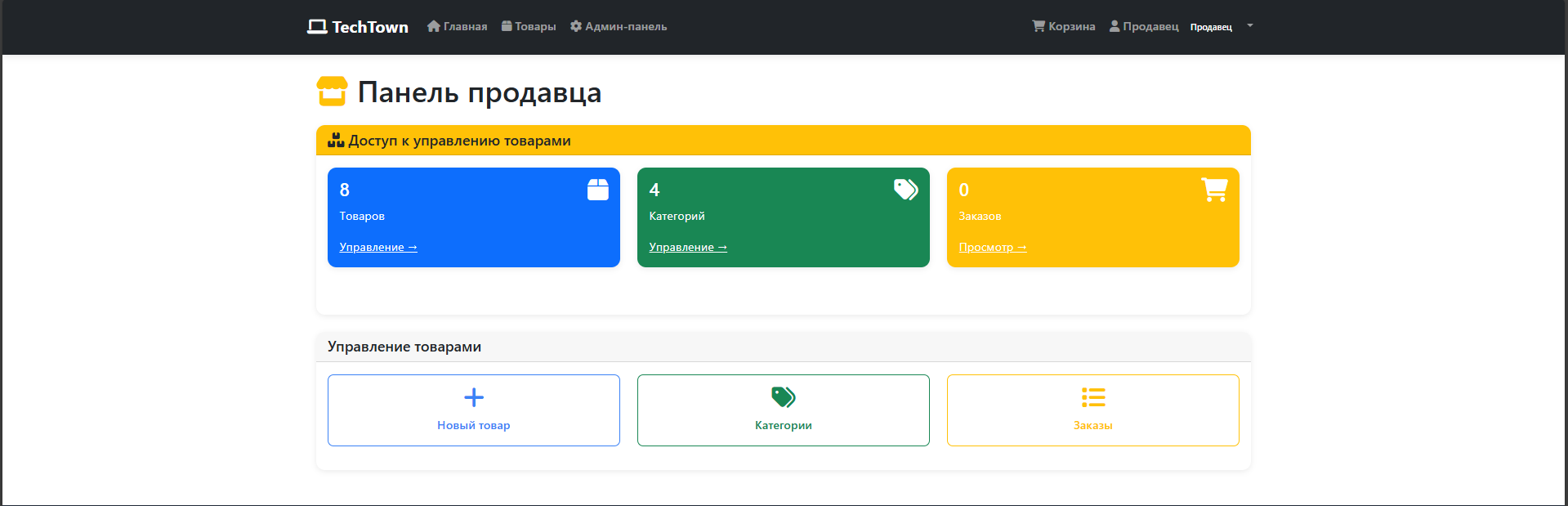


Рис.29. Панель продавца

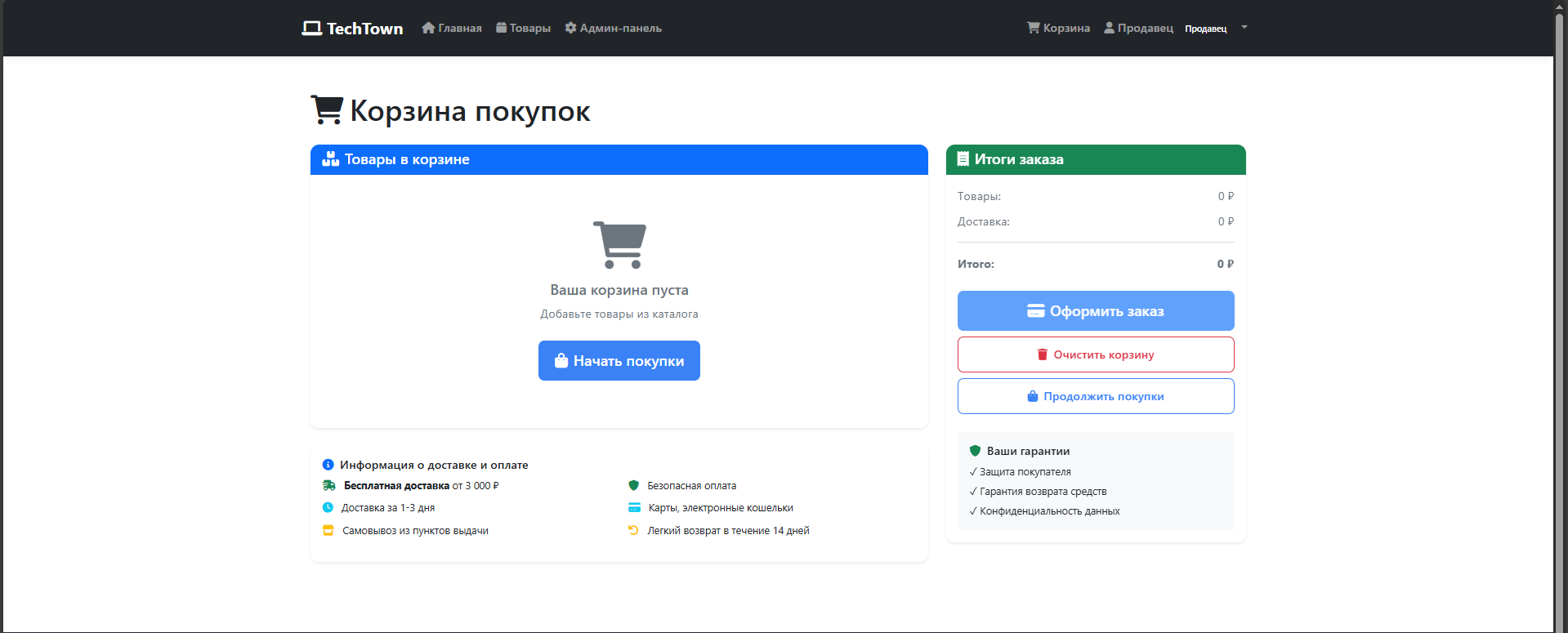


Рис.30. Корзина покупок

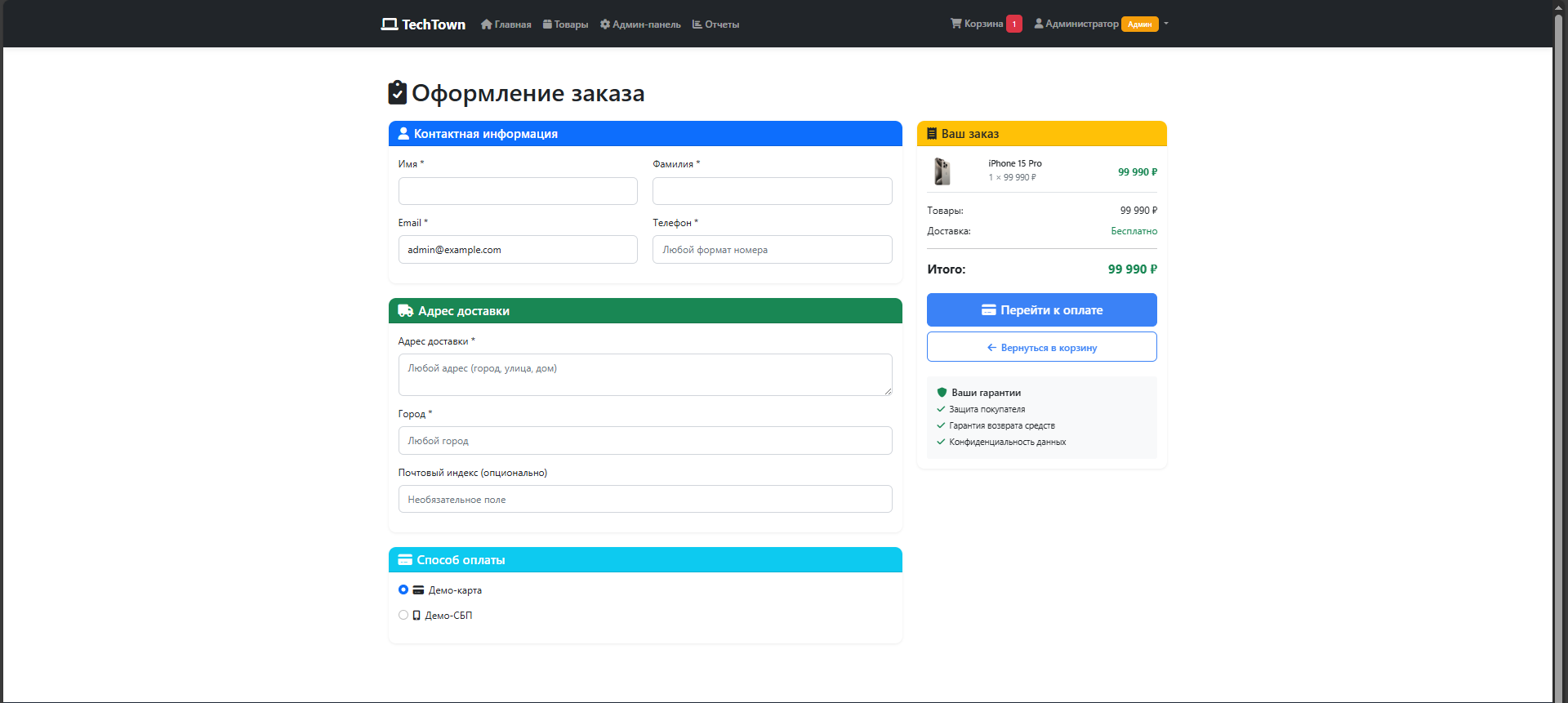
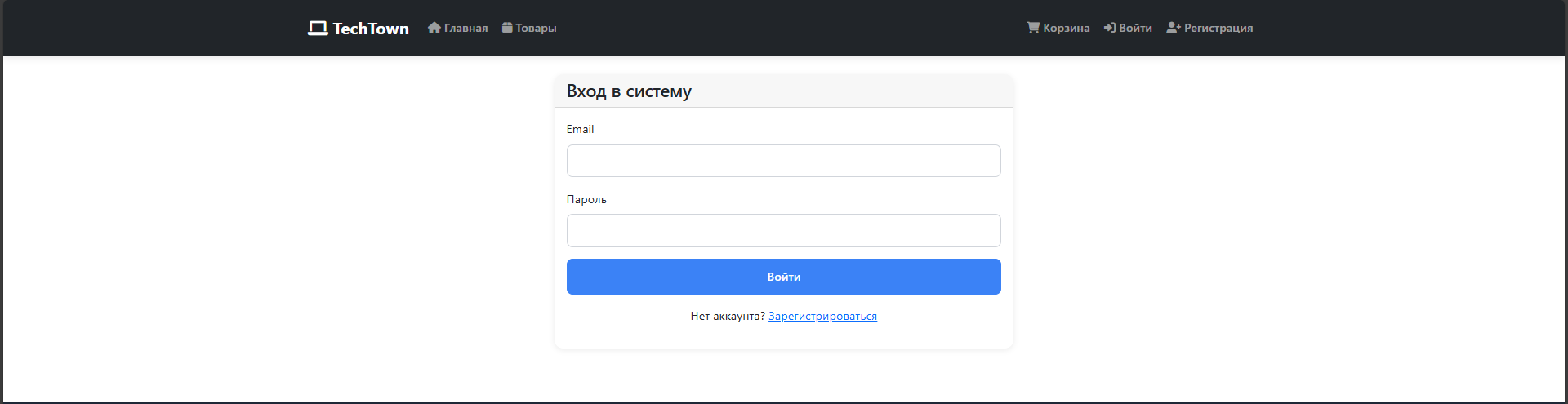


Рис.31. Оформление заказа

  
Рис.32. Авторизация

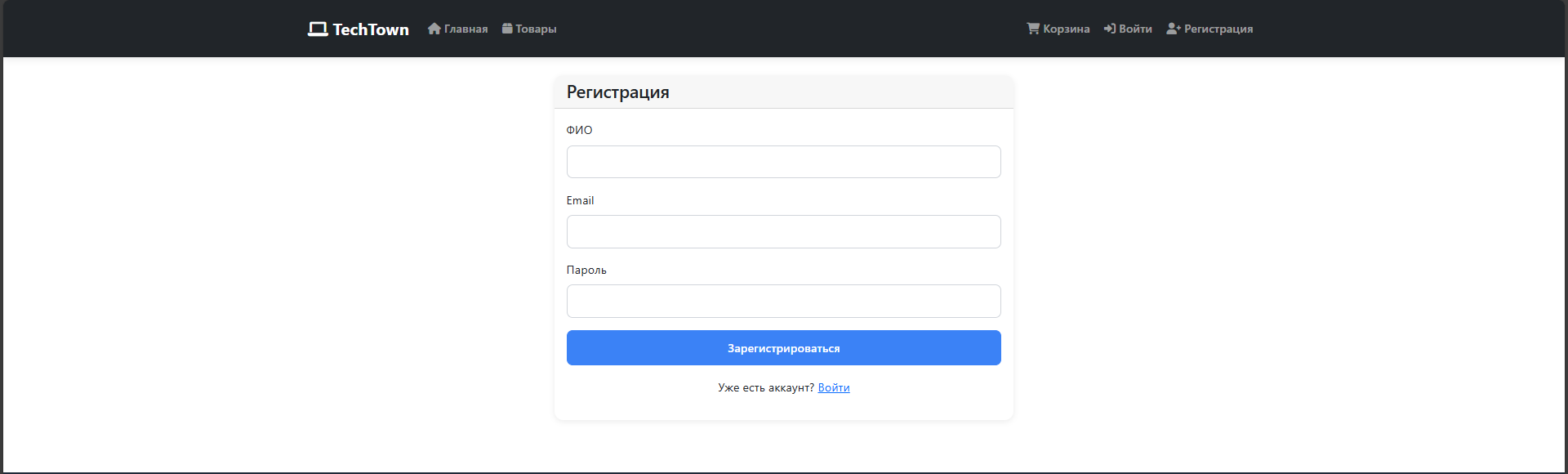


Рис.33. Регистрация

Во всех разделах приложения используется единая дизайн-система. Это обеспечивает целостность и удобство пользовательского опыта.

Листинг основных модулей:

* admin\_dashboard.html – основной html файл админ-панели.
* admin\_products.html – основной html файл добавления товаров.
* base.html – основной html файл для дизайна и навигации.
* cart.html – основной html файл корзины.
* checkout.html – основной html файл для оформления заказа и платежной системы.
* index.html – основной html файл главной страницы.
* login.html – основной html файл авторизации.
* products.html – основной html файл каталога товаров.
* register.html – основной html файл регистрации.
* reports.html – основной html файл отчетов.
* reviews.html – основной html файл отзывов.

**Глава 10. Безопасность**

Данный раздел описывает систему защиты от распределенных атак типа «отказ в обслуживании (DDoS)».

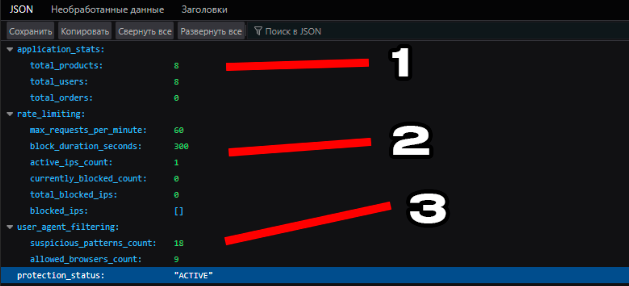


Рис.34. Статус защиты



Рис.35. Тесты DDoS-защиты (1)

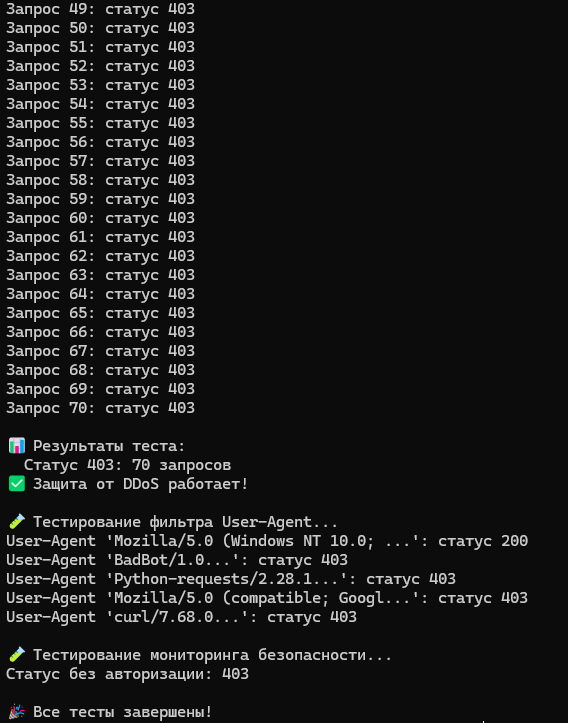


Рис.36.Тесты DDoS защиты (2)

Эта многоуровневая система защиты обеспечивает высокую доступность интернет-магазина даже при интенсивных нагрузках и автоматизированных атаках. Есть возможность просмотра статуса продуктов (1), ограничения запросов (2), фильтрация User-Agent и активнойсть защиты (3)

Листинг основных модулей:

* class RateLimiter – система ограничения частоты запросов.
* class UserAgentFilter – фильтрация пользовательских агентов.
* @app.get("/admin/security-status") – основной эндпоинт для DDoS-защиты.
* @app.get("/test/ddos-simulation") – эндпоинт для демонстрации работы защиты (имитирует быстрые запросы).
* @app.get("/test/suspicious-agent") – эндпоинт для тестирования фильтра User-Agent.

**Заключение**

В ходе проекта была успешно разработана и протестирована полнофункциональная сетевая серверная часть для интернет-магазина электроники «TechTown».

Реализованный функционал включает в себя, архитектура и база данных, система ролей, бизнес-логика, панель управления, пользовательский интерфейс и безопасность от DDoS-атак.

Для дальнейшего развития «TechTown» можно рассмотреть следующие улучшения:

* Внедрение системы рекомендаций товаров на основе машинного обучения.
* Разработка мобильного приложения для iOS и Android.
* Реализация более сложной системы скидок и программ лояльности.

Таким образом, в рамках проекта была достигнута поставленная цель: создано работоспособное, безопасное и масштабируемое серверное приложение, готовое к дальнейшему развитию и использованию в образовательных или коммерческих целях.

**Библиографический список**

1. Рамти, А. «FastAPI. Путь к созданию современных высокопроизводительных веб-приложений на Python». – М.: ДМК Пресс, 2023. – 358 с.
2. Гарсия, Х., Лангрео, П. «FastAPI. Разработка веб-сервисов на Python». – СПб.: Питер, 2024. – 304 с.
3. Грудо, М. «Python и базы данных. SQL, ORM и FastAPI». – СПб.: БХВ-Петербург, 2023. – 480 с.
4. Лусиано, Р. «Pydantic на практике. Валидация данных в Python и FastAPI». – М.: ДМК Пресс, 2024. – 256 с.
5. Мартин, Р.С. «Чистая архитектура. Искусство разработки программного обеспечения». – СПб.: Питер, 2022. – 384 с.
6. Официальная документация FastAPI [Электронный ресурс]. – Режим доступа: https://fastapi.tiangolo.com (дата обращения: 25.05.2024).
7. Официальная документация SQLAlchemy (ORM) [Электронный ресурс]. – Режим доступа: https://docs.sqlalchemy.org (дата обращения: 25.05.2024).
8. Официальная документация Pydantic [Электронный ресурс]. – Режим доступа: https://docs.pydantic.dev (дата обращения: 25.05.2024).