ОБЛАСТНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ПРОФЕССИОНАЛЬНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

**«ТОМСКИЙ ЭКОНОМИКО-ПРОМЫШЛЕННЫЙ КОЛЛЕДЖ»**

К ЗАЩИТЕ ДОПУЩЕН

Директор ОГБПОУ «ТЭПК»

\_\_\_\_\_\_\_\_\_\_\_\_ / О.Н. Пояркова

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2025 г.

**МОДЕРНИЗАЦИЯ И РЕДИЗАЙН ВНУТРЕННЕГО ПОРТАЛА ДЛЯ РАБОТЫ РАСПРЕДЕЛЕННЫХ ПОДРАЗДЕЛЕНИЙ ООО "ЧИРЧИК"**

ДИПЛОМНЫЙ ПРОЕКТ

*Специальность 09.02.07 Информационные системы и программирование*

Выполнил:

Студент IV курса группы 1912с

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / Тимофеев А.В.

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2025 г.

Проверил:

Руководитель ДП

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / Летягина И.Е.

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2025 г.

Томск 2025

Оглавление

[ВВЕДЕНИЕ 4](#_Toc201062898)

[ТЕХНОЛОГИИ И ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ 5](#_Toc201062899)

[Языки для разработки 5](#_Toc201062900)

[Программное обеспечение 5](#_Toc201062901)

[ТЕХНИЧЕСКОЕ ЗАДАНИЕ 6](#_Toc201062902)

[Назначение и структура документа 6](#_Toc201062903)

[Требования к функциональности 7](#_Toc201062904)

[Аутентификация 8](#_Toc201062905)

[Пользователи 8](#_Toc201062906)

[Администраторы 9](#_Toc201062907)

[Ролевая модель 10](#_Toc201062908)

[Ограничения 11](#_Toc201062909)

[Технологический стек 11](#_Toc201062910)

[ТЕХНИЧЕСКАЯ РЕАЛИЗАЦИЯ 12](#_Toc201062911)

[Концептуальная модель данных 12](#_Toc201062912)

[Логическая модель данных 14](#_Toc201062913)

[Физическая модель данных 14](#_Toc201062914)

[Словарь данных 16](#_Toc201062915)

[Структура базы данных 22](#_Toc201062916)

[Диаграмма прецедентов 23](#_Toc201062917)

[Диаграммы последовательностей 25](#_Toc201062918)

[Общие диаграммы-дополнения 25](#_Toc201062919)

[Авторизационные функции 26](#_Toc201062920)

[Работа с новостями 27](#_Toc201062921)

[Работа с контактами 29](#_Toc201062922)

[Работа с данными для входа 30](#_Toc201062923)

[Работа с полезными ссылками 31](#_Toc201062924)

[Работа с брендами 33](#_Toc201062925)

[Работа с юридическими лицами 34](#_Toc201062926)

[Работа с прайс-листами 35](#_Toc201062927)

[Макеты страниц 37](#_Toc201062928)

[РЕАЛИЗАЦИЯ ВЕБ-САЙТА 44](#_Toc201062929)

[Пользовательская часть 44](#_Toc201062930)

[Административная часть 77](#_Toc201062931)

[РУЧНОЕ ТЕСТИРОВАНИЕ ВЕБ-ПОРТАЛА 90](#_Toc201062932)

[РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ 95](#_Toc201062933)

[РУКОВОДСТВО АДМИНИСТРАТОРА 97](#_Toc201062934)

[ЗАКЛЮЧЕНИЕ 99](#_Toc201062935)

[СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ 100](#_Toc201062936)

[ПРИЛОЖЕНИЕ 1 101](#_Toc201062937)

# ВВЕДЕНИЕ

Современные аптеки сталкиваются с необходимостью оперативного взаимодействия между сотрудниками, управления внутренними процессами и предоставления актуальной информации в удобной форме. При этом большинство существующих решений либо слишком громоздки и перегружены функционалом, либо требуют постоянного доступа к внешним серверам и базам данных, что не всегда возможно в условиях ограниченной IT-инфраструктуры.

Разработка специализированного внутреннего веб-сайта для сотрудников аптек позволяет централизовать важную информацию, упростить рутинные операции и сократить количество ошибок, связанных с человеческим фактором. Проект ориентирован на простоту интерфейса, безопасность доступа, и отсутствие зависимости от сторонних API и баз данных. Это делает его особенно актуальным для компаний, которым необходимо контролировать внутренние процессы без внедрения сложных информационных систем.

**Описание предметной области**

Предметной областью является модернизация внутреннего веб-портала аптечной сети, включающего:

1. Веб-сайт для сотрудников аптек, обеспечивающий доступ к внутренней информации.

Портал охватывает ключевые аспекты внутренней коммуникации и организации рабочих процессов.

**Цель** **дипломного проекта** – Модернизация и редизайн существующего веб-портала для сотрудников аптечной сети.

Для достижения поставленной цели необходимо выполнить следующие **задачи**:

1. Разработать техническое задание веб-портала.
2. Разработать прототип интерфейса с учётом удобства и минимизации ошибок при работе.
3. Добавить разделы для отображения внутренних новостей, полезных ссылок, данных для входа к ним и контактной информации.
4. Разработать ручные тесты и провести тестирование веб-портала.

# ТЕХНОЛОГИИ И ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ

## Языки для разработки

**HTML / CSS** — базовые технологии для создания веб-страниц.  
HTML использовался для формирования структуры страниц, а CSS — для их визуального оформления и адаптивной стилизации.

**JavaScript —** основной язык программирования на стороне клиента, отвечающий за интерактивность пользовательского интерфейса и обработку событий в реальном времени.

**PHP —** серверный язык программирования, применявшийся для обработки данных, работы с формами, логикой взаимодействия с базой данных и генерацией динамического HTML-контента. Обеспечивает связку между интерфейсом и серверной частью.

## Программное обеспечение

**Visual Studio Code** — лёгкий, но функциональный редактор кода от Microsoft. Использовался для написания и редактирования HTML, CSS, JavaScript и PHP-кода.

**Draw.io (diagrams.net)** — инструмент для создания диаграмм и визуального моделирования архитектуры системы, бизнес-процессов и структуры базы данных.

**WinSCP** — программа для безопасной передачи файлов по протоколам SFTP, SCP и FTP. Применялась для загрузки готового веб-приложения на сервер, а также для синхронизации локальных изменений с серверной частью проекта.

# ТЕХНИЧЕСКОЕ ЗАДАНИЕ

## Назначение и структура документа

Настоящий документ предназначен для описания требований к разработке внутреннего веб-портала, используемого в работе распределённых подразделений ООО «Чирчик». Документ используется как основа для проектирования, разработки, тестирования и последующего внедрения программного продукта.

Структура технического задания:

1. Требования к функциональности
   * Аутентификация
   * Пользователи
   * Администраторы
2. Ролевая модель
3. Ограничения
4. Технологический стек

## Требования к функциональности

Функционал веб-портала разделен на две основные категории:

* **Администратор** управляет новостями, опросами, прайс-листами, городами, брендами, юридическими лицами, полезными ссылками, контактами. Подробнее на Рисунок 1.
* **Пользователь** может искать препараты, просматривать новости, полезные ссылки и контакты, оставлять комментарии к новости, также управлять данными для входа к полезным ссылкам. Подробнее на Рисунок 2.

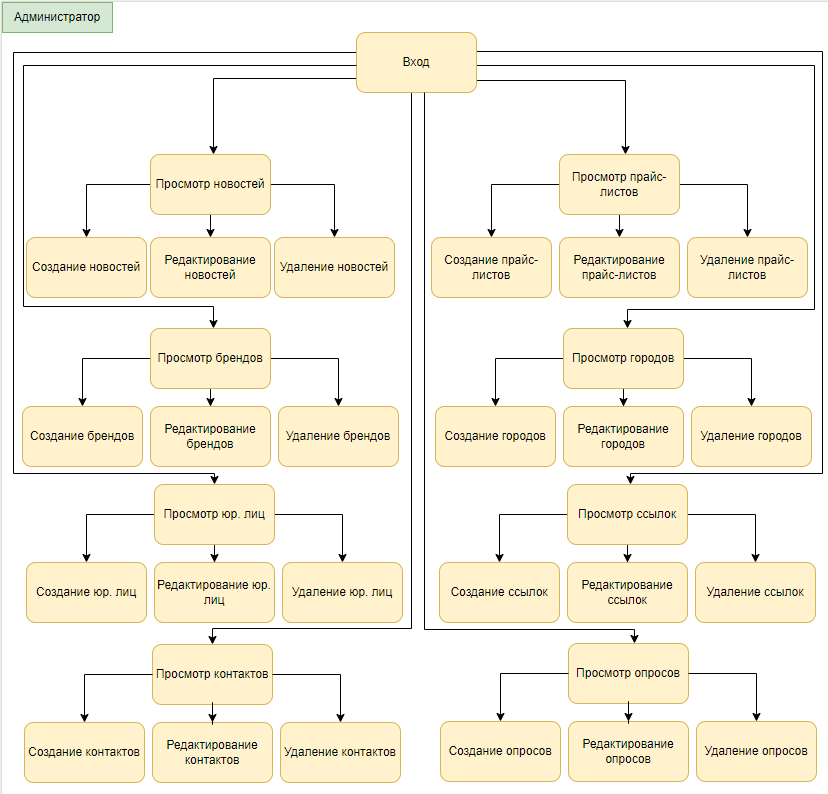


Рисунок – Функционал администратора

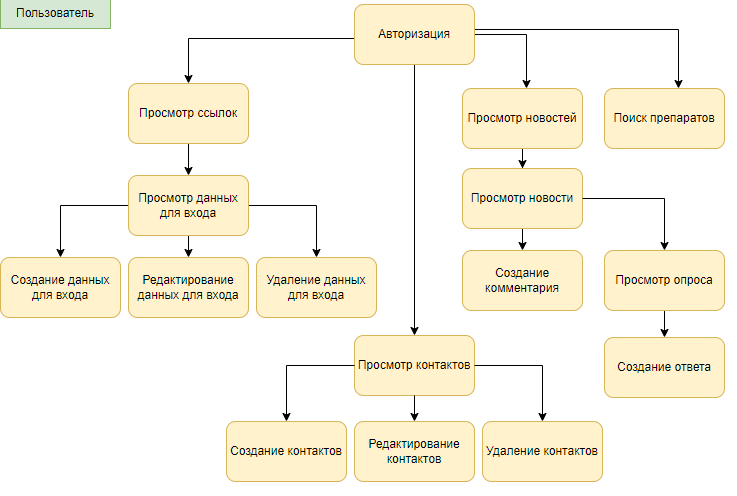


Рисунок – Функционал пользователя

### Аутентификация

* + Обычные пользователи автоматически идентифицируются по IP-адресу, который определяет, из какой аптеки осуществляется доступ. Каждый пользователь привязан к своей аптеке и обладает соответствующими правами.
  + Администраторы проходят авторизацию через логин и пароль, получая доступ к расширенному функционалу управления системой.

### Пользователи

1. Управление данными для входа:
   * Создание данных для входа.
   * Удаление и изменение данных для входа, созданных в этой аптеке.
   * Просмотр списка данных для входа, созданных в этой аптеке.
2. Просмотр полезных ссылок:
   * Просмотр списка полезных ссылок и их описания.
   * Быстрый переход по ссылкам.
3. Просмотр контактов:
   * Просмотр номеров телефона и их владельцев с дополнительной информацией.
   * Создание контактов.
   * Удаление и изменение контактов, созданных в этой аптеке.
4. Просмотр новостей:
   * Просмотр всех новостей и детальной информации о них.
   * Создание комментариев к новостям с возможностью прикрепления файлов.
5. Поиск препаратов:
   * Поиск лекарств по названию.
   * Фильтрация результатов по городу и бренду аптеки.

### Администраторы

1. Уметь всё что пользователь
2. Управление городами:
   * Создание городов.
   * Удаление и изменение городов.
3. Управление брендами:
   * Создание брендов.
   * Удаление и изменение брендов.
4. Управление юридическими лицами:
   * Создание юридических лиц.
   * Удаление и изменение юридических лиц.
5. Управление ссылками:
   * Создание ссылок.
   * Удаление и изменение ссылок.
6. Управление новостями:
   * Создание новостей и опросов к ним.
   * Удаление и изменение новостей.

## Ролевая модель

Администратор – отвечает за управление системой, добавление и обновление новостей, контактов и т.д.

Пользователь – основной пользователь сервиса, который ищет препараты, читает новости, оставляет к ним комментарии и сохраняет данные для входа.

Более структурированная и подробная информация представлена в Таблица 1.

Таблица – Ролевая модель системы

|  |  |  |  |
| --- | --- | --- | --- |
| Раздел | Фичи | Админ | Пользователь |
| Авторизация | Авторизация | нет | да |
| Выход | да | нет |
| Новости | Просмотр списка новостей | да | да |
| Добавление новостей | да | нет |
| Редактирование данных новостей | да | нет |
| Просмотр данных новости | да | да |
| Удаление новостей | да | нет |
| Создание комментариев | да | да |
| Данные для входа | Просмотр списка данных для входа | да | да(своих) |
| Добавление данных для входа | да | да(своих) |
| Редактирование данных для входа | да | да(своих) |
| Удаление данных для входа | да | да(своих) |
| Контакты | Просмотр списка контактов | да | да |
| Добавление контактов | да | нет |
| Редактирование данных контактов | да | нет |
| Удаление контактов | да | нет |
| Полезные ссылки | Просмотр списка ссылок | да | да |
| Добавление ссылок | да | нет |
| Редактирование данных ссылок | да | нет |
| Удаление ссылок | да | нет |
| Бренды | Просмотр списка брендов | да | да |
| Добавление брендов | да | нет |
| Редактирование брендов | да | нет |
| Удаление брендов | да | нет |
| Юридические лица | Просмотр списка юр. лиц | да | нет |
| Добавление юр. лиц | да | нет |
| Редактирование юр. лиц | да | нет |
| Удаление юр. лиц | да | нет |
| Прайс-листы | Просмотр списка прайс-листов | да | нет |
| Добавление прайс-листов | да | нет |
| Редактирование прайс-листов | да | нет |
| Удаление прайс-листов | да | нет |

## Ограничения

1. Авторизация пользователей ограничена IP-адресами

* Обычные пользователи автоматически идентифицируются по IP-адресу, определяя их рабочее место в аптечной сети.
* Вход через логин/пароль не поддерживается для пользователей, но требуется для администраторов.

1. Отсутствие API

* Система не взаимодействует с внешними API и не поддерживает интеграцию с другими сервисами.

## Технологический стек

**Серверная часть**

* PHP (без фреймворков)

**Клиентская часть**:

* HTML
* CSS
* JavaScript

**База данных:**

* MySQL

# ТЕХНИЧЕСКАЯ РЕАЛИЗАЦИЯ

## Концептуальная модель данных

Концептуальная модель — это абстрактное представление предметной области, отражающее основные сущности, их свойства и взаимосвязи между ними, без учёта технических деталей реализации. Она служит для понимания и описания структуры данных и бизнес-процессов на высоком уровне. Представлена на Рисунок 3.

Описание концептуальной модели:

1. **Пользователи**: содержит информацию о пользователях. Может содержать поля, такие как - Имя, Логин, Пароль.
2. **Контакты**: содержит информацию о контактах. Может содержать поля, такие как – ID типа контакта, Имя, Описание, Номер телефона, Электронная почта, Важность, Видимость.
3. **Типы контактов**: содержит информацию об типах контактов. Может содержать поля, такие как – Возможность добавлять пользователем, Название, Видимость, ID города.
4. **Города**: содержит информацию о городах. Может содержать поля, такие как – Название Ip города, Видимость, Какие города видит.
5. **Район**: содержит информацию о районах. Может содержать поля, такие как – Название, Видимость, ID города.
6. **Партнер**: содержит информацию о партнерах. Может содержать поля, такие как – ID города, ID района, Название, Статус, Код, Номер телефона, Подсеть, ID фирмы, ID бренда, ID типа партнера.
7. **Фирмы**: содержит информацию о фирмах. Может содержать поля, такие как – Название.
8. **Бренды**: содержит информацию о брендах. Может содержать поля, такие как – Название.
9. **Типы партнера**: содержит информацию о типах партнера. Может содержать поля, такие как – Название.
10. **Ссылки**: содержит информацию о ссылках. Может содержать поля, такие как – ID типа ссылки, Название, Ссылка, Описание, Возможность добавлять данные для входа к ссылке.
11. **Типы ссылки**: содержит информацию о типах ссылки. Может содержать поля, такие как – Название.
12. **Данных для входа ссылки**: содержит информацию об данных для входа ссылки. Может содержать поля, такие как – Логин, Пароль, Описание, Видимость.
13. **Посты**: содержит информацию о постах. Может содержать поля, такие как – Заголовок, Краткое описание, Полное описание, Видимость, Видимость комментариев, Возможность писать комментарии, Прикрепленные файлы, Прикрепленные фотографии, ID важности поста, Время важности, ID типа поста, ID пользователя.
14. **Логи**: содержит информацию об логах. Может содержать поля, такие как – ID пользователя, ID поста, Действие, Дата действия.
15. **Комментарии**: содержит информацию об комментариях. Может содержать поля, такие как – ID пользователя, ID поста, Текст комментария, Дата создания, Публичность.
16. **Важность поста**: содержит информацию об важностях поста. Может содержать поля, такие как – Название, Код.
17. **Тип поста**: содержит информацию об типах поста. Может содержать поля, такие как – Название, Код.
18. **Ответы пользователя**: содержит информацию об ответах пользователя. Может содержать поля, такие как – ID пользователя, ID варианта ответа на вопрос, ID вопроса, Свой ответ.
19. **Варианты ответа на вопрос**: содержит информацию об вариантах ответа на вопрос. Может содержать поля, такие как – ID вопроса, Ответ на вопрос.
20. **Вопросы**: содержит информацию об вопросах. Может содержать поля, такие как – ID типа вопроса, Текст вопроса, ID поста.
21. **Типы вопросов**: содержит информацию об типах вопросов. Может содержать поля, такие как – Название.

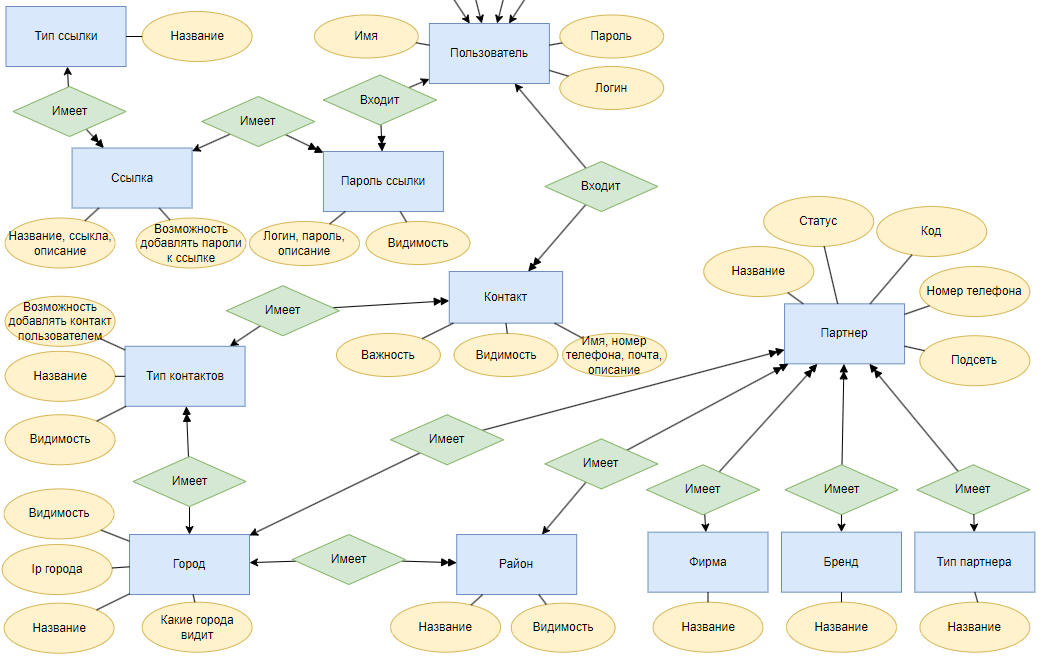
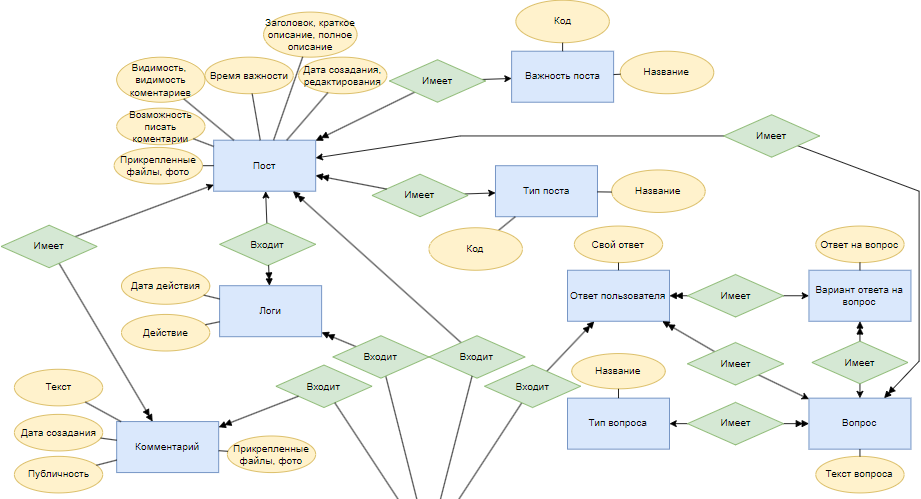


Рисунок 3 - Концептуальная модель

## Логическая модель данных

Логическая модель — это более детализированное представление данных, которое описывает структуру базы данных с учётом правил нормализации, типов данных и связей между таблицами, но остаётся независимой от конкретной СУБД и физической реализации. Представлена на Рисунок 4.

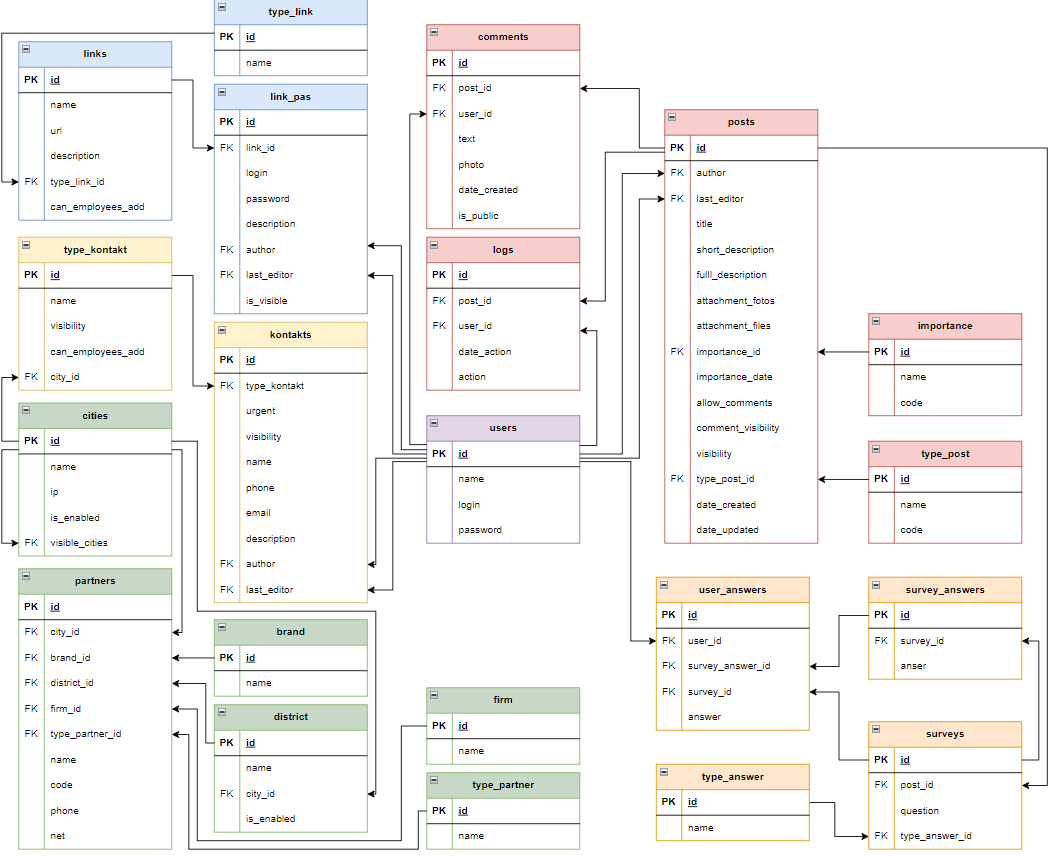


Рисунок 4 - Логическая модель

## Физическая модель данных

Физическая модель — это конкретное описание структуры данных с учётом особенностей выбранной системы управления базами данных (СУБД), включая специфику хранения, индексов, оптимизации и технических параметров реализации. Представлена на Рисунок 5.

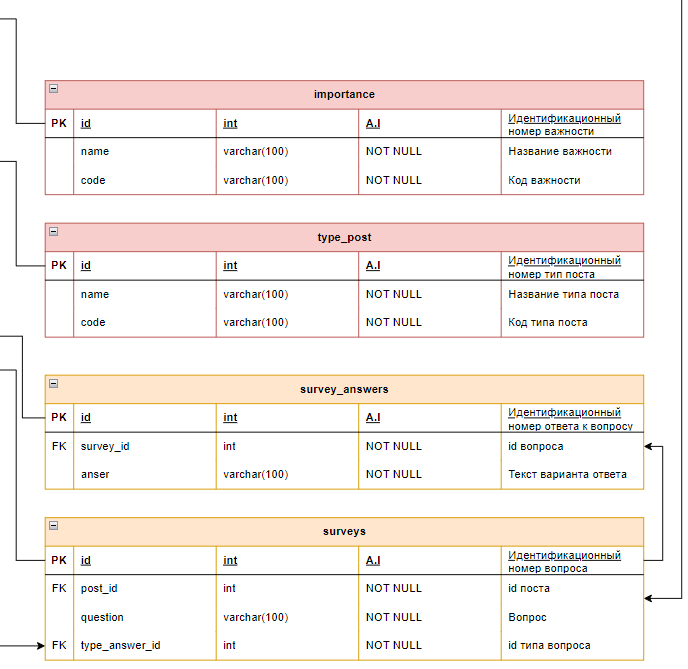
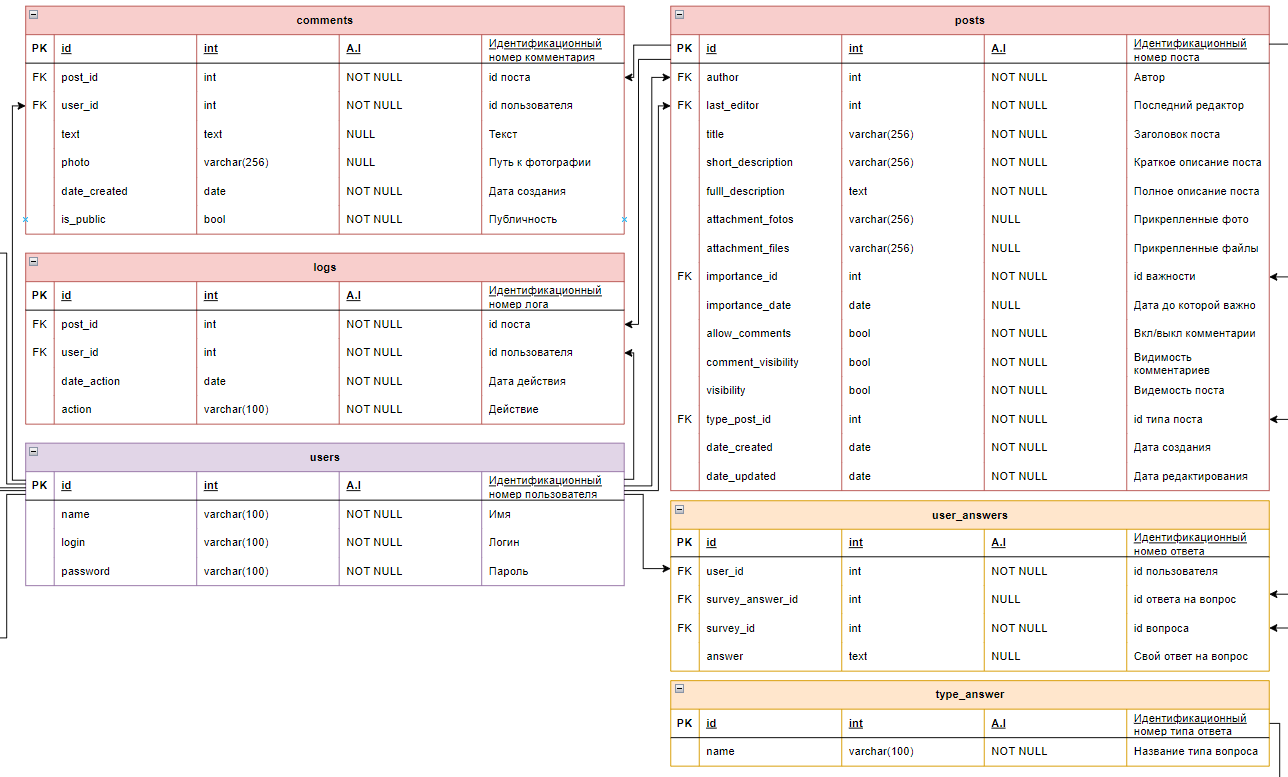
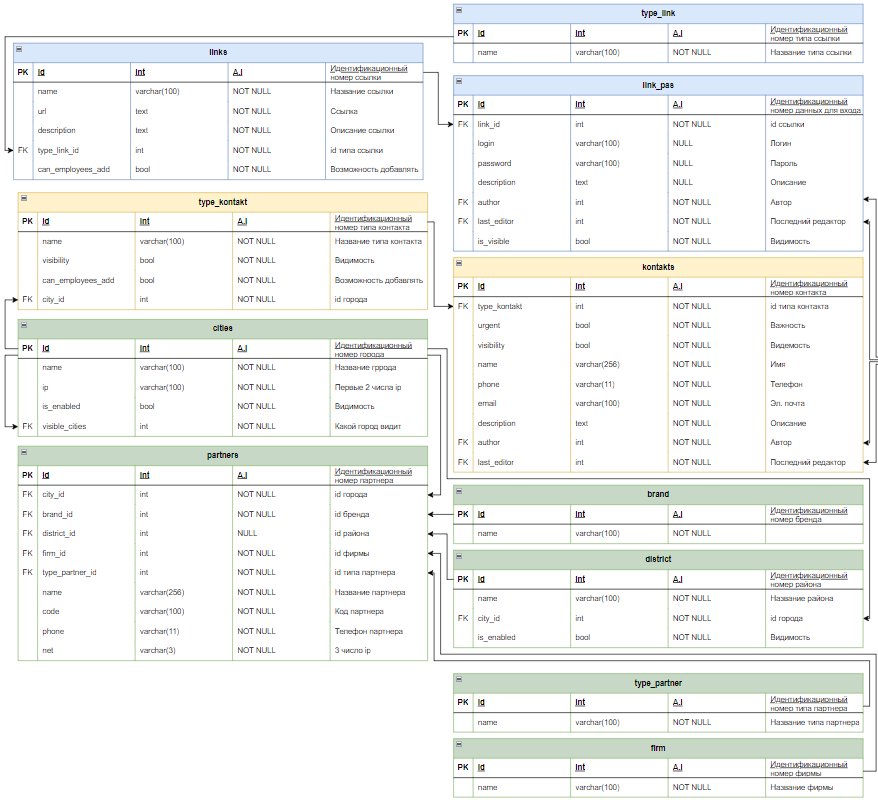


Рисунок 5 - Физическая модель

## Словарь данных

Словарь данных — это справочник, содержащий подробное описание всех элементов данных (полей, таблиц, сущностей) используемой информационной системы, включая их названия, типы, назначение и допустимые значения. Он служит для унификации понимания данных всеми участниками проекта.

**users (Пользователи)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Атрибут** | **Тип** | **Ограничение** | **Описание** |
| id | biginteger | PK, авто инкремент | Идентификатор пользователя |
| name | varchar(255) | Обязательное | Имя |
| login | varchar(255) | Обязательное | Логин |
| password | varchar(255) | Обязательное | Пароль |

**kontakts (Контакты)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Атрибут** | **Тип** | **Ограничение** | **Описание** |
| id | biginteger | PK, авто инкремент | Идентификатор контакта |
| name | varchar(255) | Обязательное | Название |
| phone | varchar(255) | Обязательное | Номер телефона |
| email | varchar(255) | Обязательное | Эл. Почта |
| desctiption | text | Обязательное | Описание |
| author | int | FK, Обязательное | Идентификатор пользователя |
| last\_editor | int | FK, Обязательное | Идентификатор пользователя |
| type\_kontakt | int | FK, Обязательное | Идентификатор типа контакта |
| urgent | bool | Обязательное | Важность |

**type\_kontakt (Типы контактов)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Атрибут** | **Тип** | **Ограничение** | **Описание** |
| id | biginteger | PK, авто инкремент | Идентификатор типа контакта |
| name | varchar(255) | Обязательное | Название типа контакта |
| visibility | bool | Обязательное | Видимость |
| can\_employees\_add | bool | Обязательное | Возможность добавлять сотрудникам |
| city\_id | int | FK, Обязательное | Идентификатор города |

**cities (Города)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Атрибут** | **Тип** | **Ограничение** | **Описание** |
| id | biginteger | PK, авто инкремент | Идентификатор города |
| name | varchar(100) | Обязательное | Название города |
| ip | varchar(100) | Обязательное | Первые 2 числа IP |
| is\_enabled | bool | Обязательное | Видимость |
| visible\_cities | int | FK, Обязательное | Какой город видит |

**links (Ссылки)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Атрибут** | **Тип** | **Ограничение** | **Описание** |
| id | biginteger | PK, авто инкремент | Идентификатор ссылки |
| name | varchar(100) | Обязательное | Название ссылки |
| url | text | Обязательное | Ссылка |
| description | text | Обязательное | Описание ссылки |
| type\_link\_id | int | Обязательное | Идентификатор типа ссылки |
| can\_employees\_add | bool | FK, Обязательное | Возможность добавлять сотрудникам |

**type\_link (Типы ссылок)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Атрибут** | **Тип** | **Ограничение** | **Описание** |
| id | biginteger | PK, авто инкремент | Идентификатор типа ссылки |
| name | varchar(100) | Обязательное | Название типа ссылки |

**link\_pas (Данные для входа)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Атрибут** | **Тип** | **Ограничение** | **Описание** |
| id | biginteger | PK, авто инкремент | Идентификатор данных для входа |
| link\_id | int | |  | | --- | |  |   FK, Обязательное | Идентификатор ссылки |
| login | varchar(100) | Обязательное | Логин |
| password | varchar(100) | Обязательное | Пароль |
| description | text | Обязательное | Описание |
| author | int | FK, Обязательное | Идентификатор пользователя |
| last\_editor | int | FK, Обязательное | Идентификатор пользователя |
| is\_visible | bool | Обязательное | Видимость |

**partners (Партнеры)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Атрибут** | **Тип** | **Ограничение** | **Описание** |
| id | biginteger | PK, авто инкремент | Идентификатор партнёра |
| city\_id | int | FK, Обязательное | Идентификатор города |
| brand\_id | int | FK, Обязательное | Идентификатор бренда |
| district\_id | int | FK, Обязательное | Идентификатор района |
| firm\_id | int | FK, Обязательное | Идентификатор фирмы |
| type\_partner\_id | int | FK, Обязательное | Идентификатор типа партнёра |
| name | varchar(256) | Обязательное | Название партнёра |
| phone | varchar(11) | Обязательное | Номер телефона |
| net | varchar(3) | Обязательное | 3 число ip |

**districts (Районы)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Атрибут** | **Тип** | **Ограничение** | **Описание** |
| id | biginteger | PK, авто инкремент | Идентификатор района |
| name | varchar(100) | Обязательное | Название района |
| city\_id | int | FK, Обязательное | Идентификатор города |
| is\_enabled | bool | Обязательное | Видимость |

**brands (Бренды)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Атрибут** | **Тип** | **Ограничение** | **Описание** |
| id | biginteger | PK, авто инкремент | Идентификатор бренда |
| name | varchar(100) | Обязательное | Название бренда |

**type\_partner (Типы партнёров)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Атрибут** | **Тип** | **Ограничение** | **Описание** |
| id | biginteger | PK, авто инкремент | Идентификатор типа партнёра |
| name | varchar(100) | Обязательное | Название типа партнёра |

**firms (Фирмы)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Атрибут** | **Тип** | **Ограничение** | **Описание** |
| id | biginteger | PK, авто инкремент | Идентификатор фирмы |
| name | varchar(100) | Обязательное | Название фирмы |

**posts (Посты)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Атрибут** | **Тип** | **Ограничение** | **Описание** |
| id | biginteger | PK, авто инкремент | Идентификатор фирмы |
| author | int | FK, Обязательное | Автор |
| last\_editor | int | FK, Обязательное | Последний редактор |
| title | varchar(256) | Обязательное | Заголовок поста |
| short\_description | varchar(256) | Обязательное | Краткое описание |
| full\_description | text | Обязательное | Полное описание |
| attachment\_fotos | varchar(256) | Обязательное | Прикреплённые фото |
| attachment\_files | varchar(256) | Обязательное | Прикреплённые файлы |
| importance\_id | int | FK, Обязательное | Идентификатор важности |
| importance\_date | date | Обязательное | Дата, до которой важно |
| allow\_comments | bool | Обязательное | Включить/выключить комментарии |
| comment\_visibility | bool | Обязательное | Видимость комментариев |
| visibility | bool | Обязательное | Видимость поста |
| type\_post\_id | int | FK, Обязательное | Идентификатор типа поста |
| date\_created | date | Обязательное | Дата создания |
| date\_updated | date | Обязательное | Дата редактирования |

**importance (Важность)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Атрибут** | **Тип** | **Ограничение** | **Описание** |
| id | biginteger | PK, авто инкремент | Идентификатор важности |
| name | varchar(100) | Обязательное | Название важности |
| code | varchar(100) | Обязательное | Код важности |

**type\_post (Типы постов)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Атрибут** | **Тип** | **Ограничение** | **Описание** |
| id | biginteger | PK, авто инкремент | Идентификатор типа поста |
| name | varchar(100) | Обязательное | Название типа поста |
| code | varchar(100) | Обязательное | Код типа поста |

**comments (Комментарии)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Атрибут** | **Тип** | **Ограничение** | **Описание** |
| id | biginteger | PK, авто инкремент | Идентификатор комментария |
| post\_id | int | FK, Обязательное | id поста |
| user\_id | int | FK, Обязательное | id пользователя |
| text | text | Обязательное | Текст |
| photo | varchar(256) | NULL | Путь к фотографии |
| date\_created | date | Обязательное | Дата создания |
| is\_public | bool | Обязательное | Публичность |

**logs (Логи)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Атрибут** | **Тип** | **Ограничение** | **Описание** |
| id | biginteger | PK, авто инкремент | Идентификатор лога |
| post\_id | int | FK, Обязательное | id поста |
| user\_id | int | FK, Обязательное | id пользователя |
| date\_action | date | Обязательное | Дата действия |
| action | varchar(100) | Обязательное | Действие |

**surveys (Опросы)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Атрибут** | **Тип** | **Ограничение** | **Описание** |
| id | biginteger | PK, авто инкремент | Идентификатор вопроса |
| post\_id | int | FK, Обязательное | id поста |
| question | varchar(100) | Обязательное | Вопрос |
| type\_answer\_id | int | FK, Обязательное | Тип ответа |

**type\_answer (Типы ответов)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Атрибут** | **Тип** | **Ограничение** | **Описание** |
| id | biginteger | PK, авто инкремент | Идентификатор типа ответа |
| name | varchar(100) | Обязательное | Название типа вопроса |

**survey\_answers (Ответы на вопросы)**

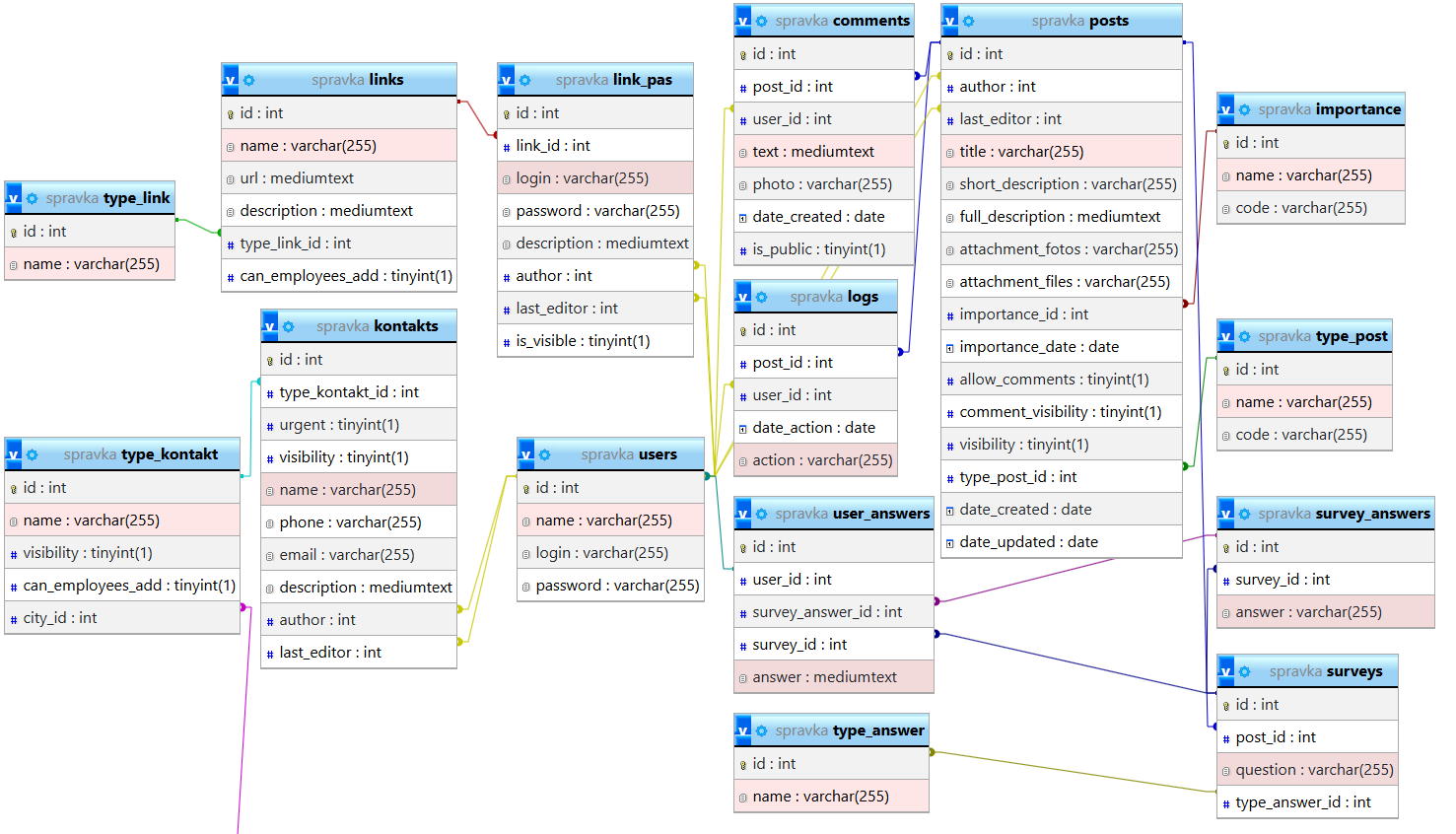
|  |  |  |  |
| --- | --- | --- | --- |
| **Атрибут** | **Тип** | **Ограничение** | **Описание** |
| id | biginteger | PK, авто инкремент | Идентификатор ответа |
| survey\_id | int | FK, Обязательное | id вопроса |
| anser | varchar(100) | Обязательное | Текст варианта ответа |

**user\_answers (Ответы пользователя)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Атрибут** | **Тип** | **Ограничение** | **Описание** |
| id | biginteger | PK, авто инкремент | Идентификатор ответа пользователя |
| user\_id | int | FK, Обязательное | id пользователя |
| survey\_answer\_id | int | FK, NULL | id ответа на вопрос |
| survey\_id | int | FK, Обязательное | id вопроса |
| answer | text | NULL | Свой ответ на вопрос |

## Структура базы данных

На основе физической модели и словаря данных была сформирована итоговая структура базы данных, представленная на Рисунке 6. Она реализована в MySQL и отражает основные связи между таблицами и ключевые атрибуты, необходимые для работы системы.



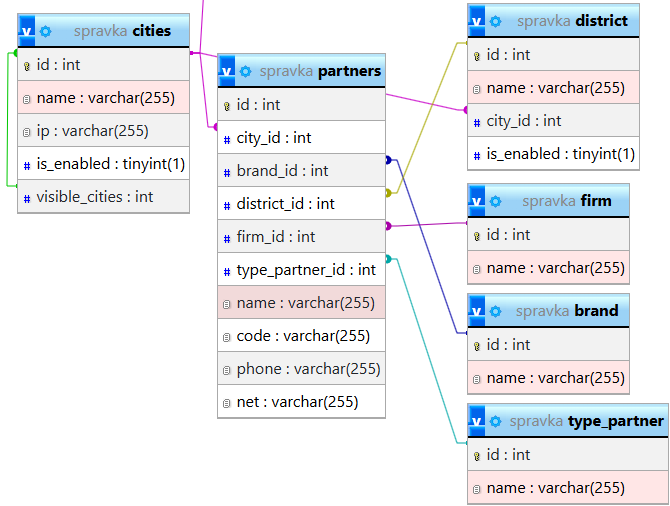


Рисунок 6 – Структура базы данных

Исходный SQL-код создания таблиц, а также набор тестовых данных размещены в Приложении 1.

## Диаграмма прецедентов

Диаграмма прецедентов используется для моделирования функциональных требований системы и взаимодействия между пользователями и системой. Она помогает определить основные сценарии использования системы и её ключевые функции. Состоит из следующих ключевых элементов:

* **Акторы** — роли пользователей или внешних систем, взаимодействующих с системой.
* **Прецеденты** — функциональные возможности или сценарии, которые система предоставляет.
* **Связи** — взаимодействия между акторами и прецедентами.
* **Включение и расширение** — отношения между прецедентами для повторного использования или добавления дополнительного поведения.

В системе выделено 2 актора: пользователь и администратор. 7 подсистем прецедентов, как общие объекты случаев использования: управление новостями, данными для входа, контактами, полезными ссылками, брендами, юр. лицами, прайс-листами. Диаграмма прецедентов разработана с учётом всех этих аспектов (см. Рисунок 7).

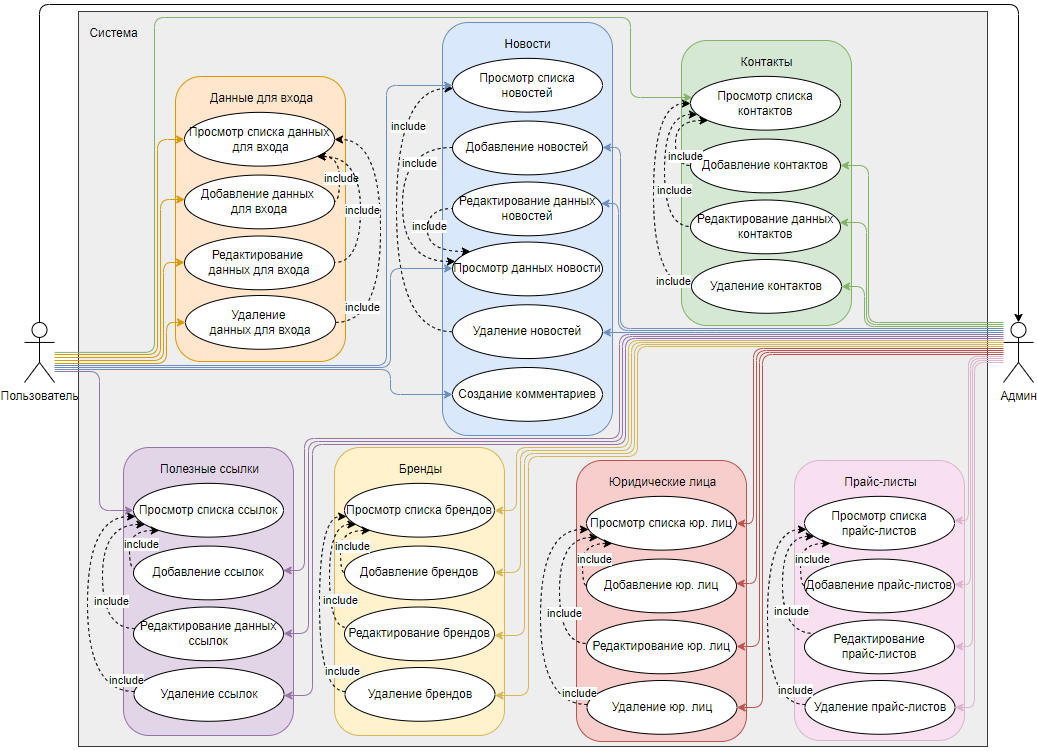


Рисунок 7. Диаграмма прецедентов

## Диаграммы последовательностей

Диаграмма последовательностей отображает взаимодействие объектов в системе в рамках определённого сценария. Она показывает порядок обмена сообщениями между объектами и их временную последовательность. Основные элементы диаграммы:

* Объекты и акторы — участники взаимодействия.
* Сообщения — вызовы методов или передача данных между объектами.
* Временная ось — вертикальная линия, показывающая порядок событий.
* Активация — период активности объекта при выполнении операции.
* Фреймы — обёртки над элементами для описания конкретного случая или цикла.

### Общие диаграммы-дополнения

Общие диаграммы последовательности не являются самостоятельными и предназначены для дополнения к другим последовательностям для разгрузки повторяющихся элементов. Обычные диаграммы ссылаются на дополнительны посредством цветной активации по цвету выхода.

Одним из общих элементов, выделенных в отдельную диаграмму-дополнение, является механизм валидации POST-запросов. Эта диаграмма используется в различных сценариях взаимодействия с системой при передаче данных.

Валидация тела POST запроса осуществляется посредником, проверяя заданные правила на ключах (см. Рисунок 8).

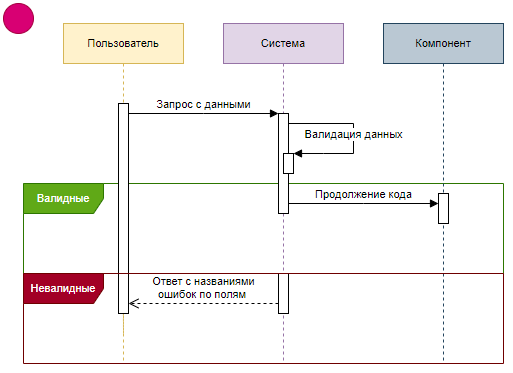


Рисунок 8. Валидация тела запроса

Проверка прав доступа начинается с проверки наличия пользователя. Если его нет, система открывает страницу авторизации, иначе запрос продолжается с данными пользователя. (см. Рисунок 9).

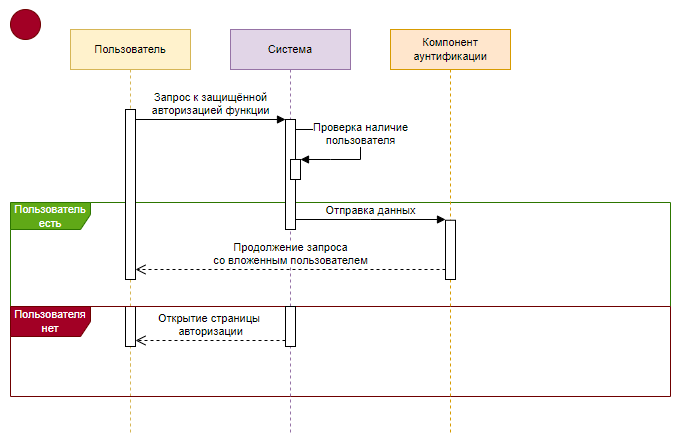


Рисунок 9. Проверка токена и внесение в контекст пользователя

### Авторизационные функции

Авторизация представлена на Рисунок 10.

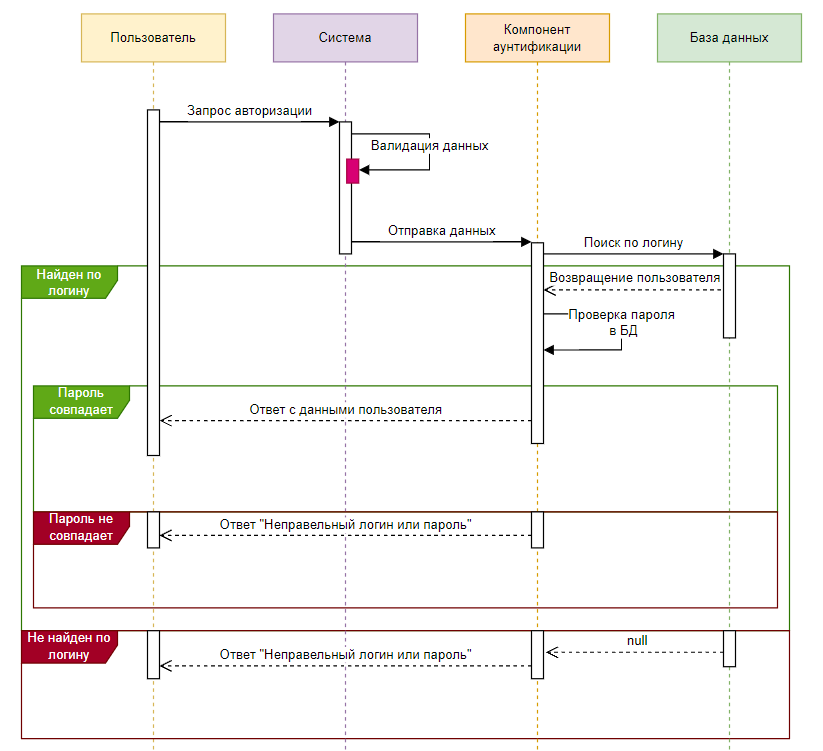


Рисунок 10. Авторизация

Выход представлен на Рисунок 11.

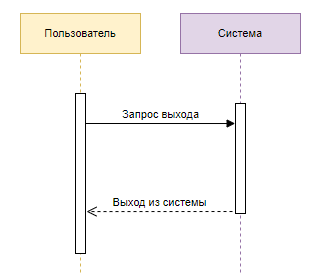


Рисунок 11. Выход

### Работа с новостями

Просмотр новостей представлен на Рисунок 12.

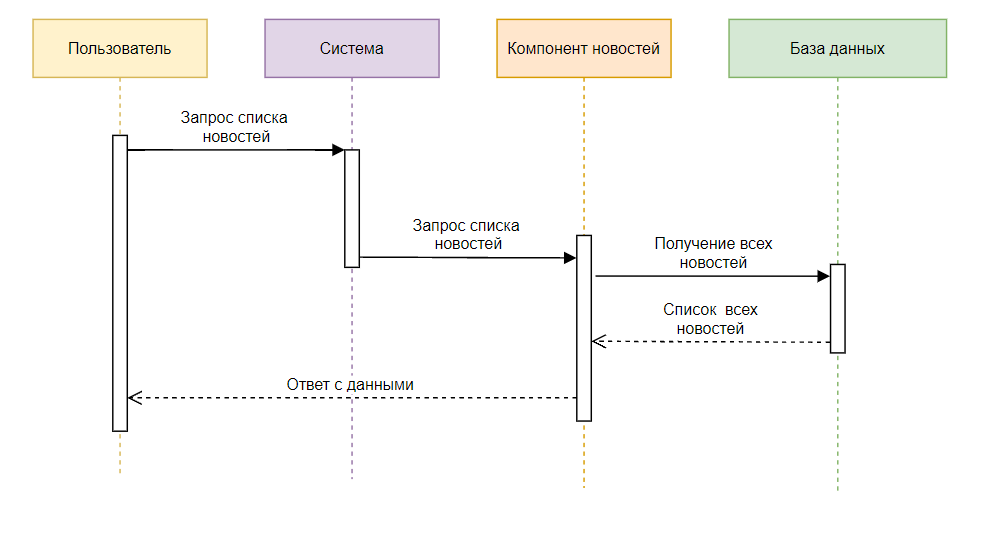


Рисунок 12 Просмотр новостей

Создание новости представлено на Рисунок 13.

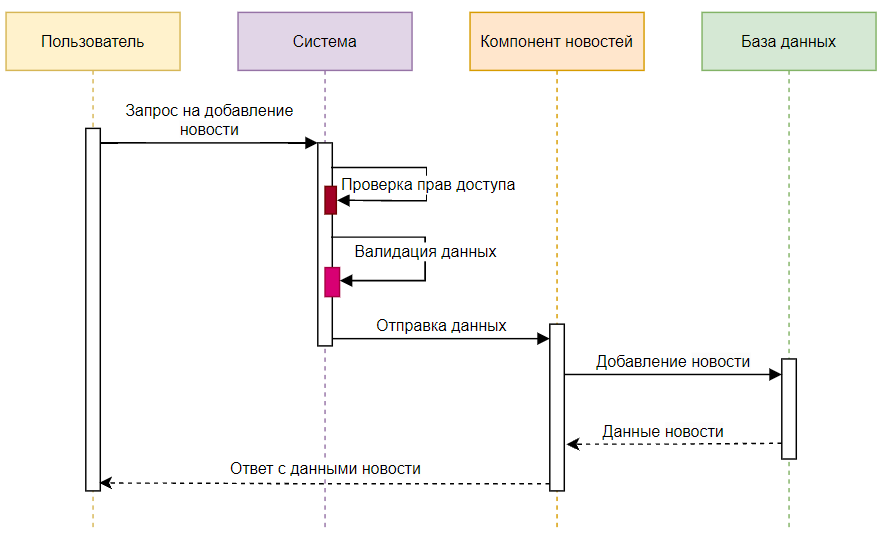


Рисунок 13 Создание новостей

Редактирование новости представлена на Рисунок 14.

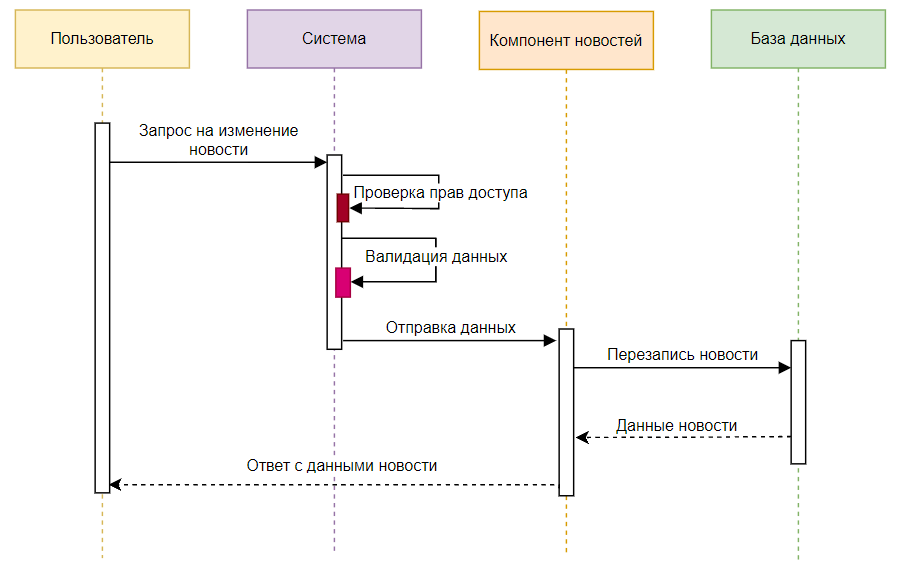


Рисунок 14 Редактирование новости

Удаление новости представлена на Рисунок 15.

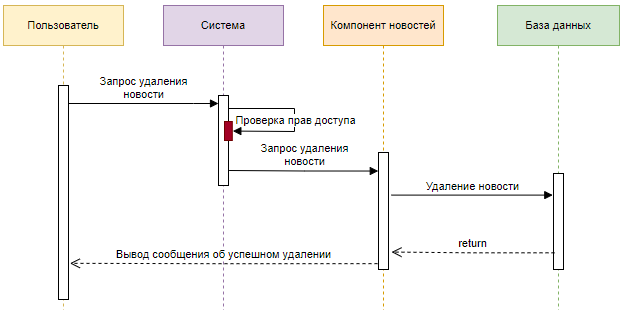


Рисунок 15 Удаление новости

Создание комментария представлено на Рисунок 16.

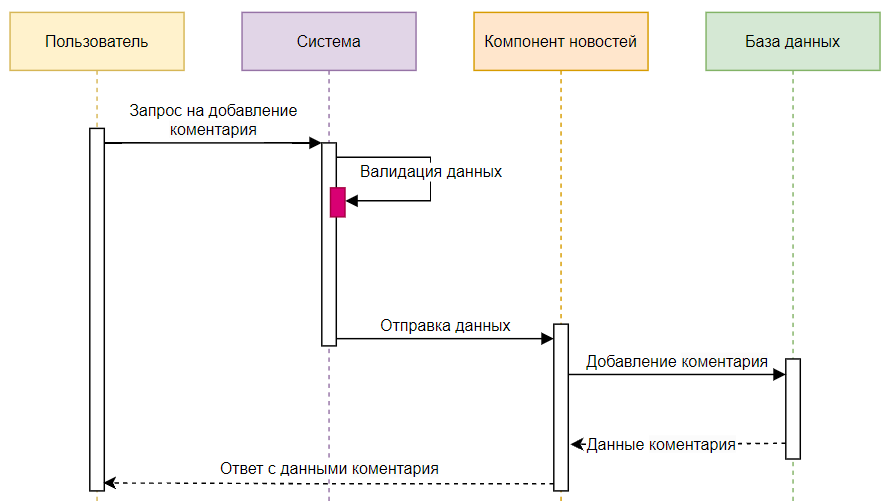


Рисунок 16 Создание комментария

### Работа с контактами

Просмотр контактов представлен на Рисунок 17.

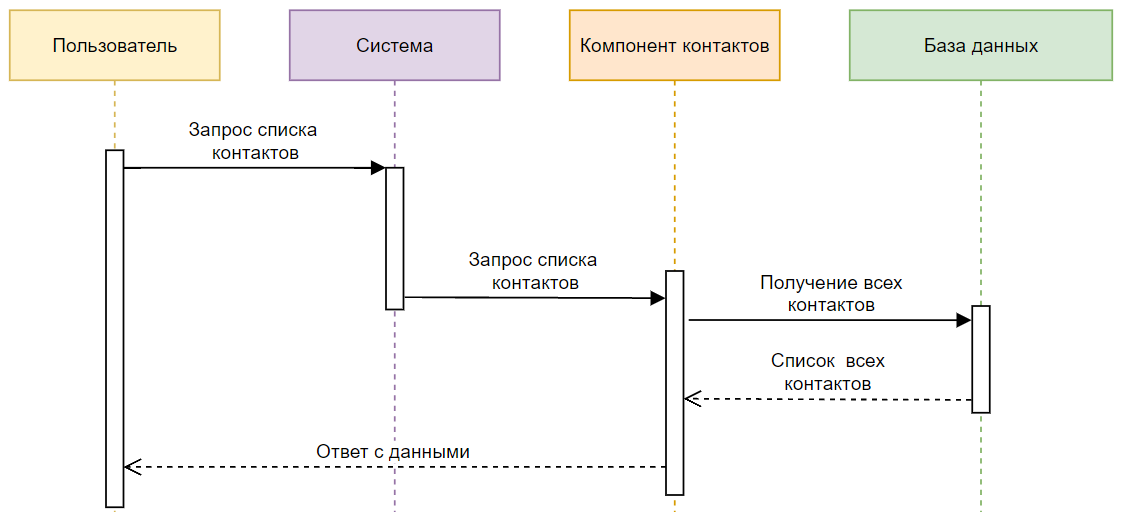


Рисунок 17 Просмотр контактов

Создание контактов представлено на Рисунок 18.

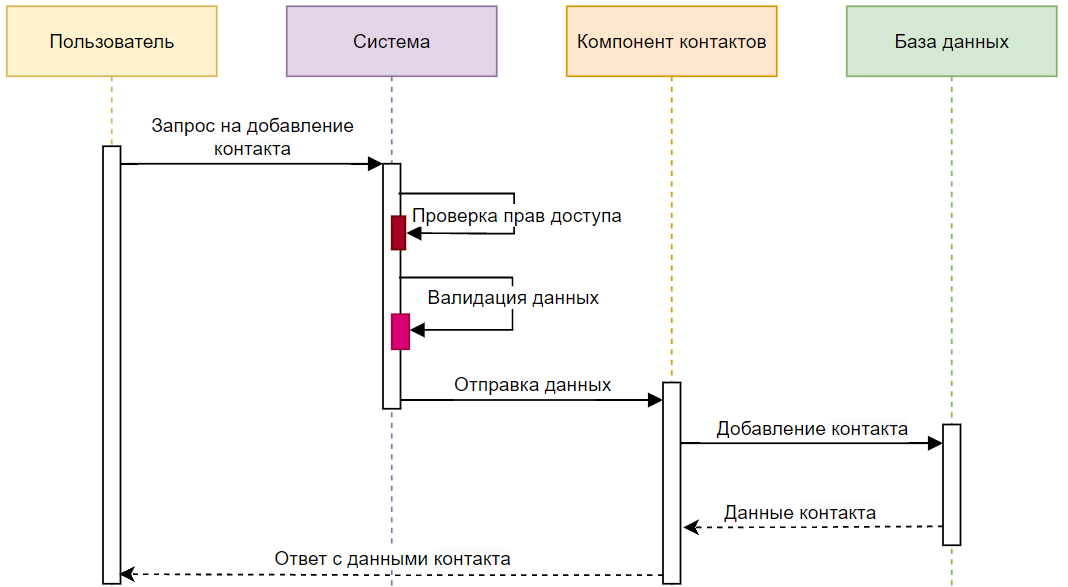


Рисунок 18 Создание контактов

Редактирование контактов представлено на Рисунок 19.

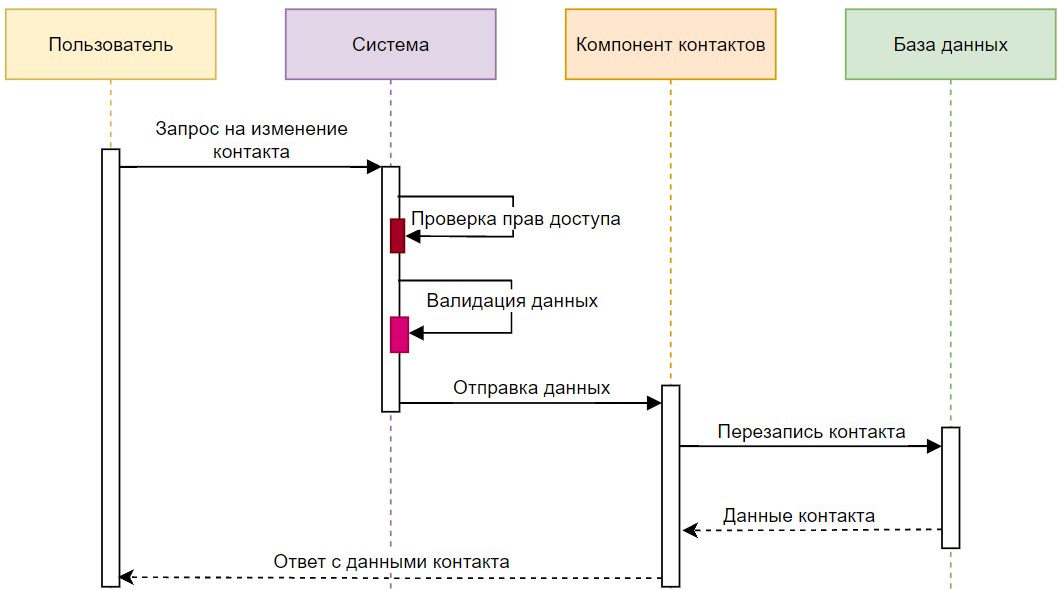


Рисунок 19 Редактирование контактов

Удаление контактов представлено на Рисунок 20.

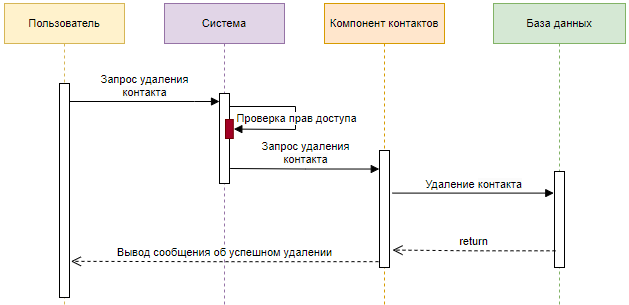


Рисунок 20 Удаление контактов

### Работа с данными для входа

Просмотр данных для входа представлен на Рисунок 21.

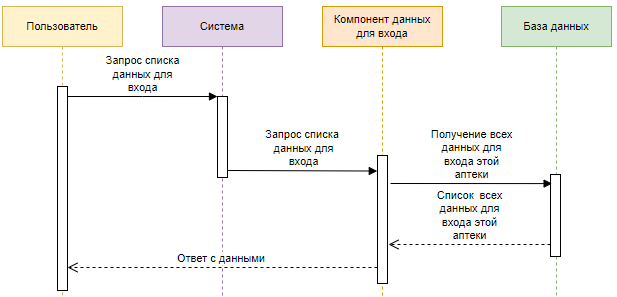


Рисунок 21 Просмотр данных для входа

Создание данных для входа представлена на Рисунок 22.

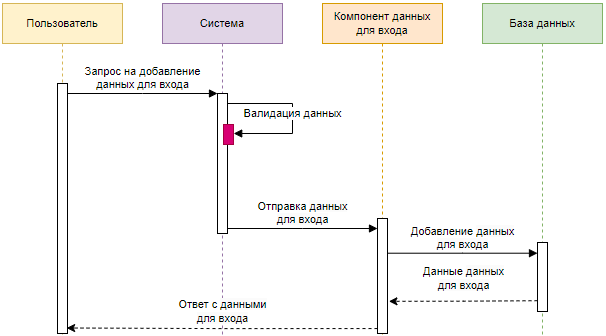


Рисунок 22 Создание данных для входа

Редактирование данных для входа представлена на Рисунок 23.

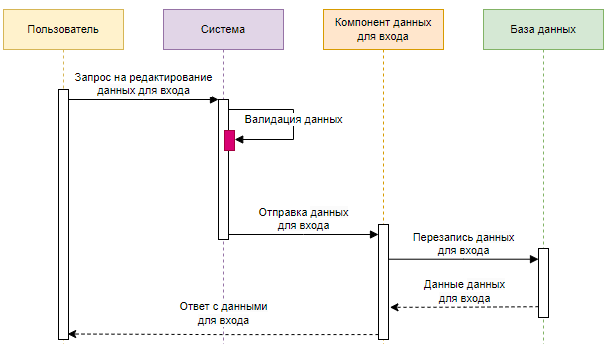


Рисунок 23 Редактирование данных для входа

Удаление данных для входа представлена на Рисунок 24.

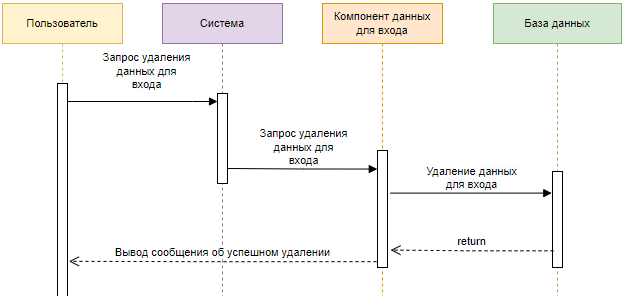


Рисунок 24 Удаление данных для входа

### Работа с полезными ссылками

Просмотр полезных ссылок представлен на Рисунок 25.

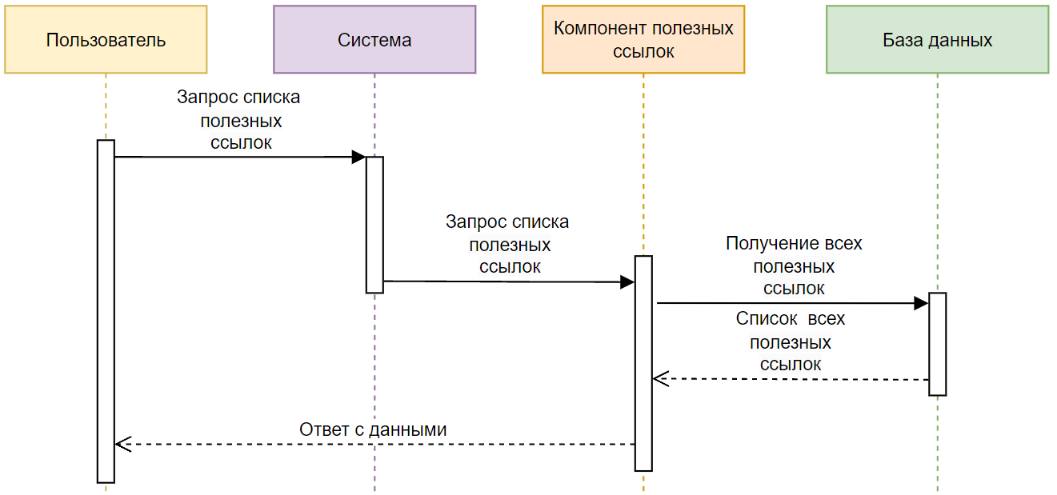


Рисунок 25 Просмотр полезных ссылок

Создание полезных ссылок представлена на Рисунок 26.

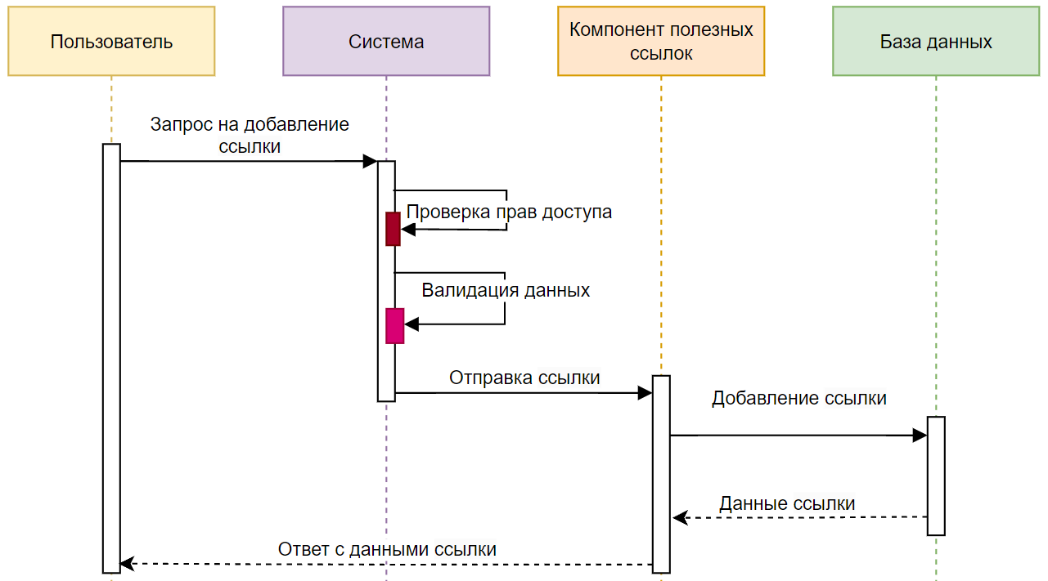


Рисунок 26 Создание полезных ссылок

Редактирование полезных ссылок представлена на Рисунок 27.

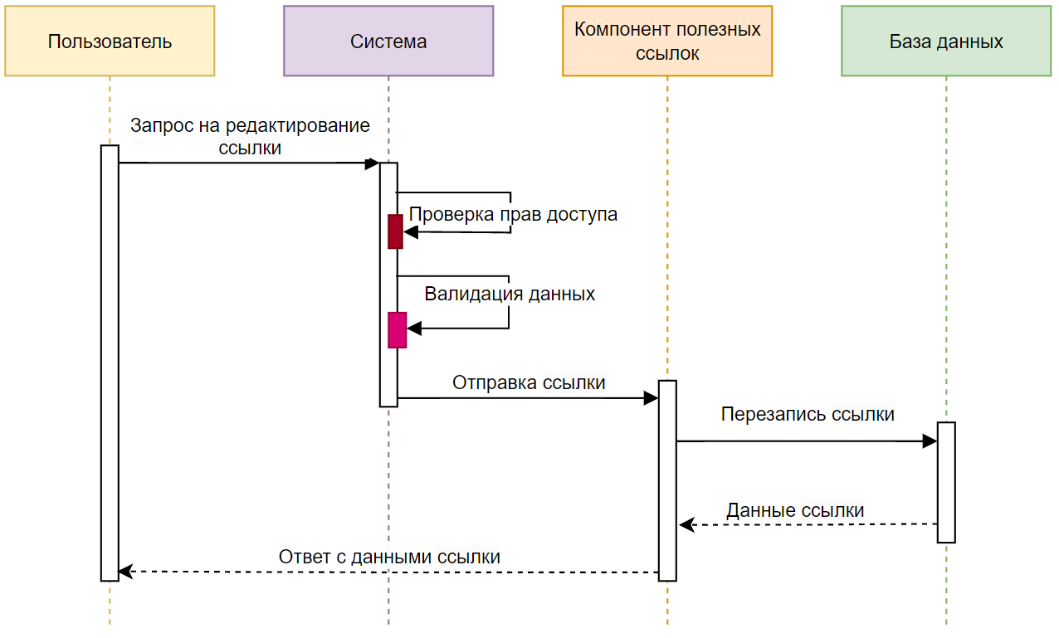


Рисунок 27 Редактирование полезных ссылок

Удаление полезных ссылок представлена на Рисунок 28.

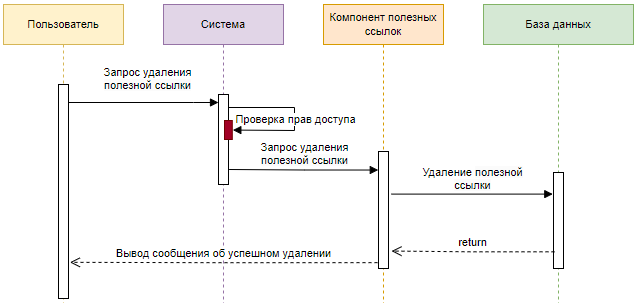


Рисунок 28 Удаление полезных ссылок

### Работа с брендами

Просмотр брендов представлен на Рисунок 29.

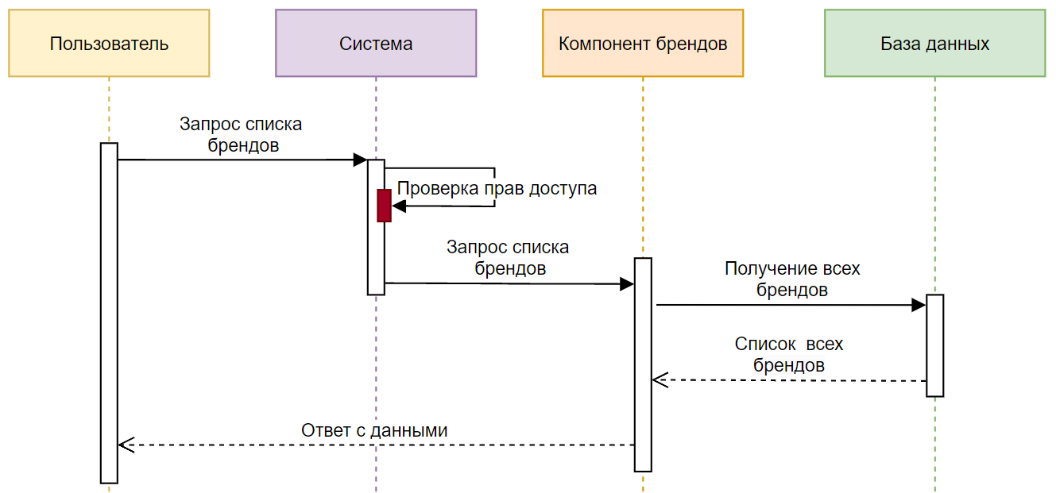


Рисунок 29 Просмотр брендов

Создание брендов представлено на Рисунок 30.

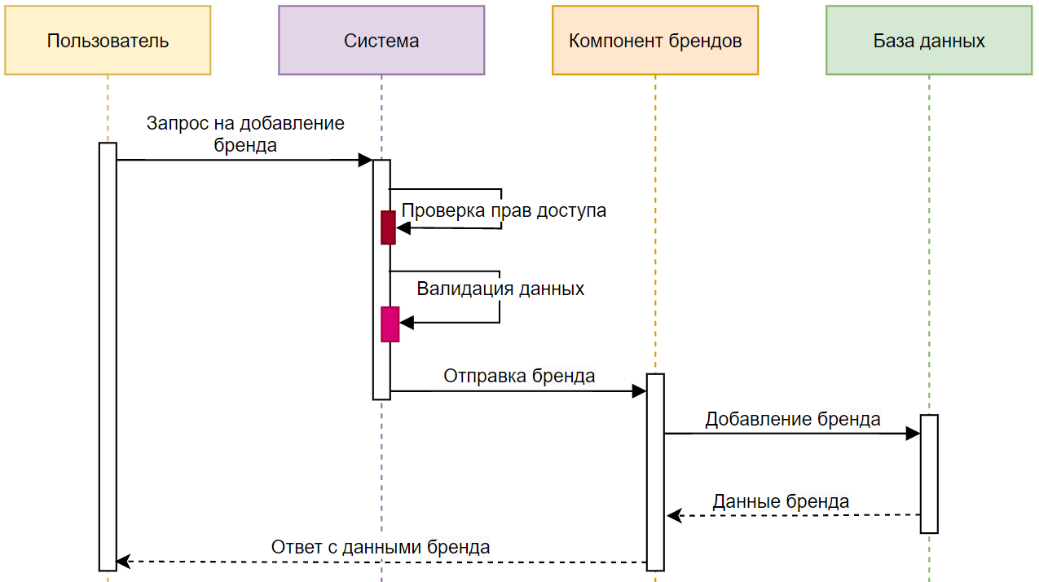


Рисунок 30 Создание брендов

Редактирование брендов представлено на Рисунок 31.

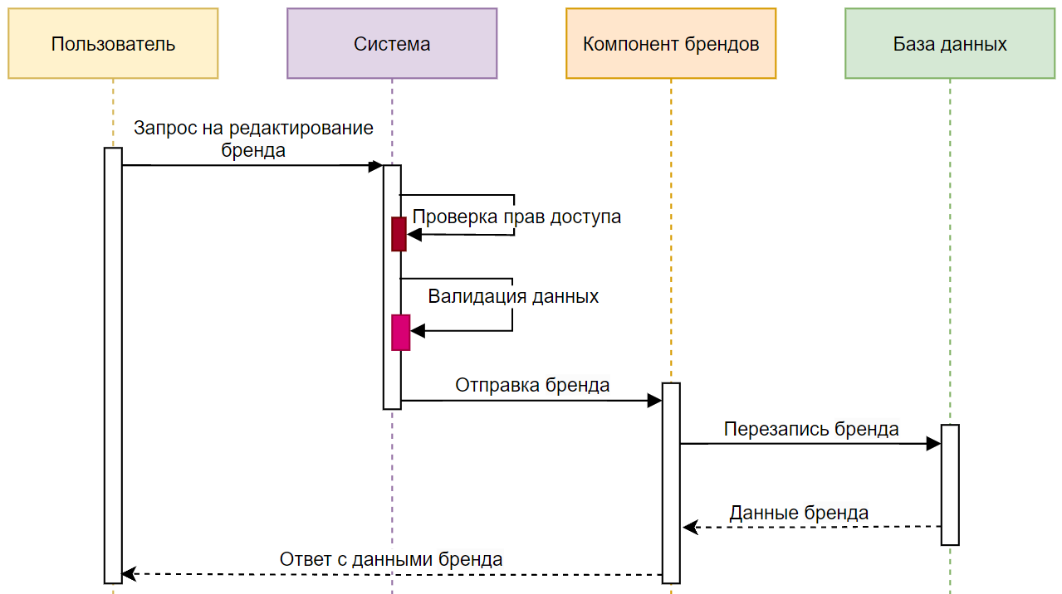


Рисунок 31 Редактирование брендов

Удаление брендов представлено на Рисунок 32.

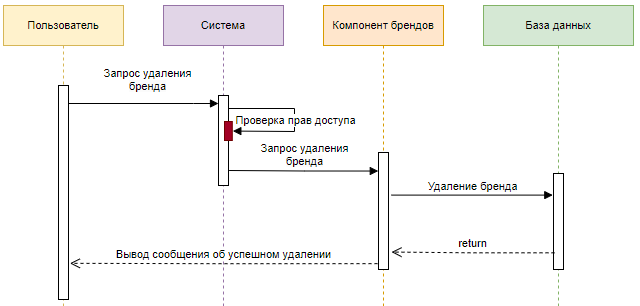


Рисунок 32 Удаление брендов

### Работа с юридическими лицами

Просмотр юр. лиц представлен на Рисунок 33.

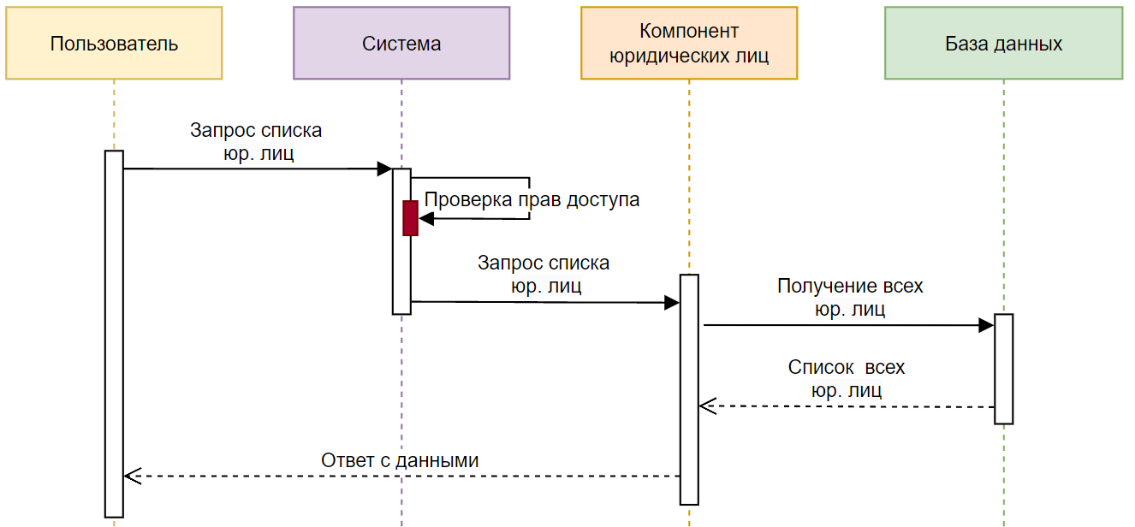


Рисунок 33 Просмотр юр. лиц

Создание юр. лиц представлено на Рисунок 34.



Рисунок 34 Создание юр. лиц

Редактирование юр. лиц представлено на Рисунок 35.

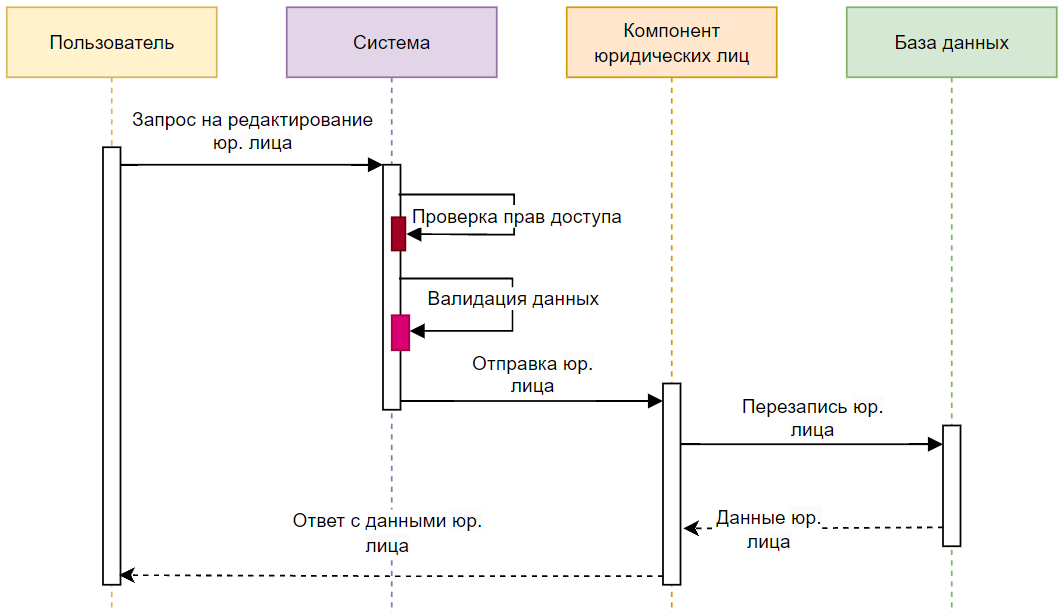


Рисунок 35 Редактирование юр. лиц

Удаление юр. лиц представлено на Рисунок 36.

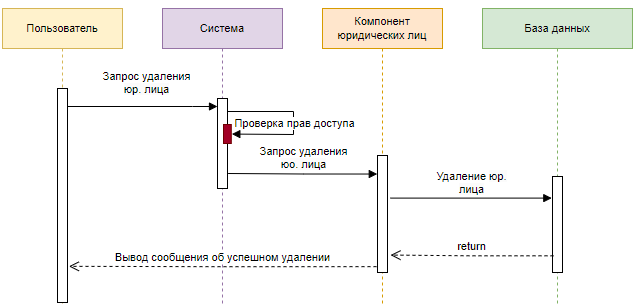


Рисунок 36 Удаление юр. лиц

### Работа с прайс-листами

Просмотр прайс-листов представлен на Рисунок 37.

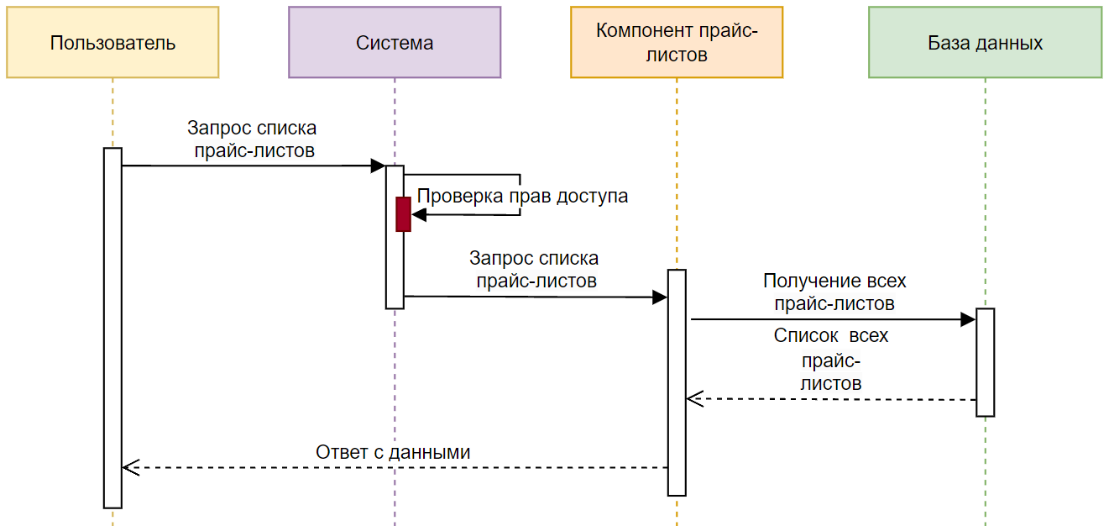


Рисунок 37 Просмотр прайс-листов

Создание прайс-листов представлено на Рисунок 38.



Рисунок 38 Создание прайс-листов

Редактирование прайс-листов представлено на Рисунок 39.

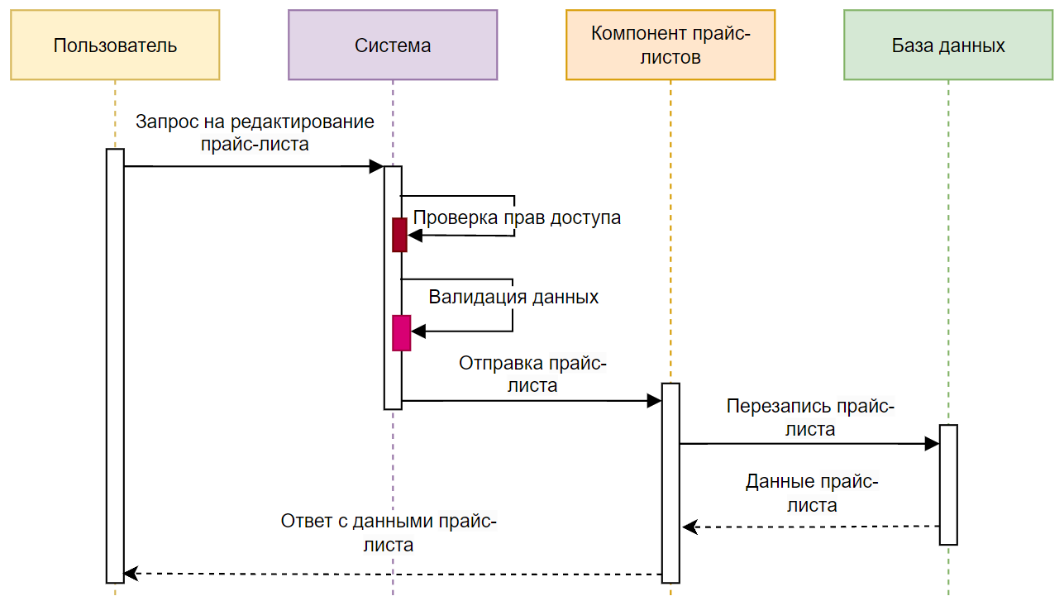


Рисунок 39 Редактирование прайс-листов

Удаление прайс-листов представлено на Рисунок 40.

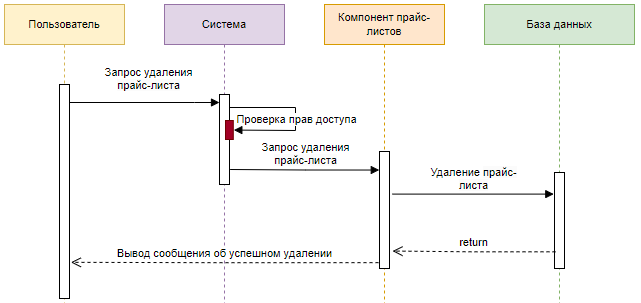


Рисунок 40 Удаление прайс-листов

## Макеты страниц

На основе анализа словаря данных, диаграммы прецендентов и диаграмм последовательностей разработать макеты страниц, отражающие сценарии взаимодействия пользователя с системой.

При выборе действия, в котором нужна авторизация неавторизированного пользователя, выводить страницу с авторизацией (см. Рисунок 41).

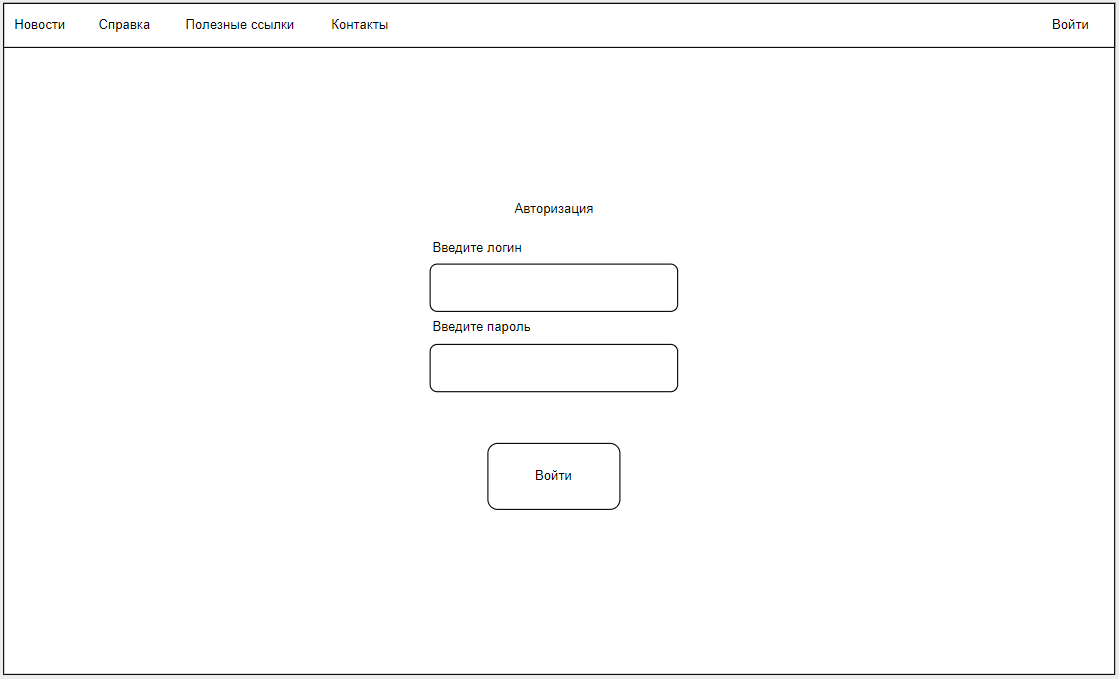


Рисунок 41 Макет авторизации

При открытии сайта открывается страница “Справка” для поиска препаратов (см. Рисунок 42). Препарат можно искать по аптекам, поставщикам и складам, а также указывать город и аптечный бренд, снизу страницы расположены последние новости.

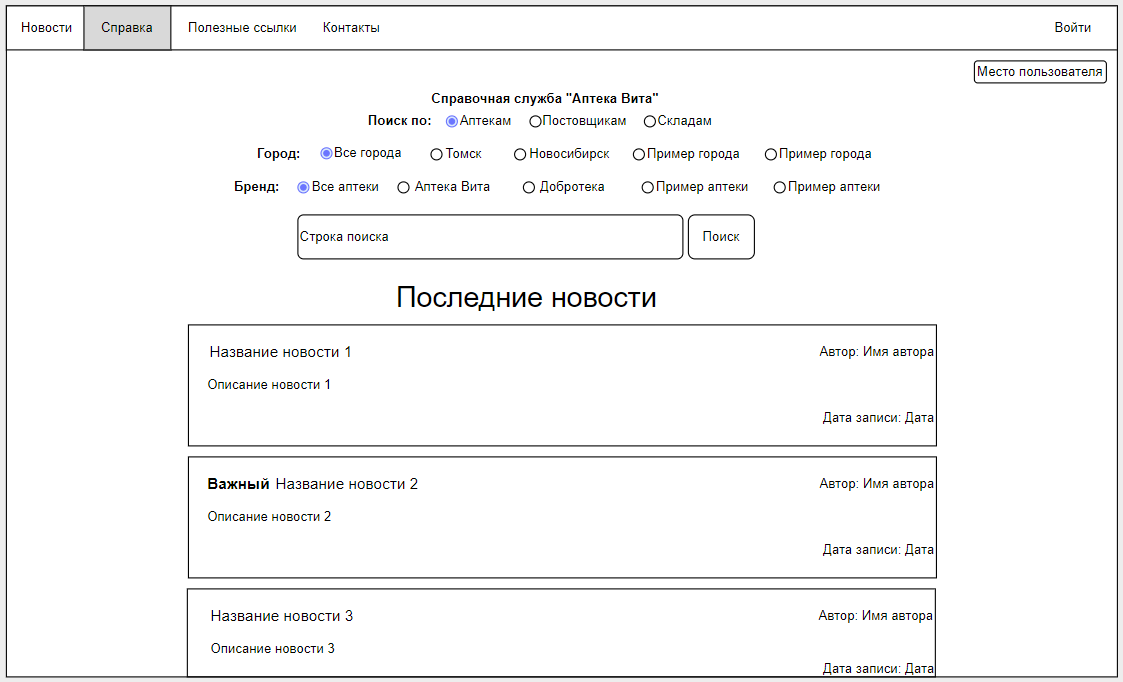


Рисунок 42 Макет “Справка”

При поиске препарата блок с новостями заменяется на таблицу с препаратами и данными по ним (см. Рисунок 43).

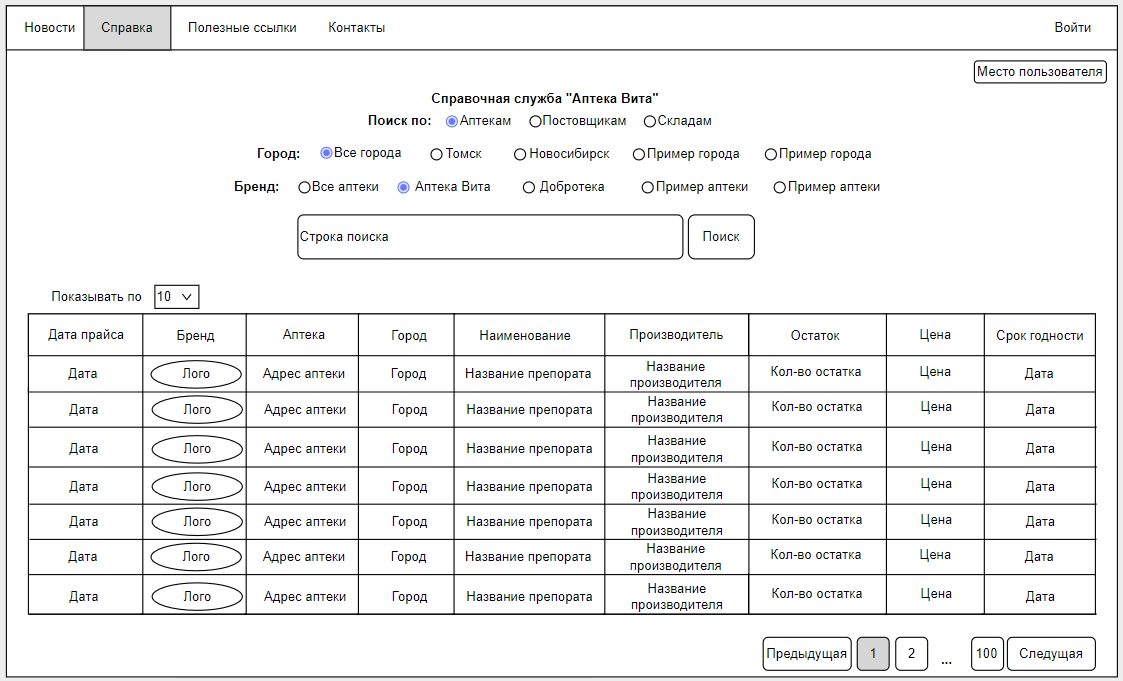


Рисунок 43 Макет “Справка” поиск препарата

Если в навигационном меню выбрать новости откроется страница “Новости” со всеми новостями и информацией по ним. Каждая новость имеет параметр важности. В зависимости от него новость может содержать специальную приписку, подчеркивающую её значимость. Обычные новости таких приписок не имеют (см. Рисунок 44).

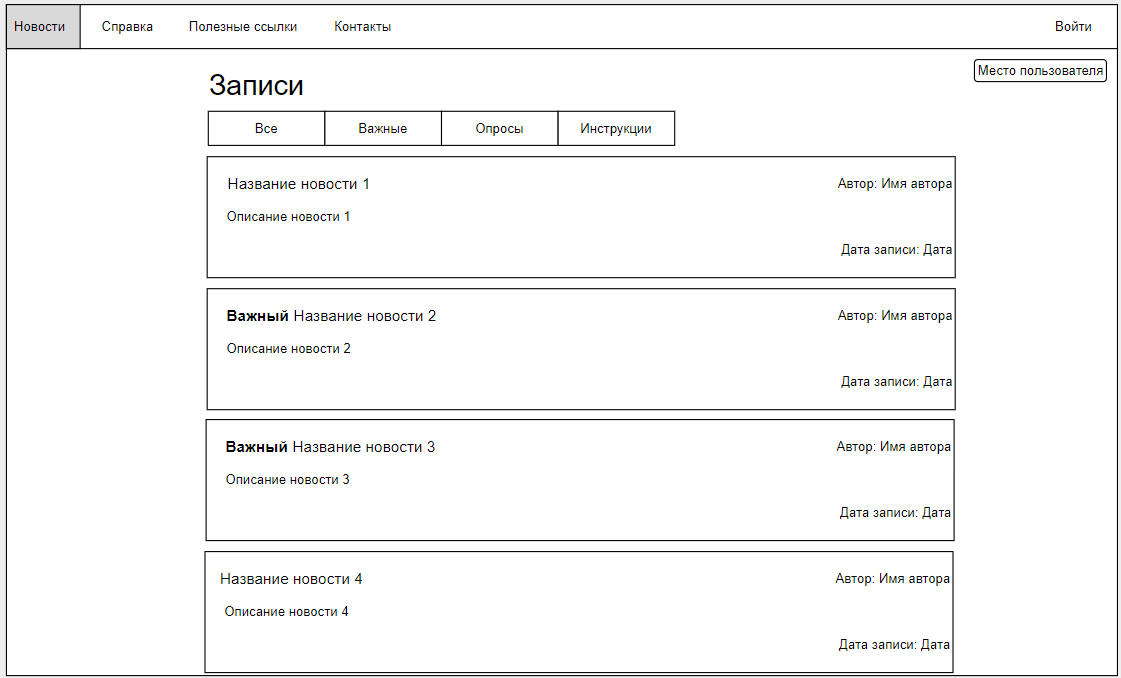


Рисунок 44 Макет “Новости”

При выборе нужной новости открывается страница с полным текстом новости, к которой можно написать комментарий с прикреплённым файлом, автором комментария является аптека, из которой написан он (см. Рисунок 45).

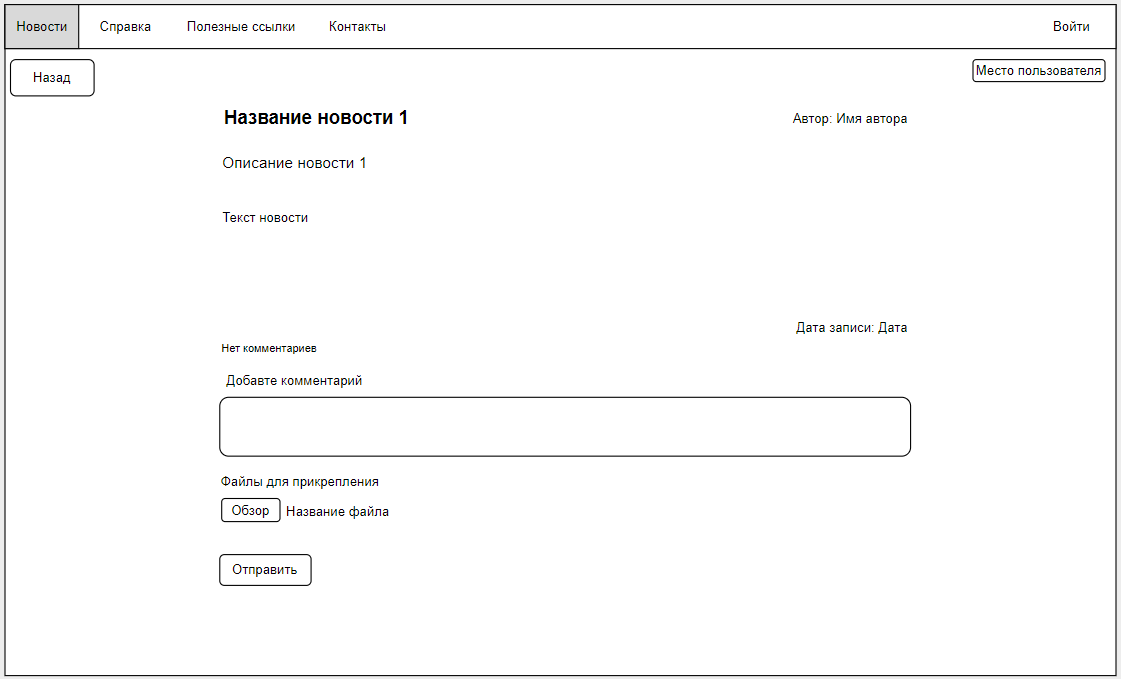


Рисунок 45 Макет открытой новости

Для создания новости пользователь должен нажать кнопку “Войти” после чего открывается страница с авторизацией (см. Рисунок 41) после авторизации нажать на кнопку “добавить новость” (см. Рисунок 46) после чего откроется страница с созданием новости (см. Рисунок 47).

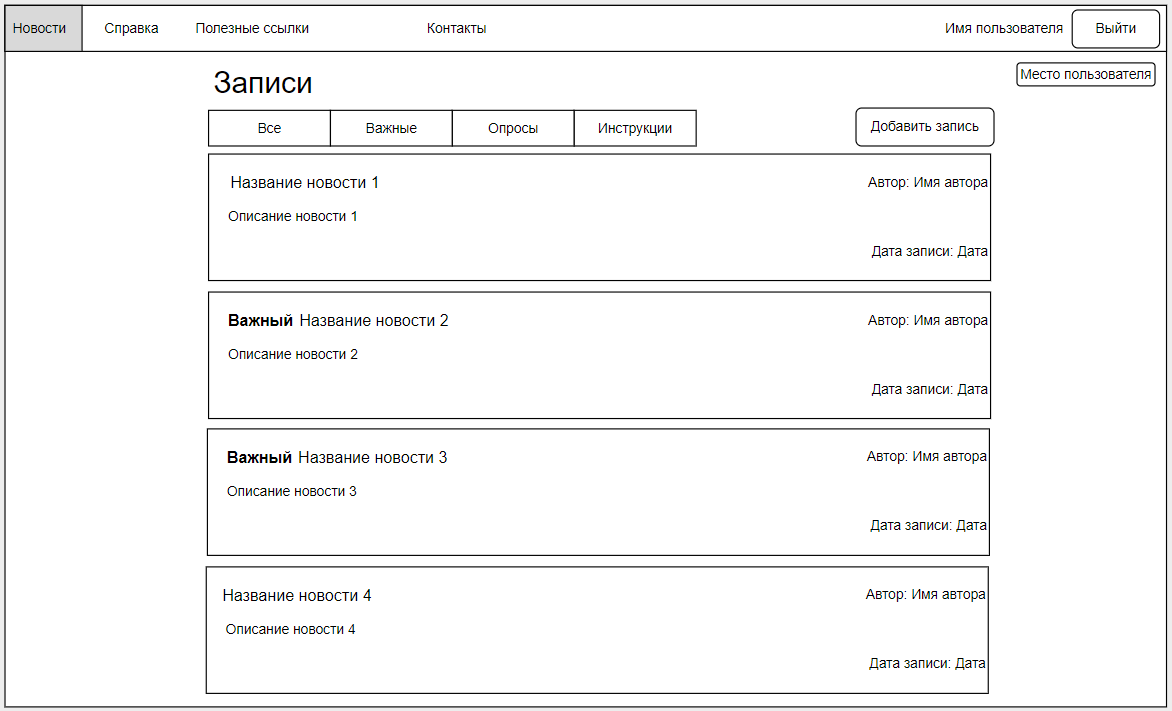


Рисунок 46 Макет “Новости” после авторизации

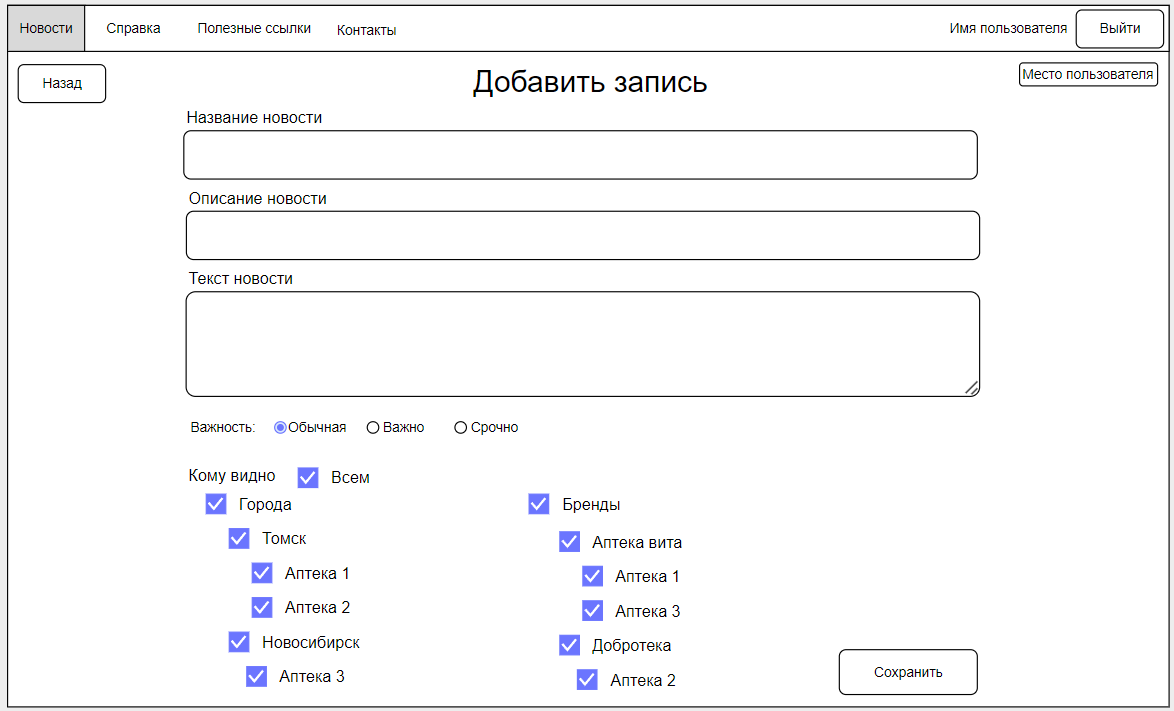


Рисунок 47 Макет создания новости

Если в навигационном меню выбрать пункт “Полезные ссылки” откроется страница “Полезные ссылки” со всеми ссылками, которые могут пригодится при работе (см. Рисунок 48). При нажатии на кнопку “Данные для входа” Открывается страница с данными для входа для этой страницы (см. Рисунок 49).

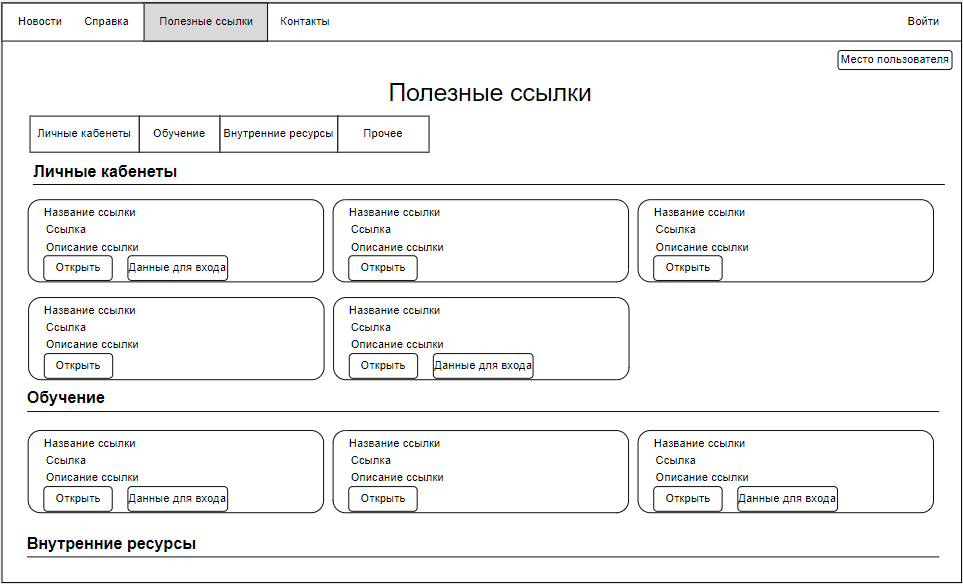


Рисунок 48 Макет “Полезные ссылки”

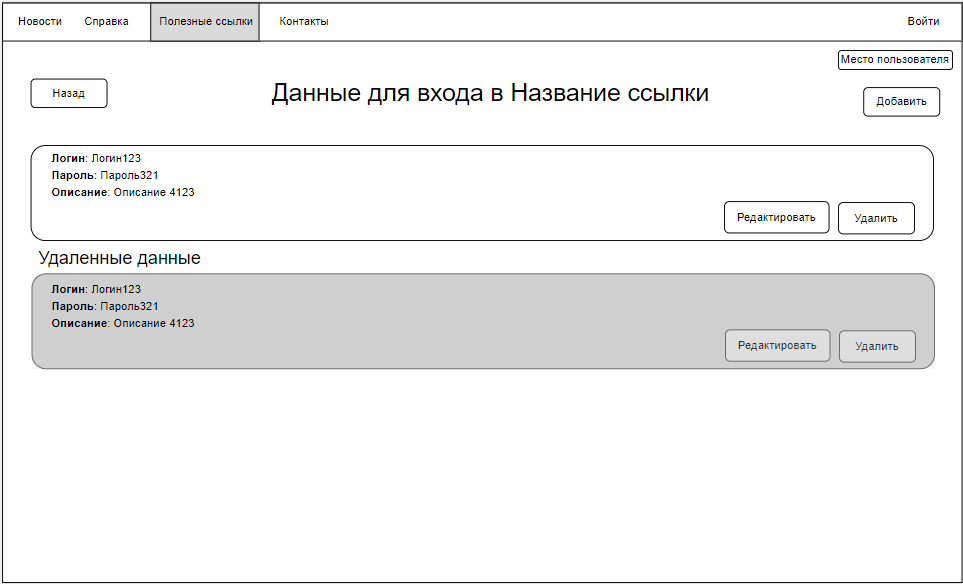


Рисунок 49 Макет “Данные для входа”

Если в навигационном меню выбрать пункт “Контакты” откроется страница “Контакты” с контактами которые могут пригодится в ходе работы (см. Рисунок 50).

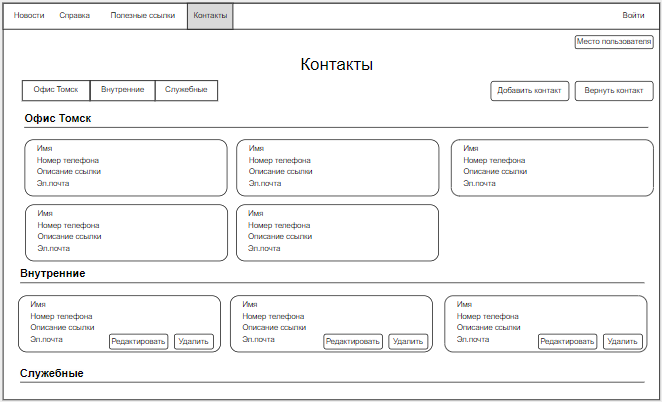


Рисунок 50 Макет “Контакты”

Так же для администратора сделать страницы для управления городами, брендами, юридическими лицами, полезными ссылками, контактами по примеру макета “Управление ссылками” (см. Рисунок 51) и макета добавления ссылки (см. Рисунок 52).

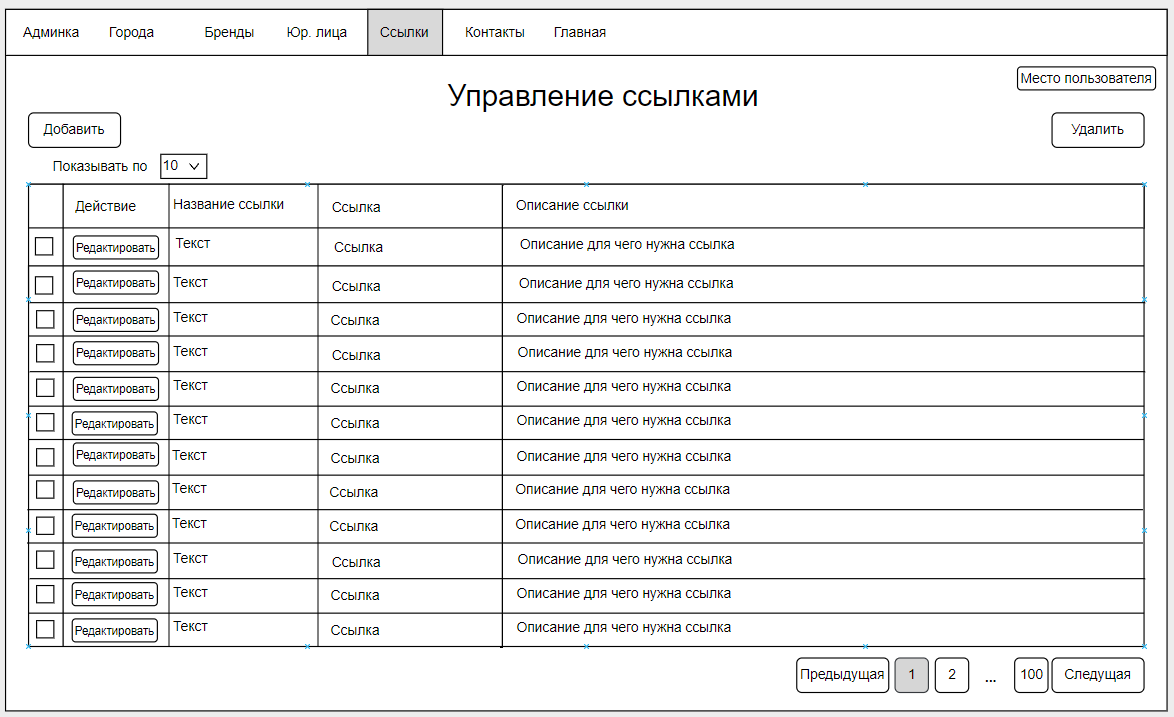


Рисунок 51 Макет “Управление ссылками”

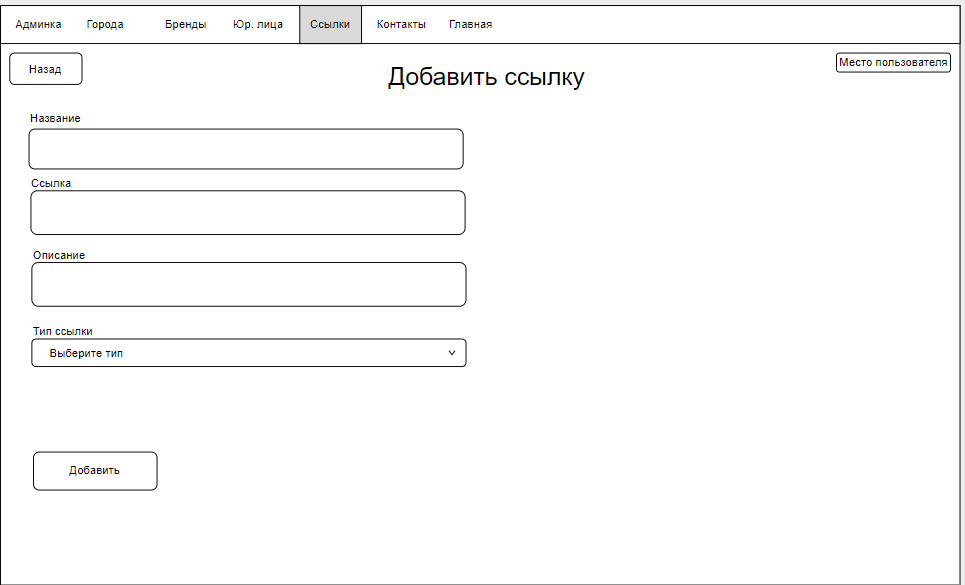


Рисунок 52 Макет добавление ссылки

# РЕАЛИЗАЦИЯ ВЕБ-САЙТА

Веб-сайт внутреннего портала аптечной сети разрабатывается с использованием HTML, CSS, JavaScript и PHP. Особое внимание уделено удобству пользователей за счёт соблюдения принципов юзабилити, простой структуры и компонентного подхода.

Интерфейс сайта разрабатывается адаптивным и интуитивно понятным, с учетом различных ролей сотрудников и их задач. В процессе разработки предусматривается система навигации для удобного перехода между разделами, а также механизмы защиты от случайных ошибок при вводе и обработке данных.

Сайт разделен на две части пользовательская и административная.

1. Пользовательская часть
   * Взаимодействие с новостями:
     + Просмотр новостей.
     + Создание комментариев.
     + Создание ответов на опросы.
   * Поиск препаратов.
   * Доступ к полезным ссылкам.
   * Просмотр контактов.
2. Административная часть
   * Управление данными через CRUD-операции (создание, чтение, обновление, удаление):
     + Прайс-листы.
     + Города.
     + Бренды.
     + Юридические лица.
     + Ссылки.
     + Контакты и их категории.
   * Создание новостей и опросов к ним.

## Пользовательская часть

При открытии сайта по умолчанию загружается страница “Справка” (см. Рисунок 53). В верхней части страницы расположен блок поиска препаратов с возможностью фильтрации по городам, районам и брендам аптек. Ниже отображаются последние новости.  
После выполнения поиска блок с новостями автоматически заменяется на таблицу с результатами поиска препаратов (см. Рисунок 54).

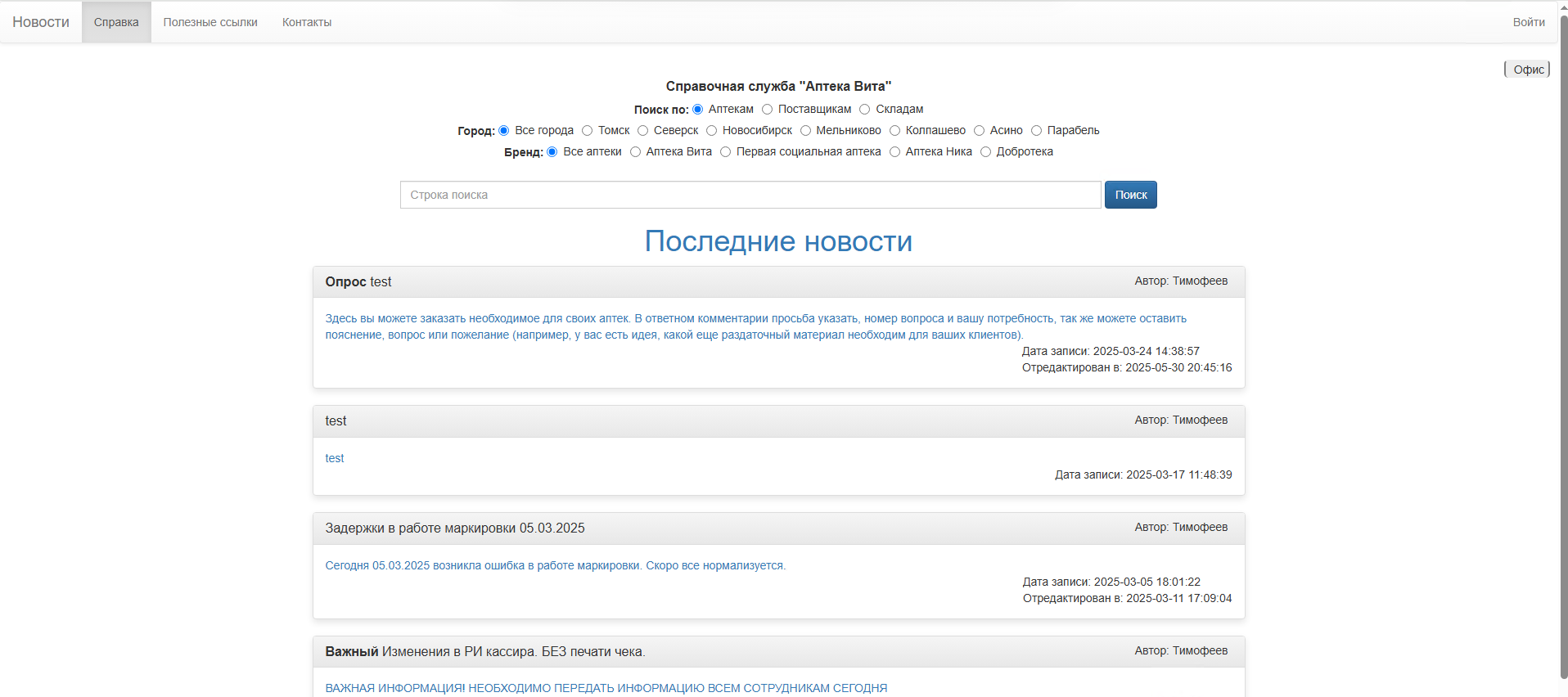


Рисунок 53 Страница “Справка”

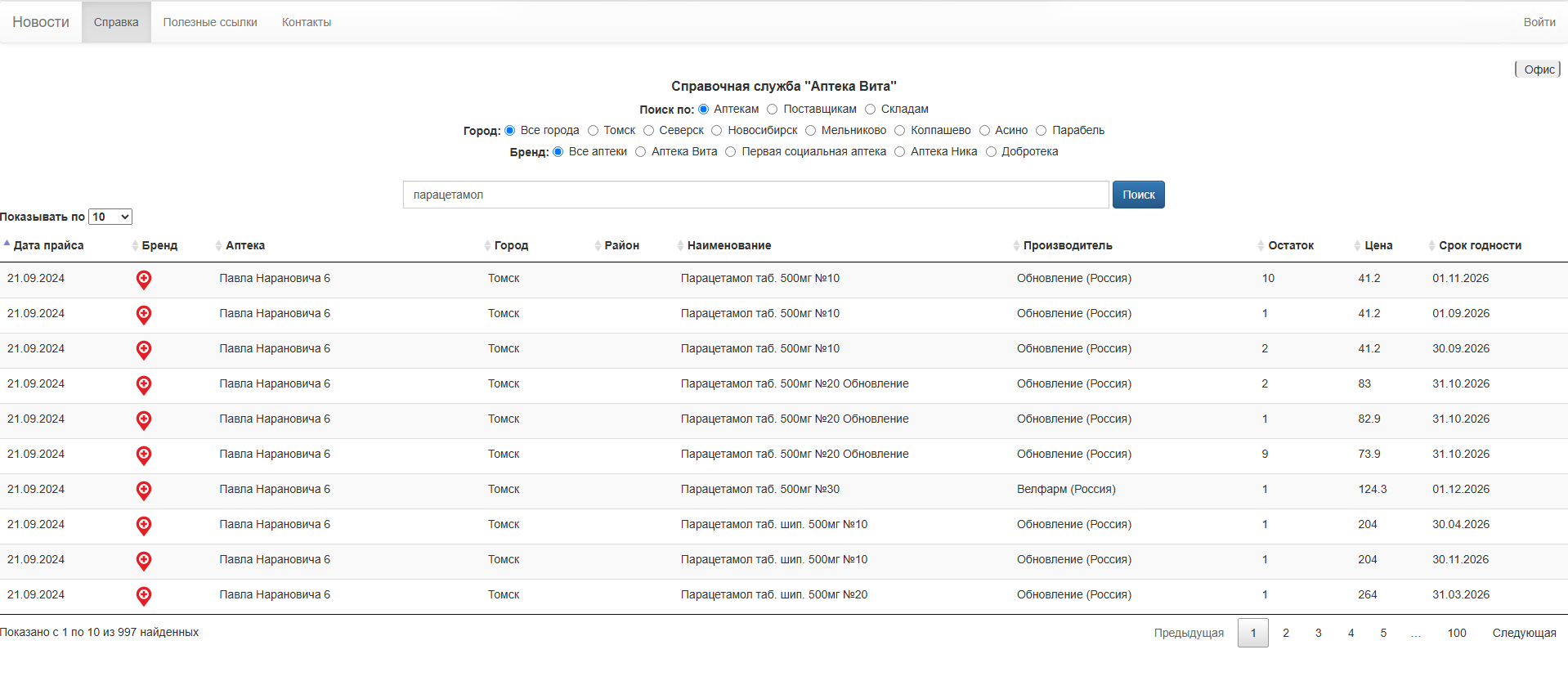


Рисунок 54 Страница “Справка” поиск препаратов

Листинг 1 представляет собой HTML-разметку тела страницы Справка, в которой размещены элементы фильтрации, формы поиска и блоки, динамически заменяемые в зависимости от наличия результатов.

Листинг 1 – Body страницы “Справка”

<body>

<table width=100% border=0>

<tr>

<td width=100% >

<?php

$navbarType = "user";

include 'topnavbar.php';

?>

<form class="form-inline" method="POST">

<div class="searcharea2" id="searcharea2" style="width: 100%; align-content: center; margin: 0 auto;" >

<div style="text-align: center;" >

<label class="headerfont">Справочная служба "Аптека Вита"</label>

<div style="text-align: center;">

<div1 style='text-align: center;'>

<label>Поиск по: </label>

</div1>

<?php

$i = 0;

foreach ($searching\_type as $k => $v) {

echo "<div class='radio-inline' style='text-align: center;'><input type='radio' name='searching\_type' id='$v' value='$k'";

if (isset($\_POST['searching\_type']) && ($\_POST['searching\_type'] == $k)) {

echo 'checked';

}

if (!isset($\_POST['searching\_type']) && $i == 0) {

echo 'checked';

}

echo "><label style='cursor: pointer; font-weight: normal;' for='$v'>$v</label></div>";

$i++;

}

?>

</div>

</div>

<div style='text-align: center;'>

<div1 style='text-align: center;'>

<label>Город: </label>

</div1>

<div1 style='text-align: center; '>

<?php

$visibleCities = 0;

$usercity = 0;

// Если пользователь не в офисе, по умолчанию выбираем первый город в списке

if (isset($cityuser[1])) {

$cities\_check = $cityuser[1];

$usercity = $cityuser[1];

$q = "SELECT `visiblecities` FROM " . CITIESTABLE . " WHERE `idcity` = $usercity";

$quer = mysqli\_query($link, $q);

$res = mysqli\_fetch\_row($quer);

$visibleCities = explode(',', $res[0]);

$visibleCities[] = $cities\_check;

// Проверяем, была ли отправлена форма

if (isset($\_POST['searching\_city'])) {

$cities\_check = $\_POST['searching\_city'];

}

} else if (isset($\_POST['searching\_city'])) {

$cities\_check = $\_POST['searching\_city'];

}

// Если пользователь в офисе, отображаем все города

if ($usercity == 0) {

foreach ($cities as $k => $v) {

echo "<div style='text-align: center;' class='radio-inline'>";

echo "<input type='radio' name='searching\_city' id='$k' value='$k'";

if ($cities\_check == $k) {

echo 'checked';

}

echo "><label style='cursor: pointer; font-weight: normal;' for='$k'>";

echo $v['namecity'] . "</label></div>";

}

}

// Если пользователь не в офисе, отображаем только города из $visibleCities

else {

foreach ($cities as $k => $v) {

if ($k == 0 || in\_array($k, $visibleCities)) {

echo "<div style='text-align: center;' class='radio-inline'>";

echo "<input type='radio' name='searching\_city' id='$k' value='$k'";

if ($cities\_check == $k) {

echo 'checked';

}

echo "><label style='cursor: pointer; font-weight: normal;' for='$k'>";

echo $v['namecity'] . "</label></div>";

}

}

}

?>

</div1>

</div>

<div style='text-align: center;' id="searching\_district">

<?php

if (isset($\_POST['searching\_district'])) {

$district\_check = $\_POST['searching\_district'];

}

?>

</div>

<div style='text-align: center;'>

<div1 style='text-align: center; '>

<label>Бренд: </label>

</div1>

<div1 style='text-align: center; '>

<?php

$j = 0;

foreach ($seller as $k => $v) {

$n = $v['name'];

echo "<div style='text-align: center;' class='radio-inline'>";

echo "<input type='radio' name='seller' id='$n' value='$k'";

if (isset($\_POST['seller']) && ($\_POST['seller'] == $k)) {

echo 'checked';

}

if (!isset($\_POST['seller']) && $j == 0) {

echo 'checked';

}

echo "><label style='cursor: pointer; font-weight: normal;' for='$n'>";

echo $n . "</label></div>";

$j++;

}

?>

</div1>

<div style=" text-align: center">

<div1 class="input-group" style="width: 45%; text-align: center">

<input style="margin-top: 20px; " type="text" class="form-control" name="search"

placeholder="Строка поиска"

value="<?php if (isset($\_POST['search'])) { echo($\_POST['search']); } else { echo ""; } ?>">

</div1>

<div1>

<button style="margin-top: 20px;" type="submit" name="submit" id="submit" class="btn btn-primary">Поиск</button>

</div1>

</div>

</div>

</div>

</form>

</td>

</tr>

</table>

<?php  
  
if (isset($\_POST["submit"])) {  
 $key1 = (isset($\_POST["search"])) ? $\_POST["search"] : '';  
 $key2 = explode(" ", $key1);  
 $key = implode("%' AND `name` LIKE '%", $key2);  
 $key = "%$key%";  
 $stock = array\_search($seller[$\_POST["seller"]], $seller, true);   
if(mb\_strlen($key1) >= 3) {  
 if(isset($\_POST["searching\_district"])){  
 search($key, $\_POST["searching\_type"], $stock, $\_POST["searching\_city"], $\_POST["searching\_district"]);  
 }else{  
 search($key, $\_POST["searching\_type"], $stock, $\_POST["searching\_city"], 0);  
 }  
} else {  
 echo "<div style='border: medium'>";  
 echo "<strong>Минимальная строка запроса - 3 символа</strong>";  
 echo '</div>';  
}  
}  
// SQL-запрос для получения записей, сортированных по важности и дате  
$sql\_posts = "SELECT \*   
FROM posts where is\_public = 1  
ORDER BY   
 CASE   
 WHEN dateimportance >= CURRENT\_DATE THEN 0   
 ELSE 1   
 END,  
 CASE   
 WHEN dateimportance >= CURRENT\_DATE THEN dateimportance   
 END ASC,  
 CASE   
 WHEN dateimportance < CURRENT\_DATE THEN date\_created   
 END DESC";  
$result\_posts = $link->query($sql\_posts); // Выполнение запроса  
// SQL-запрос для получения последних новостей  
$sql\_latest\_posts = "SELECT \* FROM posts ORDER BY date\_created DESC";  
$result\_latest\_posts = $link->query($sql\_latest\_posts); // Выполнение запроса  
// Проверка, является ли пользователь администратором  
$isadmin = 0;  
if(isset($\_SESSION['login'])){  
 if($\_SESSION['login'] == 'admin'){  
 $isadmin = 1;  
 }  
}  
?>  
<div class="container" style='display:block' id='newsdiv'>  
<h1 class="text-center"><a href="news.php" style="outline: none;"> Последние новости</a></h1>  
<?php  
$i = 0;  
$current\_user\_id = finduser(3); // Получение текущего ID пользователя  
// Сначала собираем все опросы  
$survey\_map = []; // post\_id => survey\_id  
$survey\_query = "SELECT id, post\_id FROM surveys";  
$survey\_result = $link->query($survey\_query);  
if ($survey\_result && $survey\_result->num\_rows > 0) {  
 while ($row = $survey\_result->fetch\_assoc()) {  
 $survey\_map[$row['post\_id']] = $row['id'];  
 }  
}  
// Загружаем все ответы пользователя  
$user\_answers = []; // survey\_id => true  
$user\_answer\_query = "SELECT survey\_id FROM user\_answers WHERE user = $current\_user\_id";  
$user\_answer\_result = $link->query($user\_answer\_query);  
if ($user\_answer\_result && $user\_answer\_result->num\_rows > 0) {  
 while ($row = $user\_answer\_result->fetch\_assoc()) {  
 $user\_answers[$row['survey\_id']] = true;  
 }  
}  
if ($result\_posts && $result\_posts->num\_rows > 0) {  
 while ($post = $result\_posts->fetch\_assoc()) {  
 $isanswered = false; // Флаг ответа  
 $isviewed = false; // Флаг просмотра  
 $enable = false; // Флаг видимости поста  
  
 $selected\_pharmacies = explode(',', $post['visibility']); // Список видимых аптек  
  
 foreach ($selected\_pharmacies as $pharmacy) {  
 if ($current\_user\_id == $pharmacy || $current\_user\_id == 100000 || $current\_user\_id == 200000 || $pharmacy == 0) {  
 $enable = true;  
 break;  
 }  
 } if ((($enable && $post['is\_public'] == 1) || $isadmin == 1) && $i < 5) {  
 $i++;  
 $post\_id = $post['id'];   
 // Получение комментариев  
 $sql\_comments = "SELECT \* FROM comments WHERE post\_id = $post\_id AND is\_public = 1";  
 $result\_comments = $link->query($sql\_comments);   
 // Получение логов (просмотров)  
 $sql\_logs = "SELECT \* FROM logs WHERE postid = $post\_id";  
 $result\_logs = $link->query($sql\_logs);   
 $has\_comment = false;  
 $has\_answer = false;   
 // Проверка комментариев  
 if ($result\_comments && $result\_comments->num\_rows > 0) {  
 while ($comment = $result\_comments->fetch\_assoc()) {  
 $user\_id\_to\_check = $comment['user\_id'];  
 if (($user\_id\_to\_check == $current\_user\_id && !isset($\_SESSION['user\_id'])) || (isset($\_SESSION['user\_id'])) || ($current\_user\_id == 100000)) {  
 $has\_comment = true;  
 break;  
 }  
 }  
 }  
 // Проверка ответа на опрос  
 if ($post['post\_type'] == '2') {  
 if (isset($survey\_map[$post\_id])) {  
 $survey\_id = $survey\_map[$post\_id];  
 if (isset($user\_answers[$survey\_id])) {  
 $has\_answer = true;  
 }  
 }  
 }  
 // Итоговая установка флага ответа  
 if ($has\_comment || $has\_answer || (isset($\_SESSION['user\_id'])) || ($current\_user\_id == 100000)) {  
 $isanswered = true;  
 }  
 // Проверка просмотров  
 if ($result\_logs && $result\_logs->num\_rows > 0) {  
 while ($log = $result\_logs->fetch\_assoc()) {  
 if ($log['user'] == findip()) {  
 $isviewed = true;  
 break;  
 }  
 }  
 } elseif ((isset($\_SESSION['user\_id'])) || ($current\_user\_id == 100000)) {  
 $isviewed = true;  
 }  
?>   
 <div class="panel panel-default">  
 <div class="panel-heading <?php if (($post['post\_type'] == '2' && !$isanswered) || (($post['importance'] == 'important'|| $post['dateimportance'] >= date("Y-m-d")) && !$isviewed)) echo ' blinking'; ?>">  
 <?php if (isset($\_SESSION["user\_id"]) && $post['is\_public'] == 1) { ?>  
 <div class="btn-group pull-right" style="margin-top:-6px">  
 <a href="deletenews.php?id=<?php echo $post\_id; ?>" class="btn btn-danger btn-sm" onclick="return confirm('Вы уверены, что хотите удалить эту запись?');">Удалить</a>  
 </div>  
 <?php }   
 if($post['is\_public'] == 0 && $isadmin == 1){  
 $postid = $post['id'];  
 echo "<div class='btn-group pull-right' style='margin-top:-6px'>";  
 echo "<a href='retnews.php?id=$postid' class='btn btn-warning btn-sm' onclick='return confirmReturn();'>Вернуть новость</a>";  
 echo "</div>";  
 }  
 ?>  
 <?php   
 // Отображение имени редактора, если запись была отредактирована  
 if ($post['autor'] != $post['lasteditor']) {  
 echo "<div class='btn-group pull-right' style='margin-top:-3px; margin-right: 5px'>Отредактирован:";  
 $sql\_user = "SELECT name FROM users WHERE id = " . $post['lasteditor'];   
 $result\_user = $link->query($sql\_user);  
 if ($result\_user && $result\_user->num\_rows > 0) {  
 $user = $result\_user->fetch\_assoc();  
 echo htmlspecialchars($user['name']);  
 }  
 echo "</div>";  
 }  
 ?>  
 <div class="btn-group pull-right" style="margin-top:-3px; margin-right: 5px">Автор: <?php  
 $sql\_user = "SELECT name FROM users WHERE id = " . $post['autor'];   
 $result\_user = $link->query($sql\_user);  
 if ($result\_user && $result\_user->num\_rows > 0) {  
 $user = $result\_user->fetch\_assoc();  
 echo htmlspecialchars($user['name']);  
 }  
 ?></div>  
 <h3 class="panel-title">  
 <a href='post.php?id=<?php echo $post\_id; ?>'><b><?php if ($post['dateimportance'] >= date("Y-m-d")) echo 'Актуально '; if ($post['post\_type'] == '2') echo 'Опрос '; if ($post['importance'] == 'important') echo 'Важный ';?></b><?php echo htmlspecialchars($post['title']); ?></a>  
 </h3>  
 </div>  
 <div class="panel-body">  
 <div ><a style="word-wrap: break-word; word-break: break-word;" href='post.php?id=<?php echo $post\_id; ?>'><?php echo htmlspecialchars\_decode(stripslashes($post['short\_description'])); ?></a></div>  
 <div class='btn-group pull-right'>  
 <div>Дата записи: <?php echo $post['date\_created']; ?></div>  
 <?php   
 if ($post['date\_created'] != $post['date\_updated']) {  
 echo "<div>Отредактирован в: " . htmlspecialchars($post['date\_updated']) . "</div>";  
 }  
 if ($post['dateimportance'] >= date("Y-m-d")) {  
 echo "<div>Актуально до: " . htmlspecialchars($post['dateimportance']) . "</div>";  
 }  
 ?>  
 </div>  
 </div>  
 </div>  
<?php  
 }  
 }  
} else {  
?>  
 <p>Нет записей</p>  
<?php  
}  
?>

</body>

Листинг 2 содержит JavaScript-код, обеспечивающий обработку результатов поиска и замену блока новостей на таблицу с найденными препаратами.

Листинг 2 – Script станицы “Справка”

<script>

$(document).ready(function() {

// Находим кнопку по её ID

const table1 = document.getElementById('Farma\_wrapper');

const table2 = document.getElementById('Diller\_wrapper');

const newsdiv = document.getElementById('newsdiv');

if(table1 || table2){

newsdiv.style.display = 'none';

}else{

newsdiv.style.display = 'block';

}

var district\_check = '<?php echo $district\_check ?>';

// Выбираем первый город из списка

var firstCity = $('input[name="searching\_city"]:checked').val();

// Если есть выбранный город, загружаем районы для него

if (firstCity) {

$.ajax({

url: 'get\_districts\_index.php', // Путь к файлу, который будет возвращать список районов

type: 'POST',

data: {cityId: firstCity, districtCheck: district\_check}, // Передаем district\_check

success: function(response) {

$('#searching\_district').html(response);

}

});

}

// Обработчик события на изменение выбранного города

$(document).on('change', 'input[name="searching\_city"]', function() {

district\_check = 0;

var cityId = $(this).val();

$.ajax({

url: 'get\_districts\_index.php',

type: 'POST',

data: {cityId: cityId, districtCheck: district\_check}, // Передаем district\_check

success: function(response) {

$('#searching\_district').html(response);

}

});

});

// Обработчик события на изменение выбранного района

$(document).on('change', 'input[name="searching\_district"]', function() {

var districtId = $(this).val();

district\_check = $(districtId).is(':checked') ? 1 : 0; // Обновляем district\_check

localStorage.setItem('district\_check', district\_check); // Сохраняем district\_check в localStorage

$.ajax({

url: 'get\_districts\_index.php',

type: 'POST',

data: {cityId: firstCity, districtCheck: district\_check}, // Передаем district\_check

success: function(response) {

// Обновляем содержимое, если необходимо

}

});

});

});

</script>

Если в навигационном меню выбрать пункт "Новости", откроется страница со всеми опубликованными новостями и информацией по ним (см. Рисунок 55). Каждая новость содержит параметр важности. В зависимости от этого параметра новость может иметь специальную приписку, подчёркивающую её значимость. Кроме того, непрочитанные важные новости подсвечиваются мигающим красным цветом, что сигнализирует о необходимости их просмотра.

Также реализована фильтрация по типу новости: "Все", "Новости", "Опросы", "Инструкции". При выборе нужного типа отображаются только соответствующие записи (см. Рисунок 56). Обычные новости не имеют приписок и не подсвечиваются.

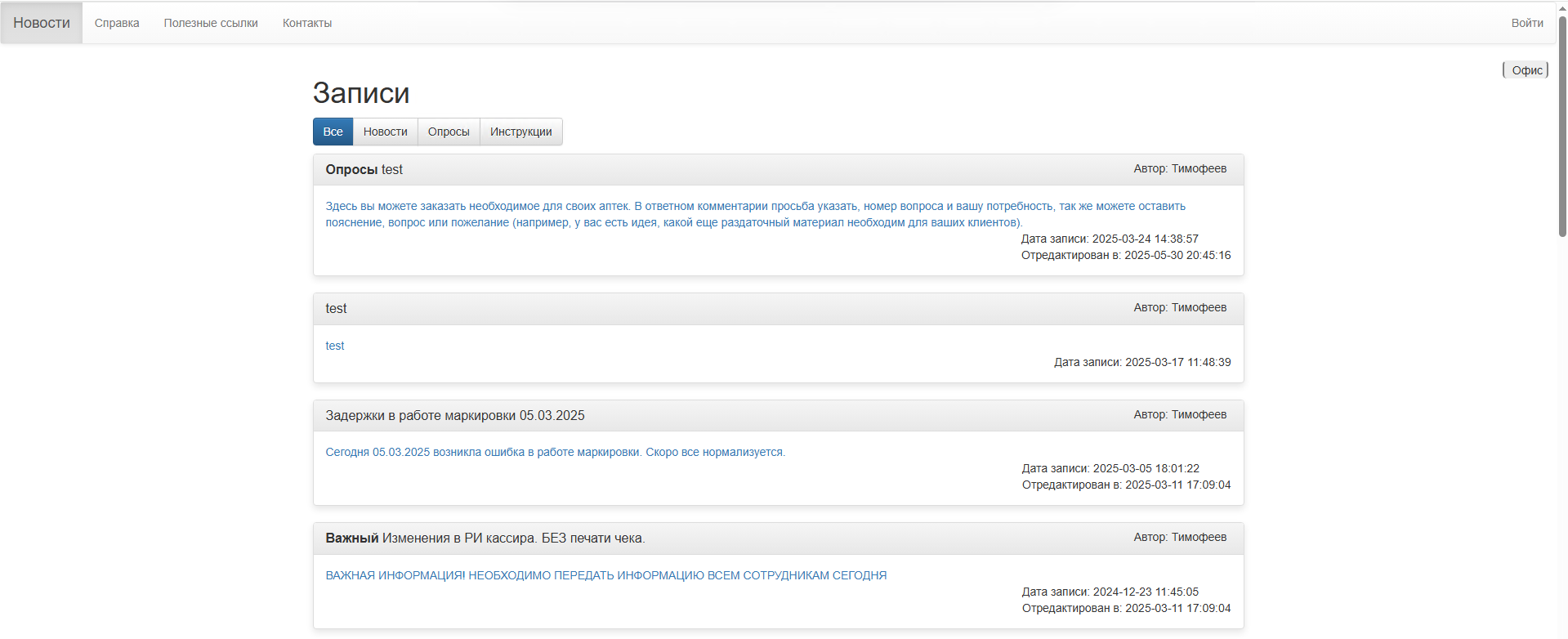


Рисунок 55 Страница “Новости”

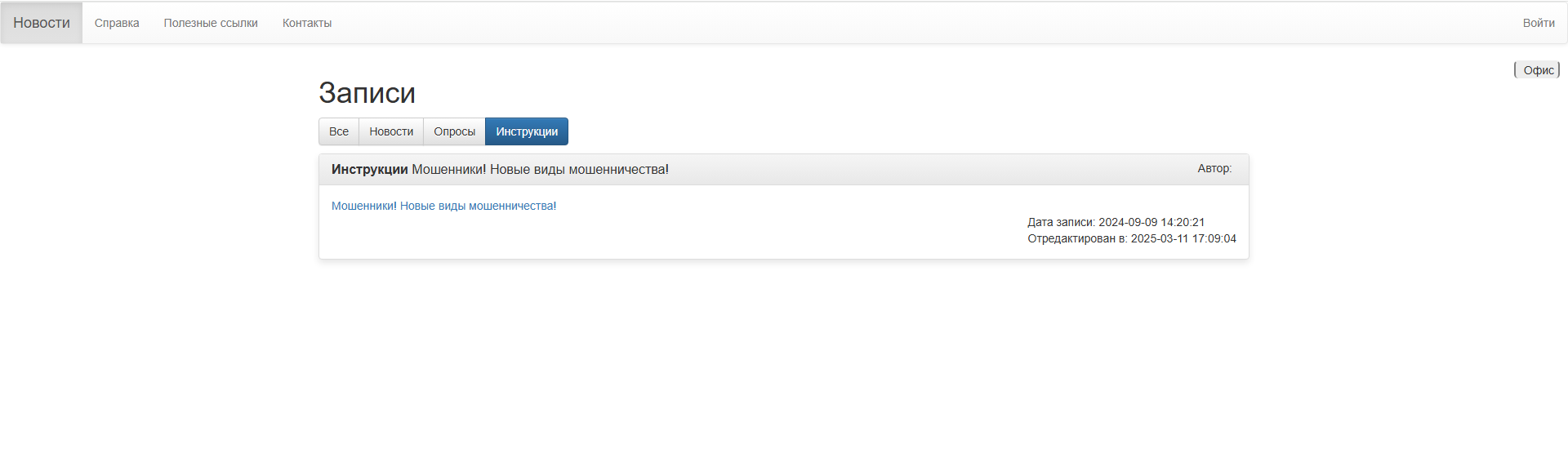


Рисунок 56 Страница “Новости” с фильтрацией

Для отображения новостей реализована логика, учитывающая тип записи, её важность, статус прочтения, наличие комментариев или ответов, а также доступность для конкретного пользователя. В зависимости от параметров записи добавляется визуальное выделение (например, мигание для важных непрочитанных новостей), а также приписки: "Актуально", "Важный", либо указание типа (например, "Опрос"). Отображаются дата создания, срок актуальности, редактор и автор записи. Листинг 3 демонстрирует ключевой фрагмент, отвечающий за генерацию и отображение новостей.

Листинг 3 – Вывод новостей

<?php  
$current\_user\_id = finduser(3); // Получение текущего ID пользователя  
// Собераем все связи постов и опросов  
$survey\_map = []; // post\_id => survey\_id  
$survey\_query = "SELECT id, post\_id FROM surveys";  
$survey\_result = $connect->query($survey\_query);  
if ($survey\_result && $survey\_result->num\_rows > 0) {  
 while ($row = $survey\_result->fetch\_assoc()) {  
 $survey\_map[$row['post\_id']] = $row['id'];  
 }  
}  
// Потом загрузим все ответы пользователя  
$user\_answers = []; // survey\_id => true  
$user\_answer\_query = "SELECT survey\_id FROM user\_answers WHERE user = $current\_user\_id";  
$user\_answer\_result = $connect->query($user\_answer\_query);  
if ($user\_answer\_result && $user\_answer\_result->num\_rows > 0) {  
 while ($row = $user\_answer\_result->fetch\_assoc()) {  
 $user\_answers[$row['survey\_id']] = true;  
 }  
}  
if ($result\_posts && $result\_posts->num\_rows > 0) {  
 while ($post = $result\_posts->fetch\_assoc()) {  
 $isanswered = true; // Флаг, указывающий на наличие ответа  
 $isviewed = false;  
 $selected\_pharmacies = explode(',', $post['visibility']);  
 $enable = false;  
 foreach ($selected\_pharmacies as $pharmacy) {  
 if ($current\_user\_id == $pharmacy || $current\_user\_id == 100000 || $current\_user\_id == 200000 || $pharmacy == 0) {  
 $enable = true;  
 break;  
 }  
 }  
 if (($enable && $post['is\_public'] == 1) || $isadmin == 1) {  
 $post\_id = $post['id'];  
 // SQL-запрос для получения комментариев к записи  
 $sql\_comments = "SELECT \* FROM comments WHERE post\_id = $post\_id AND is\_public = 1";  
 $result\_comments = $connect->query($sql\_comments);  
 // SQL-запрос для получения логов (просмотров)  
 $sql\_logs = "SELECT \* FROM logs WHERE postid = $post\_id";  
 $result\_logs = $connect->query($sql\_logs);  
 $has\_comment = false;  
 $has\_answer = false;  
 // Проверка комментариев  
 if ($result\_comments && $result\_comments->num\_rows > 0) {  
 while ($comment = $result\_comments->fetch\_assoc()) {  
 $user\_id\_to\_check = $comment['user\_id'];  
 if (($user\_id\_to\_check == $current\_user\_id && !isset($\_SESSION['user\_id'])) || (isset($\_SESSION['user\_id'])) || ($current\_user\_id == 100000)) {  
 $has\_comment = true;  
 break;  
 }  
 }  
 }  
 // Проверка ответа на опрос (если пост - опрос)  
 if ($post['post\_type'] == '2') {  
 if (isset($survey\_map[$post\_id])) {  
 $survey\_id = $survey\_map[$post\_id];  
 if (isset($user\_answers[$survey\_id])) {  
 $has\_answer = true;  
 }  
 }  
 }  
 // Итоговая проверка  
 if (!$has\_comment && !$has\_answer) {  
 if (!((isset($\_SESSION['user\_id'])) || ($current\_user\_id == 100000))) {  
 $isanswered = false;  
 }  
 }  
 // Проверка на просмотр  
 if ($result\_logs && $result\_logs->num\_rows > 0) {  
 while ($log = $result\_logs->fetch\_assoc()) {  
 if ($log['user'] == findip()) {  
 $isviewed = true;  
 break;  
 }  
 }  
 } elseif ((isset($\_SESSION['user\_id'])) || ($current\_user\_id == 100000)) {  
 $isviewed = true;  
 } ?>  
 <div class="panel panel-default">  
 <div class="panel-heading <?php if (($post['post\_type'] == '2' && !$isanswered) || (($post['importance'] == 'important'|| $post['dateimportance'] >= date("Y-m-d")) && !$isviewed)) echo ' blinking'; ?>">  
 <?php if (isset($\_SESSION["user\_id"]) && $post['is\_public'] == 1) { ?>  
 <div class="btn-group pull-right" style="margin-top:-6px">  
 <a href="deletenews.php?id=<?php echo $post\_id; ?>" class="btn btn-danger btn-sm" onclick="return confirm('Вы уверены, что хотите удалить эту запись?');">Удалить</a>  
 </div>  
 <?php }   
 if($post['is\_public'] == 0 && $isadmin == 1){  
 $postid = $post['id'];  
 echo "<div class='btn-group pull-right' style='margin-top:-6px'>";  
 echo "<a href='retnews.php?id=$postid' class='btn btn-warning btn-sm' onclick='return confirmReturn();'>Вернуть новость</a>";  
 echo "</div>";  
 }  
 ?>  
 <?php   
 // Отображение имени редактора, если запись была отредактирована  
 if ($post['autor'] != $post['lasteditor']) {  
 echo "<div class='btn-group pull-right' style='margin-top:-3px; margin-right: 5px'>Отредактирован:";  
 $sql\_user = "SELECT name FROM users WHERE id = " . $post['lasteditor'];   
 $result\_user = $connect->query($sql\_user);  
 if ($result\_user && $result\_user->num\_rows > 0) {  
 $user = $result\_user->fetch\_assoc();  
 echo htmlspecialchars($user['name']);  
 }  
 echo "</div>";  
 }  
 ?>  
 <div class="btn-group pull-right" style="margin-top:-3px; margin-right: 5px">Автор: <?php  
 $sql\_user = "SELECT name FROM users WHERE id = " . $post['autor'];   
 $result\_user = $connect->query($sql\_user);  
 if ($result\_user && $result\_user->num\_rows > 0) {  
 $user = $result\_user->fetch\_assoc();  
 echo htmlspecialchars($user['name']);  
 }  
 ?></div>  
 <h3 class="panel-title">  
 <a href='post.php?id=<?php echo $post\_id; ?>'>  
 <b>  
 <?php   
 // Проверка для актуальности  
 if ($post['dateimportance'] >= date("Y-m-d")) echo 'Актуально ';  
 // Проверка для типа поста, выводим название типа из базы  
 if ($post['post\_type\_name'] != 'Новости') echo $post['post\_type\_name']. " ";  
 // Проверка для важности  
 if ($post['importance'] == 'important') echo 'Важный ';  
 ?>  
 </b>  
 <?php echo htmlspecialchars($post['title']); ?>  
 </a>  
 </h3>  
 </div>  
 <div class="panel-body">  
 <div ><a style="word-wrap: break-word; word-break: break-word;" href='post.php?id=<?php echo $post\_id; ?>'><?php echo htmlspecialchars\_decode(stripslashes($post['short\_description'])); ?></a></div>  
 <div class='btn-group pull-right'>  
 <div>Дата записи: <?php echo $post['date\_created']; ?></div>  
 <?php   
 if ($post['date\_created'] != $post['date\_updated']) {  
 echo "<div>Отредактирован в: " . htmlspecialchars($post['date\_updated']) . "</div>";  
 }  
 if ($post['dateimportance'] >= date("Y-m-d")) {  
 echo "<div>Актуально до: " . htmlspecialchars($post['dateimportance']) . "</div>";  
 }  
 ?>  
 </div>  
 </div>  
 </div>  
 <?php  
 }  
 }  
 } else {  
 ?>  
 <p>Нет записей</p>  
 <?php  
 }  
 ?>  
</div>

При авторизации пользователя в верхней части сайта появляется кнопка "Добавить", а у каждой новости отображается кнопка для её удаления (см. Рисунок 57). При нажатии на кнопку "Добавить" открывается страница создания новости (см. Рисунок 58). Если при создании выбрать тип новости "Опрос", становится доступен функционал для добавления опроса к новости (см. Рисунок 59).

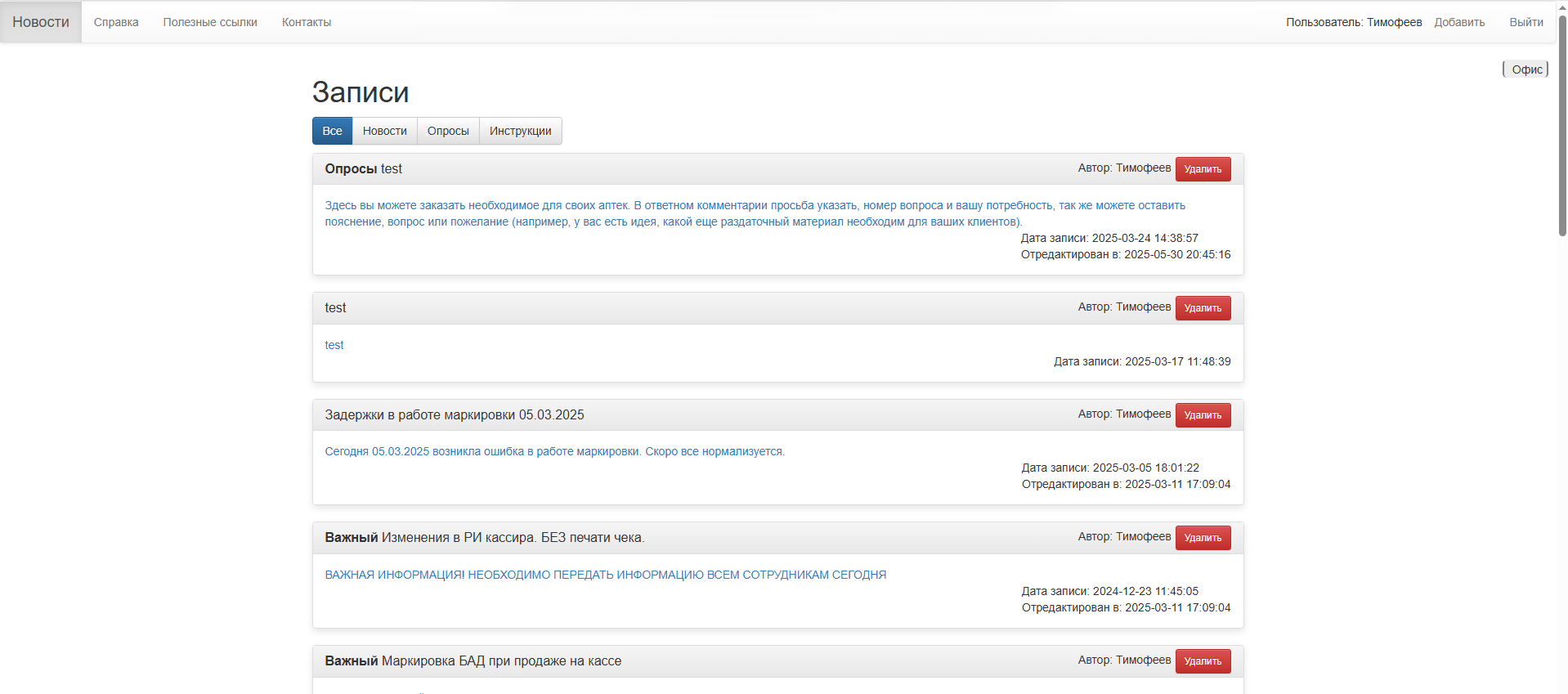


Рисунок 57 Страница “Новости” авторизованным пользователем

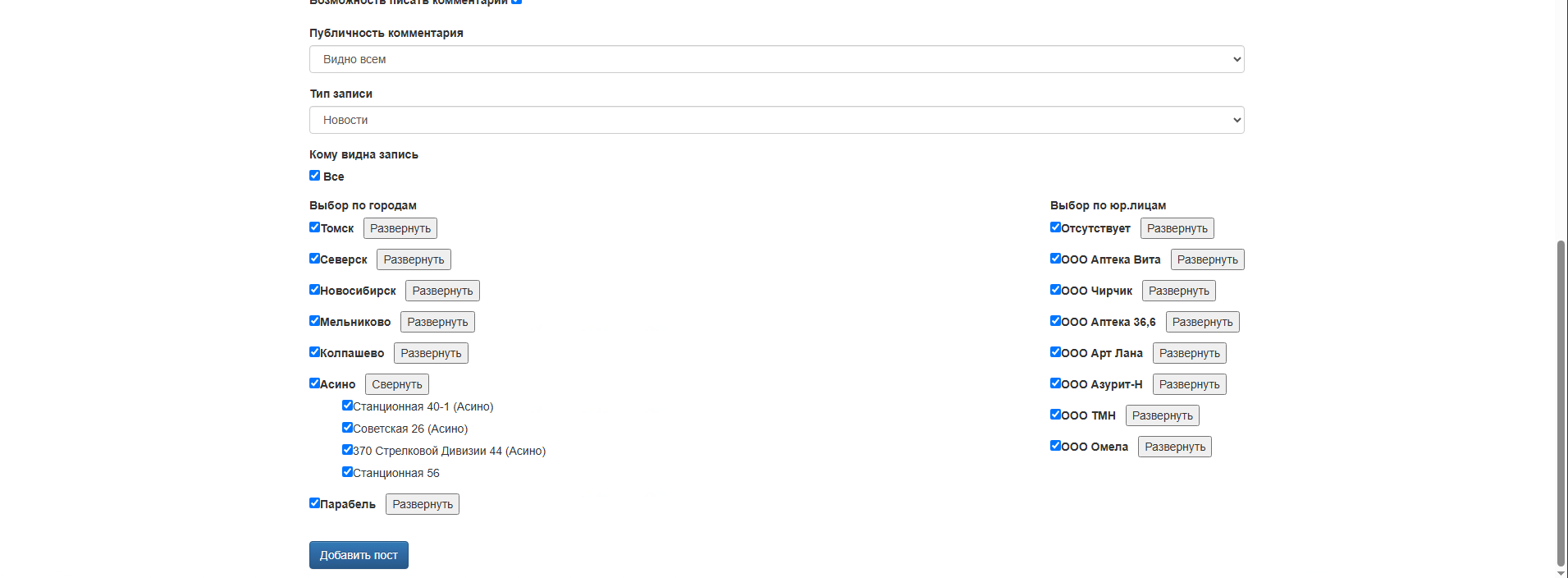
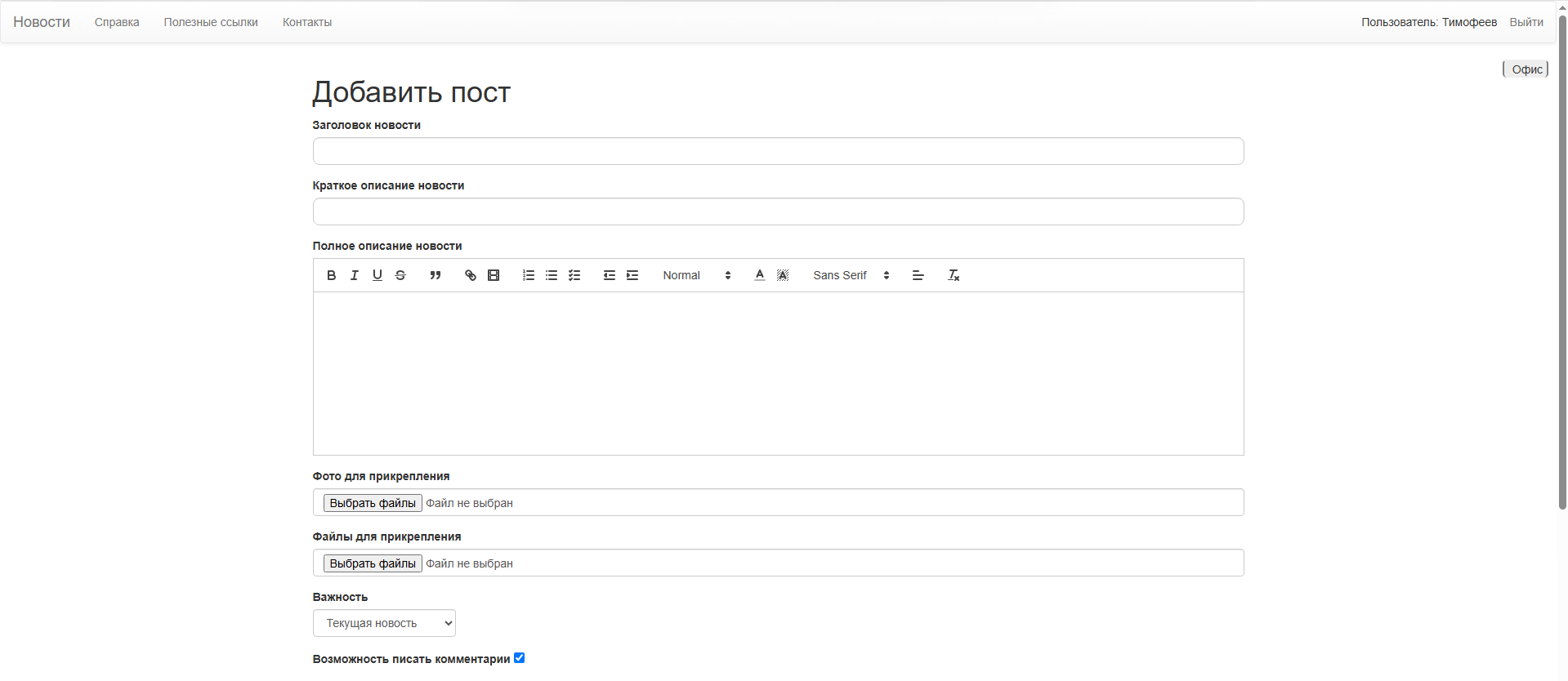


Рисунок 58 Страница создания новости



Рисунок 59 Страница создания новости создание опроса

В Листинге 4 показана форма для создания новости. В ней можно ввести заголовок, краткое и полное описание, добавить фотографии и файлы. Также можно выбрать важность новости, разрешить или запретить комментарии, выбрать тип записи и настроить, кто будет видеть эту новость — по городам или юридическим лицам. В скрипте реализована логика раскрытия/сокрытия списка аптек, синхронизации чекбоксов городов и аптек, а также инициализация редактора текста с помощью Quill для полного описания новости. Скрипт также отслеживает изменение типа записи, чтобы показывать или скрывать блок опроса.

Листинг 4 – Форма и часть скрипта создания новости

<form action="addnews.php" method="post" enctype="multipart/form-data">  
 <div class="form-group">  
 <label for="title">Заголовок новости</label>  
 <input type="text" class="form-control" id="title" name="title" required>  
 </div>  
 <div class="form-group">  
 <label for="short\_description">Краткое описание новости</label>  
 <input type="text" class="form-control" id="short\_description" name="short\_description" required>  
 </div>  
 <div class="form-group">  
 <label for="full\_description">Полное описание новости</label>  
 <div id="full\_description\_editor" class="editor-container"></div>  
 <input type="hidden" id="full\_description" name="full\_description" required>  
 </div>  
 <div class="form-group">  
 <label for="attachment\_fotos">Фото для прикрепления</label>  
 <input type="file" class="form-control" id="attachment\_fotos" name="attachment\_fotos[]" multiple>  
 </div>  
 <div class="form-group">  
 <label for="attachment\_files">Файлы для прикрепления</label>  
 <input type="file" class="form-control" id="attachment\_files" name="attachment\_files[]" multiple>  
 </div>  
 <div class="form-group" style="display: flex; align-items: center;">  
 <div>  
 <label for="importance">Важность</label>  
 <select class="form-control" id="importance" name="importance" required>  
 <option value="current">Текущая новость</option>  
 <option value="important">Важная новость</option>  
 <option value="temporary">Временная новость</option>  
 </select>  
 </div>  
 <div style="margin-left: 40px; display:none; flex-direction: column;" id="divdate" >  
 <label>Выбирите дату до которой актуальна запись</label>  
 <input type="date" id="dateimportance" name="dateimportance">  
 </div>  
 </div>  
 <div class="form-group">  
 <label for="allow\_comments">Возможность писать комментарии</label>  
 <input type="checkbox" id="allow\_comments" name="allow\_comments" checked>  
 </div>  
 <div class="form-group">  
 <label for="comment\_visibility">Публичность комментария</label>  
 <select class="form-control" id="comment\_visibility" name="comment\_visibility" required>  
 <option value="public" selected>Видно всем</option>  
 <option value="private">Видно только создателю статьи</option>  
 </select>  
 </div>  
 <div class="form-group">  
 <label for="post\_type">Тип записи</label>  
 <select class="form-control" id="post\_type" name="post\_type" required >  
 <?php  
 // Проходим по результатам запроса и генерируем <option> для каждого типа  
 while ($row = mysqli\_fetch\_assoc($resulttypes)) {  
 $post\_type\_id = $row['id'];  
 $post\_type\_name = htmlspecialchars($row['name']);  
 echo "<option value='$post\_type\_id'>$post\_type\_name</option>";  
 }  
 ?>  
 </select>  
 </div>  
 <div id="survey\_section" style="display: none;">  
 <h3>Опрос:</h3>  
 <div id="questions"></div>  
 <button id="add\_question" type="button" style="margin: 10px 0;" class="btn btn-success">Добавить вопрос</button>  
 </div>  
 <div class="form-group">  
 <label for="visibility">Кому видна запись</label>  
 <div > <input style='cursor:pointer;' type='checkbox' data-partners="0" id="Все" name='cities[]' value='0' data-city='0' checked>  
 <label style='cursor:pointer;' for='Все'>Все</label><ul></ul>  
 <div style="display:flex; justify-content: space-between;">  
 <div class='form-group'>  
 <label>Выбор по городам</label>  
 <div class='form-group'>  
 <?php  
 // Сначала создаем ассоциативный массив, в котором ключами будут ID городов, а значениями - массивы с аптеками  
 $cities\_with\_pharmacies = []; // Запрос на получение всех городов  
 $query\_cities = "SELECT \* FROM cities ";  
 $result\_cities = mysqli\_query($connect, $query\_cities);  
 while ($row\_city = mysqli\_fetch\_assoc($result\_cities)) {  
 $city\_id = $row\_city['idcity'];  
 $city\_name = $row\_city['namecity'];   
 // Запрос на получение аптек в данном городе  
 $query\_pharmacies = "SELECT \* FROM partners WHERE city = ? and type = 1";  
 $stmt\_pharmacies = $connect->prepare($query\_pharmacies);  
 $stmt\_pharmacies->bind\_param("i", $city\_id);  
 $stmt\_pharmacies->execute();  
 $result\_pharmacies = $stmt\_pharmacies->get\_result();  
   
 $pharmacy\_names = [];  
 while ($row\_pharmacy = $result\_pharmacies->fetch\_assoc()) {  
 $pharmacy\_names[$row\_pharmacy['itemcode']] = $row\_pharmacy['itemname']; // Сохраняем ID аптеки как ключ массива  
 }  
 // Добавляем город и его аптеки в массив  
 $cities\_with\_pharmacies[$city\_name] = $pharmacy\_names;  
 }  
 // Теперь отображаем данные на странице  
 $i = 0;  
 $j = 100;  
 foreach ($cities\_with\_pharmacies as $cities\_name => $pharmacies) {  
 $i++;  
 $visible = $i . "visible";  
 $visibleb = "visible".$i;  
 // Включаем идентификатор города в идентификаторы чекбоксов аптеки  
 echo "<label style='cursor:pointer' for='$i'><input type='checkbox' id='$i' name='cities[]' checked value='$i'>$cities\_name</label>";  
 echo "<input type='button' id='$visibleb' name='visible' style='margin-left:12px' class='visible-checkbox' value='Развернуть'>";  
 echo "<ul >";  
 foreach ($pharmacies as $pharmacy\_id => $pharmacy\_name) {  
 $j++;  
 echo "<li style='list-style-type: none; display:none;' name='$visible'>";  
 echo "<input checked type='checkbox' id='$j' data-partners='$i$j' name='partners[]' value='$pharmacy\_id' data-city='$i'>";  
 echo "<label style='cursor:pointer; font-weight: normal;' for='$j'>$pharmacy\_name</label></li>";  
 }  
 echo "</ul>";  
 }  
 ?>  
 </div> </div>  
 <div class="form-group">  
 <label>Выбор по юр.лицам</label>  
 <?php  
 // Сначала создаем ассоциативный массив, в котором ключами будут ID городов, а значениями - массивы с аптеками  
 $firm\_with\_pharmacies = [];  
 // Запрос на получение всех городов  
 $query\_firm = "SELECT \* FROM firm ";  
 $result\_firm = mysqli\_query($connect, $query\_firm);  
 while ($row\_firm = mysqli\_fetch\_assoc($result\_firm)) {  
 $firm\_id = $row\_firm['id'];  
 $firm\_name = $row\_firm['name']; // Запрос на получение аптек в данном городе  
 $query\_pharmacies = "SELECT \* FROM partners WHERE firm = ? and type = 1";  
 $stmt\_pharmacies = $connect->prepare($query\_pharmacies);  
 $stmt\_pharmacies->bind\_param("i", $firm\_id);  
 $stmt\_pharmacies->execute();  
 $result\_pharmacies = $stmt\_pharmacies->get\_result();  
 $pharmacy\_names = [];  
 while ($row\_pharmacy = $result\_pharmacies->fetch\_assoc()) {  
 $pharmacy\_names[$row\_pharmacy['itemcode']] = $row\_pharmacy['itemname']; // Сохраняем ID аптеки как ключ массива  
 }  
 // Добавляем город и его аптеки в массив  
 $firm\_with\_pharmacies[$firm\_name] = $pharmacy\_names;  
 }  
 // Теперь отображаем данные на странице  
 echo "<div class='form-group'>";  
 foreach ($firm\_with\_pharmacies as $firm\_name => $pharmacies) {  
 $i++;  
 $visible = $i . "visible";  
 $visibleb = "visible".$i;  
 // Включаем идентификатор города в идентификаторы чекбоксов аптеки  
 echo "<label style='cursor:pointer' for='$i'><input type='checkbox' checked id='$i' name='firm[]' value='$i'>$firm\_name</label>";  
 echo "<input type='button' id='$visibleb' name='visible' style='margin-left:12px' class='visible-checkbox' value='Развернуть'>";  
 echo "<ul >";  
 foreach ($pharmacies as $pharmacy\_id => $pharmacy\_name) {  
 $j++;  
 echo "<li style='list-style-type: none; display:none;' name='$visible'>";  
 echo "<input type='checkbox' checked id='$j' data-partners='$i$j' value='$pharmacy\_id' data-firm='$i'>";  
 echo "<label style='cursor:pointer; font-weight: normal;' for='$j'>$pharmacy\_name</label></li>";  
 }  
 echo "</ul>";  
 }  
 echo "</div>";  
 ?>  
 </div>  
 </div>  
 </div>  
   
 <button type="submit" class="btn btn-primary">Добавить пост</button>  
</form>

<script defer>  
*document*.addEventListener('DOMContentLoaded', function() {  
 const citiesCheckboxes = *document*.querySelectorAll('input[name="cities[]"]');  
 const firmCheckboxes = *document*.querySelectorAll('input[name="firm[]"]');  
 const pharmaciesCheckboxes = *document*.querySelectorAll('input[data-partners]');  
 const citiespharmaciesCheckboxes = *document*.querySelectorAll('input[data-city]');  
 const firmpharmaciesCheckboxes = *document*.querySelectorAll('input[data-firm]');  
 const visibilityCheckboxes = *document*.querySelectorAll('input[name="visible"]');  
 const datepicker = *document*.querySelector('input[name="dateimportance"]');  
 const combobox = *document*.getElementById('importance');  
 const divdate = *document*.getElementById('divdate');  
 const allCheckbox = *document*.getElementById('Все');  
 const surveyTypeId = <?php echo $surveyType; ?>;   
 let postTypeSelect = *document*.getElementById('post\_type');  
 postTypeSelect.addEventListener('change', toggleSurveyFields);   
 function toggleSurveyFields() {  
 let surveySection = *document*.getElementById('survey\_section');  
 let postType = postTypeSelect.value;  
 let surveyTypeId = "2";  
 surveySection.style.display = postType === surveyTypeId ? 'block' : 'none';  
 } let addQuestionButton = *document*.getElementById('add\_question');   
 if (addQuestionButton) {  
 addQuestionButton.addEventListener('click', addQuestion);  
 } function addQuestion() {  
 let container = *document*.getElementById('questions');   
 let div = *document*.createElement('div');  
 let questionIndex = container.children.length; // Получаем индекс текущего вопроса  
 div.classList.add('question');  
 div.innerHTML = `  
 <label>Вопрос:</label>  
 <div class='d-flex'>  
 <input type='text' name='questions[]' required>  
 <button type='button' style='margin-left: 10px' class='btn btn-danger remove\_question'>Удалить вопрос</button>  
 </div>  
 <br>  
 <label>Формат ответа:</label>  
 <select name='answer\_types[${questionIndex}]' required>  
 <option value='1'>Ответ текстом</option>  
 <option value='2' selected>Выбор одного варианта</option>  
 <option value='3'>Выбор нескольких вариантов</option>  
 </select>  
 <div class='answers'></div>  
 <button type='button' style='margin: 10px 0' class='btn btn-xs btn-success add\_answer'>Добавить ответ</button>  
 `;  
 container.appendChild(div);   
 // Вешаем обработчик события на кнопку "Добавить ответ"  
 div.querySelector('.add\_answer').addEventListener('click', function () {  
 addAnswer(this, questionIndex);  
 });  
 // Вешаем обработчик события на кнопку "Удалить вопрос"  
 div.querySelector('.remove\_question').addEventListener('click', function () {  
 removeQuestion(this, questionIndex);  
 });  
 } function addAnswer(button, questionIndex) {  
 let container = button.previousElementSibling; // Получаем .answers  
 let answerType = *document*.querySelector(`select[name='answer\_types[${questionIndex}]']`).value; // Получаем тип ответа  
 // Если тип ответа "текст", не добавляем варианты  
 if (answerType === '1') return;   
 let div = *document*.createElement('div');  
 div.classList.add('answer');  
 if (answerType === '2' || answerType === '3') {  
 div.innerHTML = `  
 <div class='d-flex'>  
 <label>Ответ:</label>  
 <input type='text' name='answers[${questionIndex}][]' required>  
 <button type='button' style='margin-left: 10px' class='btn btn-warning remove\_answer'>Удалить ответ</button>  
 </div>  
 `;  
 } else {  
 div.innerHTML = `  
 <div class='d-flex'>  
 <label>Ответ:</label>  
 <input type='text' name='answers[${questionIndex}][]' required>  
 <button type='button' style='margin-left: 10px' class='btn btn-warning remove\_answer'>Удалить ответ</button>  
 </div>  
 `;  
 }  
 container.appendChild(div);   
 // Вешаем обработчик события на кнопку "Удалить ответ"  
 div.querySelector('.remove\_answer').addEventListener('click', function () {  
 removeAnswer(this);  
 });  
 }

visibilityCheckboxes.forEach(button => {  
 button.addEventListener('click', function() {  
 let firmId = this.id;  
 firmId = firmId.replace(/\D/g, '');  
 const pharmaciesList = *document*.getElementsByName(firmId + 'visible');  
 if (this.click && this.value === "Развернуть") {  
 pharmaciesList.forEach(pharmacy => pharmacy.style.display = 'block');  
 this.value = "Свернуть";  
 } else {  
 pharmaciesList.forEach(pharmacy => pharmacy.style.display = 'none');  
 this.value = "Развернуть";  
 }  
 });  
});  
function updatePharmaciesByCity(cityCheckbox) {  
 const cityId = cityCheckbox.value;  
 if (cityId === '0') {  
 citiesCheckboxes.forEach(city => city.checked = cityCheckbox.checked);  
 pharmaciesCheckboxes.forEach(pharmacy => pharmacy.checked = cityCheckbox.checked);  
 } else {  
 pharmaciesCheckboxes.forEach(pharmacyCheckbox => {  
 if (pharmacyCheckbox.dataset.city === cityId) {  
 pharmacyCheckbox.checked = cityCheckbox.checked;  
 }  
 });  
 }  
 updateCityAndFirmStates();  
 updateCityPharmaciesWithSameValue();  
 updateAllCheckboxState();   
}

var quillFull = new Quill('#full\_description\_editor', {  
 theme: 'snow',  
 modules: {  
 toolbar: [  
 ['bold', 'italic', 'underline', 'strike'],  
 ['blockquote'],  
 ['link', 'video'],  
 [{ 'list': 'ordered' }, { 'list': 'bullet' }, { 'list': 'check' }],  
 [{ 'indent': '-1' }, { 'indent': '+1' }],  
 [{ 'header': [1, 2, 3, 4, 5, 6, false] }],  
 [{ 'color': [] }, { 'background': [] }],  
 [{ 'font': [] }],  
 [{ 'align': [] }],  
 ['clean']  
 ]  
 }  
 });  
 *document*.querySelector('form').onsubmit = function() {  
 *document*.querySelector('#full\_description').value = quillFull.root.innerHTML;  
 };  
});  
</script>

При нажатии на новость открывается страница с полным описанием и при наличии с прикреплённым к ней опросом (см. Рисунок 60). Кнопки "Редактировать", "Просмотреть лог записи", "Кто ответил", "Просмотреть ответы" доступны только авторизованным пользователям и скрыты для остальных.

При нажатии на кнопку "Редактировать" открывается страница редактирования новости (см. Рисунок 58). Кнопка "Просмотреть лог записи" открывает страницу с журналом действий, связанных с данной записью (см. Рисунок 61). Кнопка "Кто ответил" открывает таблицу с аптеками, где отображается, оставили ли они комментарий. При нажатии на "Просмотреть ответы" открывается страница со списком ответов на опрос и информацией о том, кто именно ответил (см. Рисунок 62).

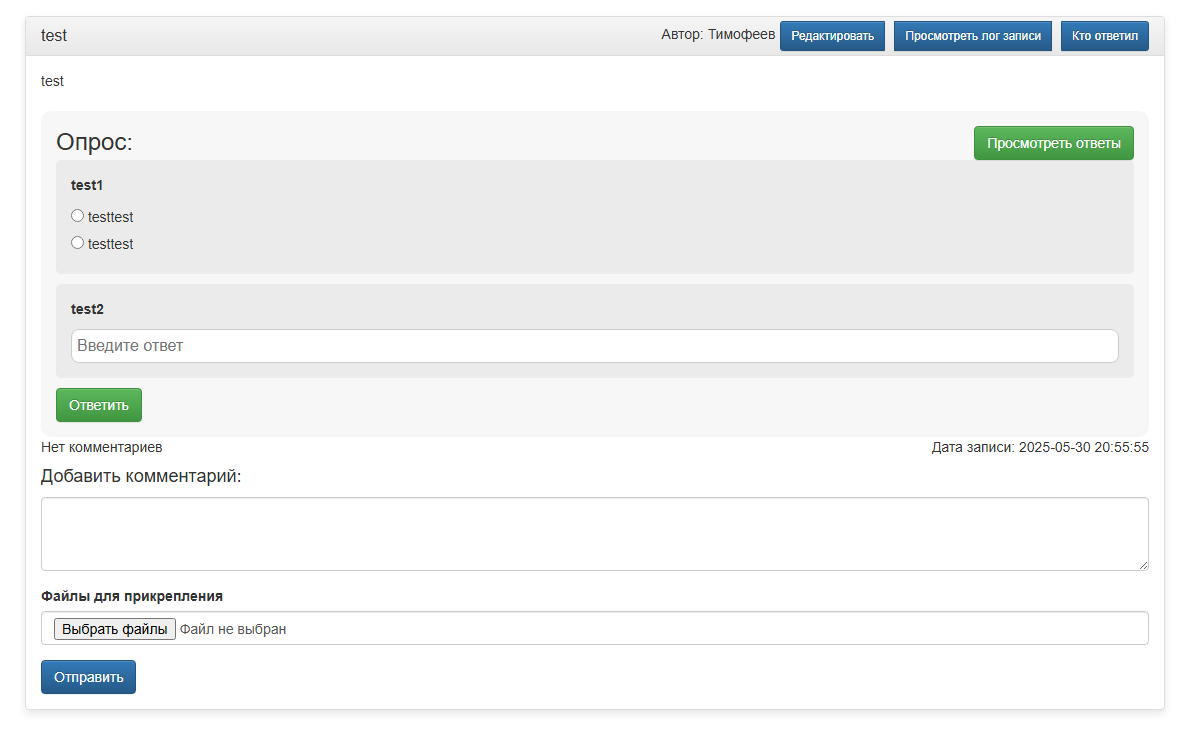


Рисунок 60 Страница просмотра новости

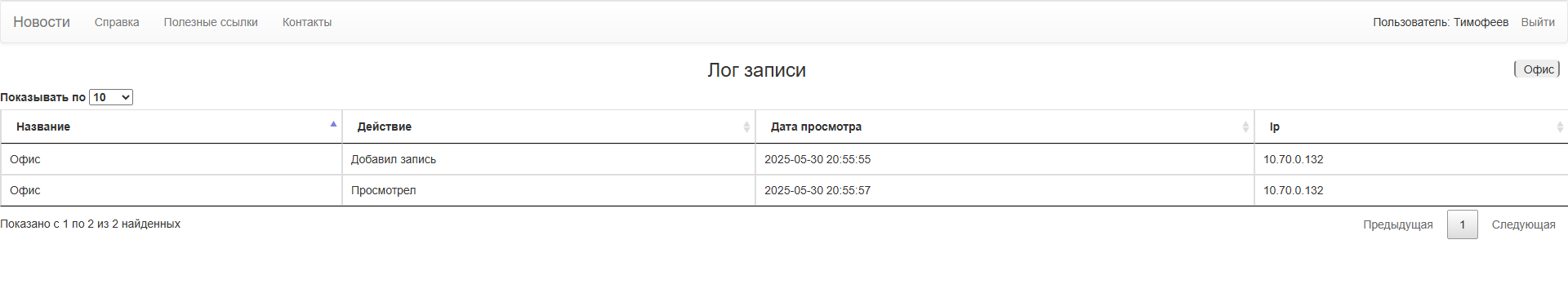


Рисунок 61 Страница просмотра логов новости

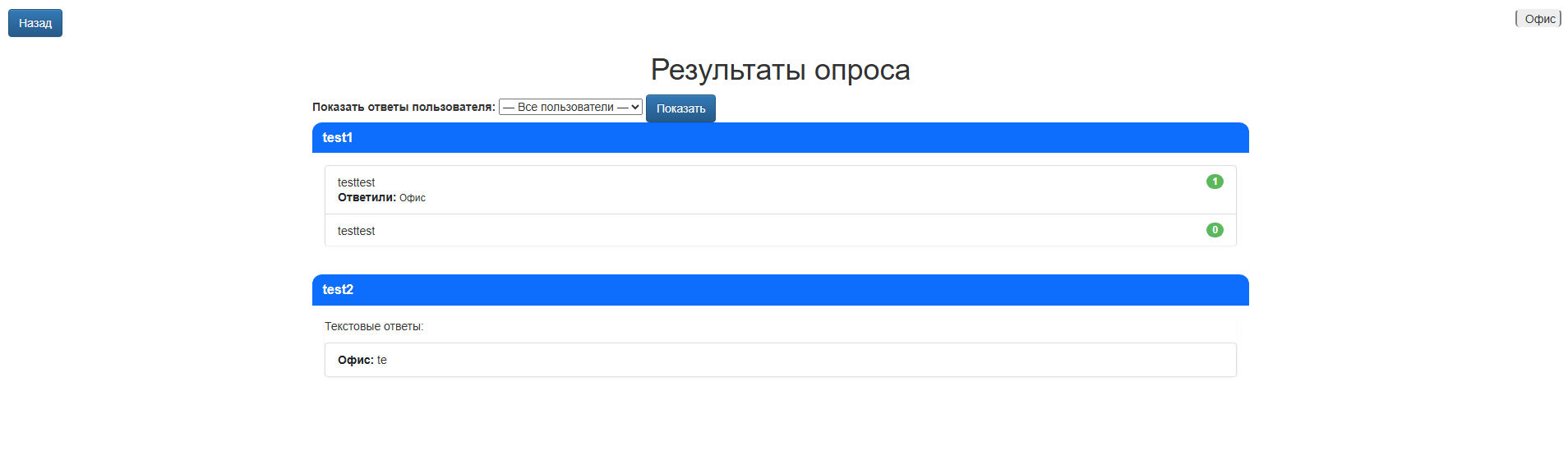


Рисунок 62 Страница просмотра результатов опроса

В Листинге 5 показан фрагмент кода, который выводит на страницу новость с заголовком, полным описанием, опросом, фотографиями и файлами. В коде реализованы элементы управления — кнопки редактирования и просмотра логов для авторизованных пользователей, а также отображение комментариев с учетом прав доступа.

Листинг 5 – Вывод новости

<div class="container">  
 <?php  
 if ($result\_posts && $result\_posts->num\_rows > 0) {  
 while ($post = $result\_posts->fetch\_assoc()) {  
 $post\_id = $post['id'];  
 ?>  
<div class="panel panel-default">  
 <div class="panel-heading">  
 <?php if (isset($\_SESSION["user\_id"])) { ?>  
 <div class="btn-group pull-right" style="margin-top:-6px" >  
 <a href="editnews.php?id=<?php echo $post\_id; ?>" style="margin-right:10px" class="btn btn-primary btn-sm">Редактировать</a>  
 <a href="whoview.php?id=<?php echo $post\_id; ?>" style="margin-right:10px" class="btn btn-primary btn-sm">Просмотреть лог записи</a>  
 <a href="whoansver.php?id=<?php echo $post\_id; ?>" class="btn btn-primary btn-sm">Кто ответил</a>  
 </div>  
 <?php } ?>  
 <?php   
 // Отображение имени редактора, если запись была отредактирована другим пользователем  
 if ($post['autor'] != $post['lasteditor']) {  
 echo "<div class='btn-group pull-right' style='margin-top:-3px; margin-right: 5px'>Отредактирован:";  
 $sql\_user = "SELECT name FROM users WHERE id = " . $post['lasteditor'];   
 $result\_user = $connect->query($sql\_user);  
 if ($result\_user && $result\_user->num\_rows > 0) {  
 $user = $result\_user->fetch\_assoc();  
 echo htmlspecialchars($user['name']);  
 }  
 echo "</div>";  
 }  
 ?>  
 <div class="btn-group pull-right" style="margin-top:-3px; margin-right: 5px">Автор: <?php  
 $sql\_user = "SELECT name FROM users WHERE id = " . $post['autor'];   
 $result\_user = $connect->query($sql\_user);  
 if ($result\_user && $result\_user->num\_rows > 0) {  
 $user = $result\_user->fetch\_assoc();  
 echo htmlspecialchars($user['name']);  
 }?></div>  
 <h3 class="panel-title"><?php echo htmlspecialchars($post['title']); ?></h3>  
 </div>  
 <div class="panel-body">  
 <div><?php echo htmlspecialchars\_decode(stripslashes($post['full\_description'])); ?></div>  
 <div class="foto">  
 <?php  
 if ($post['attachment\_fotos']) {  
 $attachments = json\_decode($post['attachment\_fotos'], true);  
 if ($attachments) {  
 foreach ($attachments as $file) {  
 $file\_name = basename($file);  
 echo "<a href='$file' target='\_blank' ><img style='max-width:250px;margin:2px' src='$file' alt='foto'></img></a>";  
 }  
 }  
 }  
 ?>  
 </div>  
 <?php  
 $user = finduser(3);  
 $user\_answers = [];  
 $query = "SELECT survey\_id, survey\_answer\_id FROM user\_answers WHERE user = ?";  
 $stmt = $connect->prepare($query);  
 $stmt->bind\_param("s", $user);  
 $stmt->execute();  
 $result = $stmt->get\_result();  
 while ($row = $result->fetch\_assoc()) {  
 if (!isset($user\_answers[$row['survey\_id']])) {  
 $user\_answers[$row['survey\_id']] = [];  
 }  
 $user\_answers[$row['survey\_id']][] = $row['survey\_answer\_id'];  
 }  
 $stmt->close();  
 $query = "SELECT \* FROM surveys WHERE post\_id = ?";  
 $stmt = $connect->prepare($query);  
 $stmt->bind\_param("i", $post\_id);  
 $stmt->execute();  
 $result = $stmt->get\_result();  
 $stmt->close();  
 if ($result->num\_rows > 0) {  
 echo "<div class='survey-container'>";  
 echo "<div class='survey-header'>";  
 echo "<h3 class='survey'>Опрос:</h3>";  
 if(isset($\_SESSION['user\_id'])){  
 echo "<a class='btn btn-success' href='view\_results.php?post\_id=" . $post['id'] . "'>Просмотреть ответы</a>";  
 }  
 echo "</div>";  
 echo "<form action='post.php?id=$post\_id' id='surveyform' method='post'>";  
 echo "<input type='hidden' name='post\_id' value='$post\_id'>";  
 while ($question = $result->fetch\_assoc()) {  
 echo "<div class='question'>";  
 echo "<p><strong>{$question['question']}</strong></p>";  
 // Получаем тип ответа  
 $type\_answer\_id = $question['type\_answer\_id'];  
 // Получаем ответы для текущего вопроса  
 $query\_answers = "SELECT \* FROM survey\_answers WHERE question\_id = ?";  
 $stmt\_answers = $connect->prepare($query\_answers);  
 $stmt\_answers->bind\_param("i", $question['id']);  
 $stmt\_answers->execute();  
 $answers = $stmt\_answers->get\_result();  
 if ($type\_answer\_id == 1) {   
 echo "<input type='text' name='answers[{$question['id']}]' placeholder='Введите ответ' >";  
 } if ($type\_answer\_id == 2) {   
 // Радио-кнопки (выбор одного варианта)  
 if ($answers->num\_rows > 0) {  
 while ($answer = $answers->fetch\_assoc()) {  
 echo "<label style='font-weight: 400;'>";  
 echo "<input type='radio' name='answers[{$question['id']}]' value='{$answer['id']}' > {$answer['answer']}";  
 echo "</label><br>";  
 }  
 }  
 } if ($type\_answer\_id == 3) {   
 // Чекбоксы (множественный выбор)  
 if ($answers->num\_rows > 0) {  
 while ($answer = $answers->fetch\_assoc()) {  
 echo "<label style='font-weight: 400;'>";  
 echo "<input type='checkbox' name='answers[{$question['id']}][]' value='{$answer['id']}'> {$answer['answer']}";  
 echo "</label><br>";  
 }  
 }   
 }  
 $stmt\_answers->close();  
 echo "</div>";  
 }  
 echo "<button class='btn btn-success' type='submit'>Ответить</button>";  
 echo "</form>";  
 echo "</div>";  
 }   
 ?>  
 <div>  
 <?php  
 if ($post['attachment\_files']) {  
 $attachments = json\_decode($post['attachment\_files'], true);  
 if ($attachments) {  
 echo "<div><b>Прикрепленные файлы</b></div>";  
 foreach ($attachments as $file) {  
 $file\_name = basename($file);  
 echo "<a href='$file' download>Скачать файл $file\_name</a><br>";  
 }  
 }  
 }  
 ?>  
 </div>  
 <div class='btn-group pull-right'>  
 <div>Дата записи: <?php echo $post['date\_created']; ?></div>  
 <?php   
 if ($post['date\_created'] != $post['date\_updated']) {  
 echo "<div>Отредактирован в:";  
 echo htmlspecialchars($post['date\_updated']);  
 echo "</div>";  
 }  
 ?>  
 </div>  
 <?php  
 // Получение и отображение комментариев к посту  
 $sql\_comments = "SELECT \* FROM comments WHERE post\_id = $post\_id ORDER BY date\_created DESC";  
 $result\_comments = $connect->query($sql\_comments);  
 if ($result\_comments && $result\_comments->num\_rows > 0) {  
 ?>  
 <h4 style="margin-top:40px">Комментарии:</h4>  
 <?php   
 $strpriv = "<div><b>Комментарии видны только создателю записи</b></div>";  
 while ($comment = $result\_comments->fetch\_assoc()) {  
 if ($comment['user\_id'] != finduser(3) && ($post['comment\_visibility'] == "private" && !isset($\_SESSION['user\_id']))){  
 if(isset($strpriv)){  
 echo $strpriv;  
 $strpriv = null;  
 }  
 }  
 elseif($comment['is\_public']== 1 || $isadmin == 1)  
 {  
 ?>  
 <div class="comment">  
 <p><b><?php   
 // Отображение имени пользователя, оставившего комментарий  
 if ($comment['user\_id'] < 9999) {  
 $sql\_user = "SELECT name FROM users WHERE id = " . $comment['user\_id'];  
 $result\_user = $connect->query($sql\_user);  
 if ($result\_user && $result\_user->num\_rows > 0) {  
 $user = $result\_user->fetch\_assoc();  
 echo htmlspecialchars($user['name']);  
 }  
 } elseif($comment['user\_id'] == 100000){  
 echo "Офис";  
 } elseif($comment['user\_id'] == 200000){  
 echo "Удаленка";  
 }  
 else {  
 $sql\_partner = "SELECT itemname, city FROM partners WHERE itemcode = " . $comment['user\_id'];  
 $result\_partner = $connect->query($sql\_partner);  
   
 if ($result\_partner && $result\_partner->num\_rows > 0) {  
 $partner = $result\_partner->fetch\_assoc();  
 $sql\_city = "SELECT namecity from cities WHERE idcity =". htmlspecialchars($partner['city']);  
 $result\_city = $connect->query($sql\_city);  
 $city = $result\_city->fetch\_assoc();  
 echo htmlspecialchars($partner['itemname']) . ', ' . htmlspecialchars($city['namecity']);  
 }  
 }  
 ?></b></p>  
 <?php   
 // Отображение кнопок для удаления или восстановления комментария  
 if (($comment['user\_id'] == finduser(3) || isset($\_SESSION['user\_id'])) && $comment['is\_public'] == 1 ) {  
 $commentid = $comment['id'];  
 echo "<div class='btn-group pull-right'>";  
 echo "<a href='delcom.php?id=$commentid' class='btn btn-danger ' onclick='return confirmDelete();'><i class='fas fa-trash'></i></a>";  
 echo "</div>";  
 }   
 if($comment['is\_public'] == 0 && $isadmin == 1){  
 $commentid = $comment['id'];  
 echo "<div class='btn-group pull-right'>";  
 echo "<a href='retcom.php?id=$commentid' class='btn btn-warning ' onclick='return confirmReturn();'>Вернуть комментарий</a>";  
 echo "</div>";  
 }  
 ?>  
 <p><?php echo nl2br(htmlspecialchars($comment['text'])); ?></p>  
 <?php  
 // Отображение прикрепленных файлов к комментариям  
 if ($comment['photo']) {  
 $photos = json\_decode($comment['photo'], true);  
 if ($photos) {  
 foreach ($photos as $photo) {  
 $photo\_name = basename($photo);  
 echo "<a href='$photo' download>Скачать файл $photo\_name</a><br>";  
 }  
 }  
 }  
 ?>  
 <p>Дата комментария: <?php echo $comment['date\_created']; ?></p>  
 </div>  
 <?php  
 }  
 }} else {  
 ?>  
 <p>Нет комментариев</p>  
 <?php  
 }  
 // Форма для добавления нового комментария  
 if ($post['allow\_comments']){?>  
 <h4>Добавить комментарий:</h4>  
 <form action="addcomment.php" method="post" enctype="multipart/form-data">  
 <input type="hidden" name="post\_id" value="<?php echo $post\_id; ?>">  
 <div class="form-group">  
 <textarea name="text" class="form-control" rows="3"></textarea>  
 </div>  
 <div class="form-group">  
 <label for="attachment\_files">Файлы для прикрепления</label>  
 <input type="file" class="form-control" id="attachment\_files" name="attachment\_files[]" multiple>  
 </div>  
 <button type="submit" class="btn btn-primary">Отправить</button>  
 </form>  
 </div>  
</div>  
 <?php  
 }else{  
 echo "<div><b>Комментарии отключены</b></div>";  
 }  
 }  
 } else {  
 ?>  
 <p>Нет записей</p>  
 <?php  
 }  
 ?>  
</div>

Если в навигационном меню выбрать пункт "Полезные ссылки", откроется одноимённая страница со списком всех ссылок, которые могут пригодиться в работе (см. Рисунок 63). При нажатии на кнопку "Данные для входа" открывается страница с учётными данными, связанными с выбранной ссылкой (см. Рисунок 64). На этой странице также доступна кнопка добавления, которая открывает форму для ввода новых данных для входа (см. Рисунок 65).

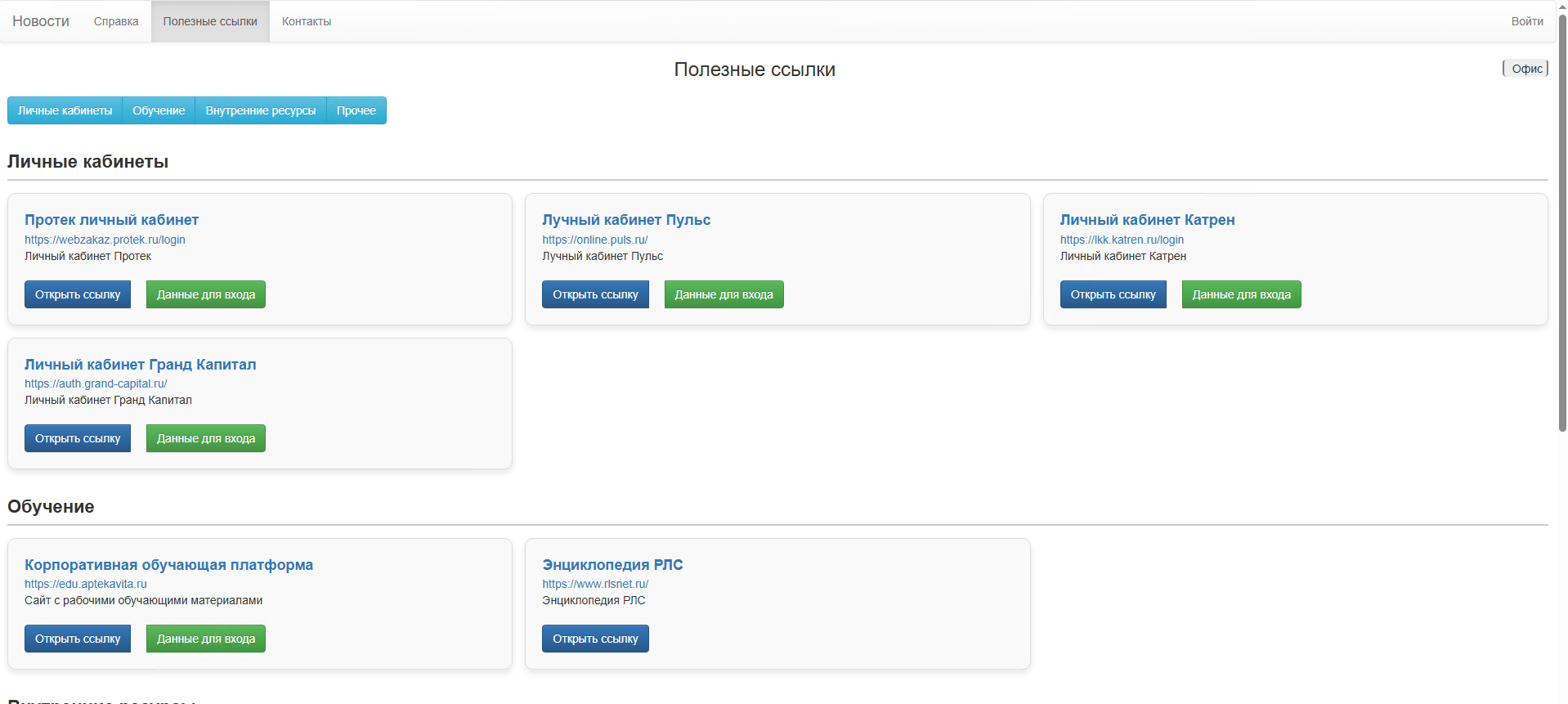


Рисунок 63 Страница “Полезные ссылки”

В Листинге 6 представлен фрагмент HTML/PHP-кода, отвечающий за отображение списка полезных ссылок, сгруппированных по категориям, с функционалом перехода к данным для входа и навигации по категориям.

Листинг 6 – Фрагмент контента страницы “Полезные ссылки”

<div class="cont">  
 <h3>Полезные ссылки</h3>  
 <!-- Кнопки-якоря -->  
 <div class="d-flex justify-content-between align-items-center flex-wrap" style="margin-bottom: 10px;">  
 <div class="btn-group">  
 <?php foreach ($categories as $categoryId => $categoryName): ?>  
 <?php if (!empty($linksByCategory[$categoryId])): ?>  
 <a class="btn btn-info" href="#category-<?= $categoryId ?>"><?= htmlspecialchars($categoryName) ?></a>  
 <?php endif; ?>  
 <?php endforeach; ?>  
 </div>  
 <?php if(isset($\_SESSION["user\_id"])): ?>  
 <div class="d-flex">  
 <div class="btn-group">  
 <a href="alllinks.php" style="margin-right: 10px;" class="btn btn-success add-btn">Ссылки аптек</a>  
 </div>  
 </div>  
 <?php endif ?>  
 </div>  
 <?php foreach ($categories as $categoryId => $categoryName): ?>  
 <?php if (!empty($linksByCategory[$categoryId])): ?>  
 <div id="category-<?= $categoryId ?>" class="category-title"><?= htmlspecialchars($categoryName) ?></div>  
 <div class="links-grid">  
 <?php foreach ($linksByCategory[$categoryId] as $one): ?>  
 <div class="link-card">  
 <div class="link-title">  
 <a target="\_blank" href="<?= htmlspecialchars($one['url']) ?>"><?= htmlspecialchars($one['name']) ?></a>  
 </div>  
 <div class="link-url">  
 <a target="\_blank" href="<?= htmlspecialchars($one['url']) ?>"><?= htmlspecialchars($one['url']) ?></a>  
 </div>  
 <p><?= htmlspecialchars($one['description']) ?></p>  
 <div class="btn-group">  
 <a target="\_blank" href="<?= htmlspecialchars($one['url']) ?>" style="margin-right: 20px;" class="btn btn-primary">Открыть ссылку</a>  
 <?php if ($one['can\_employees\_add']): ?>  
 <a href="linkpas.php?id=<?= $one['id'] ?><?php if($ispartner){ echo "&partnerid=" . $partner;}?>" class="btn btn-success">Данные для входа</a>  
 <?php endif; ?>  
 </div>  
 </div>  
 <?php endforeach; ?>  
 </div>  
 <?php endif; ?>  
 <?php endforeach; ?>  
</div>

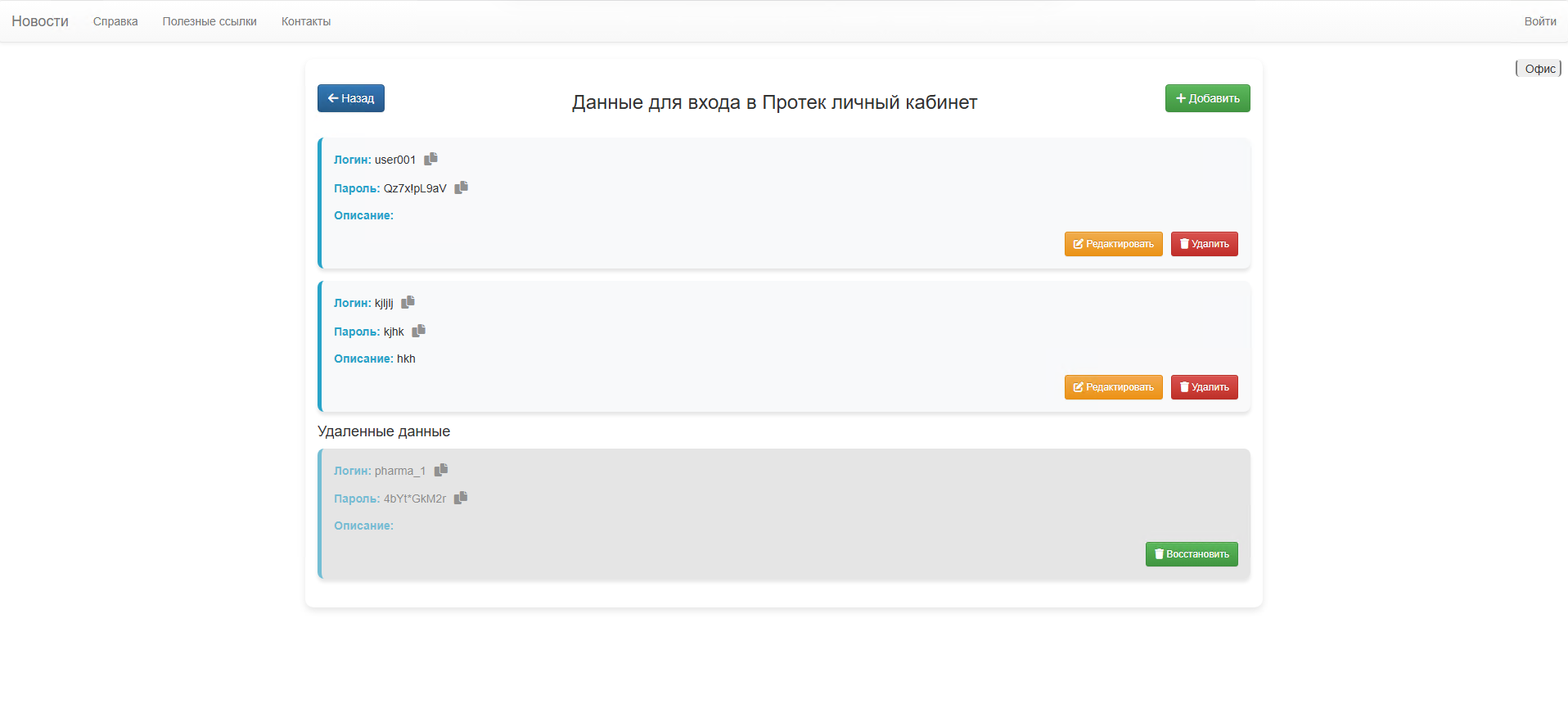


Рисунок 64 Страница с данными для входа

На странице с данными для входа отображаются все сохранённые логины и пароли, связанные с выбранной ссылкой. Пользователь может скопировать данные, отредактировать их, удалить или восстановить. Также доступна кнопка для добавления новых записей. Удалённые данные отображаются отдельно, с возможностью восстановления. Листинг 7 демонстрирует реализацию этой страницы и её клиентской логики.

Листинг 7 – Основной блок страницы с данными для входа

<div class="container">  
 <div class="header-container">  
 <a href="indexlinks.php<?php if($ispartner){ echo "?partnerid=" . $partner;} ?>" class="btn btn-primary">  
 <i class="fas fa-arrow-left"></i> Назад  
 </a>  
 <h3>Данные для входа в <?= htmlspecialchars($linkname) ?></h3>  
 <a href="formlinkpas.php?link=<?php echo $\_GET['id']; ?><?php if($ispartner){ echo "&partnerid=" . $partner;} ?>" class="btn btn-success">  
 <i class="fas fa-plus"></i> Добавить  
 </a>  
 </div>  
 <?php foreach ($linkpas as $one): ?>  
 <div class="password-card">  
 <p><strong>Логин:</strong> <?= htmlspecialchars($one['login']) ?>  
 <button class="copy-btn" onclick="copyText('login-<?= $one['id'] ?>')">  
 <i class="fas fa-copy"></i>  
 </button>  
 </p>  
 <p id="login-<?= $one['id'] ?>" style="display:none;"><?= htmlspecialchars($one['login']) ?></p>  
 <p><strong>Пароль:</strong> <?= htmlspecialchars($one['password']) ?>  
 <button class="copy-btn" onclick="copyText('password-<?= $one['id'] ?>')">  
 <i class="fas fa-copy"></i>  
 </button>  
 </p>  
 <p id="password-<?= $one['id'] ?>" style="display:none;"><?= htmlspecialchars($one['password']) ?></p>  
 <p><strong>Описание:</strong> <?= htmlspecialchars($one['description']) ?></p>  
 <div class="password-actions">  
 <a href="formlinkpas.php?link=<?= $\_GET['id'] ?>&id=<?= $one['id'] ?><?php if($ispartner){ echo "&partnerid=" . $partner;} ?>" class="btn btn-warning btn-sm">  
 <i class="fas fa-edit"></i> Редактировать  
 </a>  
 <button onclick="deletelinkpass(<?= $one['id']?>, <?=intval($\_GET['id'])?>)" class="btn btn-danger btn-sm">  
 <i class="fas fa-trash"></i> Удалить  
 </button>  
 </div>  
 </div>  
 <?php endforeach; ?>  
 <?php if (empty($linkpas) && empty($linkpas\_deleted)): ?>  
 <p class="text-center" style="font-size: 16px">Нет данных для входа</p>  
 <?php endif; ?>  
 <?php if (!(empty($linkpas) && empty($linkpas\_deleted))): ?>  
 <h4>Удаленные данные</h4>  
 <?php endif; ?>  
 <?php foreach ($linkpas\_deleted as $one): ?>  
 <div class="password-card deleted">  
 <p><strong>Логин:</strong> <?= htmlspecialchars($one['login']) ?>  
 <button class="copy-btn" onclick="copyText('login-<?= $one['id'] ?>')">  
 <i class="fas fa-copy"></i>  
 </button>  
 </p>  
 <p id="login-<?= $one['id'] ?>" style="display:none;"><?= htmlspecialchars($one['login']) ?></p>  
 <p><strong>Пароль:</strong> <?= htmlspecialchars($one['password']) ?>  
 <button class="copy-btn" onclick="copyText('password-<?= $one['id'] ?>')">  
 <i class="fas fa-copy"></i>  
 </button>  
 </p>  
 <p id="password-<?= $one['id'] ?>" style="display:none;"><?= htmlspecialchars($one['password']) ?></p>  
 <p><strong>Описание:</strong> <?= htmlspecialchars($one['description']) ?></p>  
 <div class="password-actions">  
 <button onclick="returnlinkpass(<?= $one['id']?>, <?=intval($\_GET['id'])?>)" class="btn btn-success btn-sm">  
 <i class="fas fa-trash"></i> Восстановить  
 </button>  
 </div>  
 </div>  
 <?php endforeach; ?>  
</div>  
<script src="/js/jquery-3.2.1.js"></script>  
<script src="/js/bootstrap.min.js"></script>  
<script>  
 function deletelinkpass(id, linkid) {  
 if (confirm("Вы уверены, что хотите удалить эти данные для входа?")) {  
 let partnerid = new URLSearchParams(*window*.*location*.search).get('partnerid');  
 let partnerParam = partnerid ? "&partnerid=" + partnerid : "";  
 *window*.*location*.href = "?id=" + linkid + "&delete\_id=" + id + partnerParam;  
 }  
 }  
 function returnlinkpass(id, linkid) {  
 if (confirm("Вы уверены, что хотите вернуть эти данные для входа?")) {  
 let partnerid = new URLSearchParams(*window*.*location*.search).get('partnerid');  
 let partnerParam = partnerid ? "&partnerid=" + partnerid : "";  
 *window*.*location*.href = "?id=" + linkid + "&return\_id=" + id + partnerParam;  
 }  
 }  
 function copyText(elementId) {  
 var textElement = *document*.getElementById(elementId);  
 var tempInput = *document*.createElement("input");  
 *document*.body.appendChild(tempInput);  
 tempInput.value = textElement.textContent;  
 tempInput.select();  
 *document*.execCommand("copy");  
 *document*.body.removeChild(tempInput);  
 }  
</script>

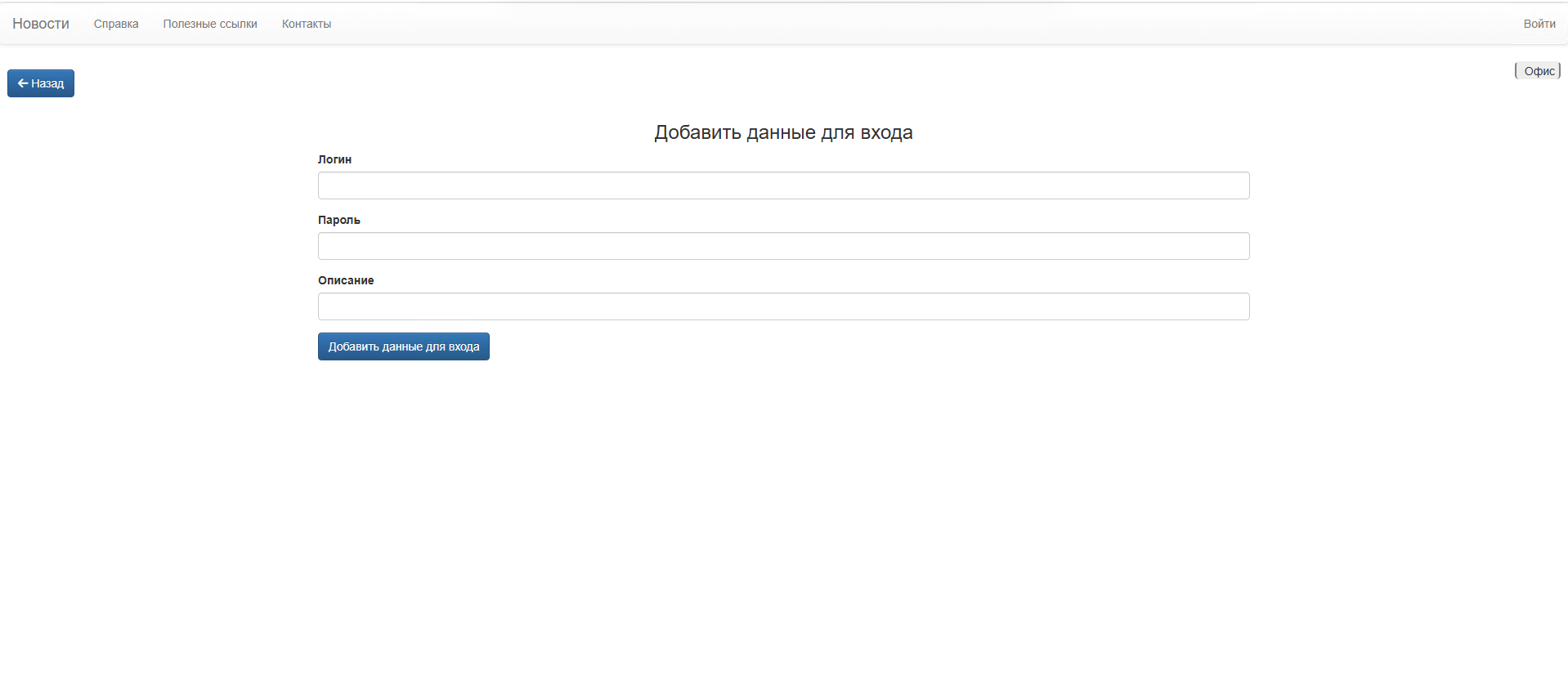


Рисунок 65 Страница с добавлением данных для входа

Если в навигационном меню выбрать пункт "Контакты", откроется одноимённая страница с перечнем контактов, которые могут понадобиться в процессе работы (см. Рисунок 66). На странице реализованы кнопки-якоря для быстрого перехода к нужной категории контактов.

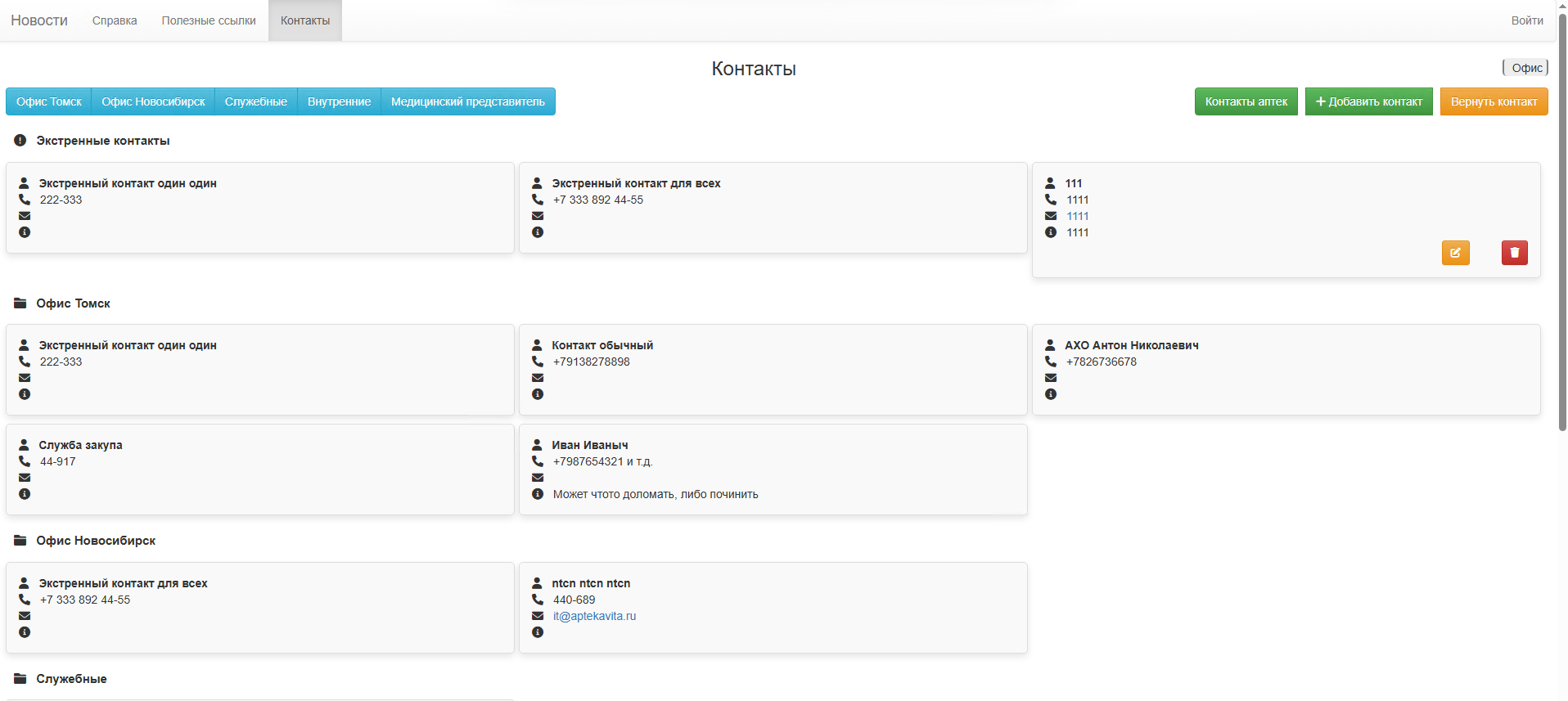


Рисунок 66 Страница “Контакты”

При удалении контакт не удаляется окончательно, а временно скрывается из общего списка. Для просмотра таких записей необходимо нажать кнопку "Вернуть контакты" — откроется страница, где можно либо окончательно удалить, либо восстановить случайно удалённый контакт (см. Рисунок 67).

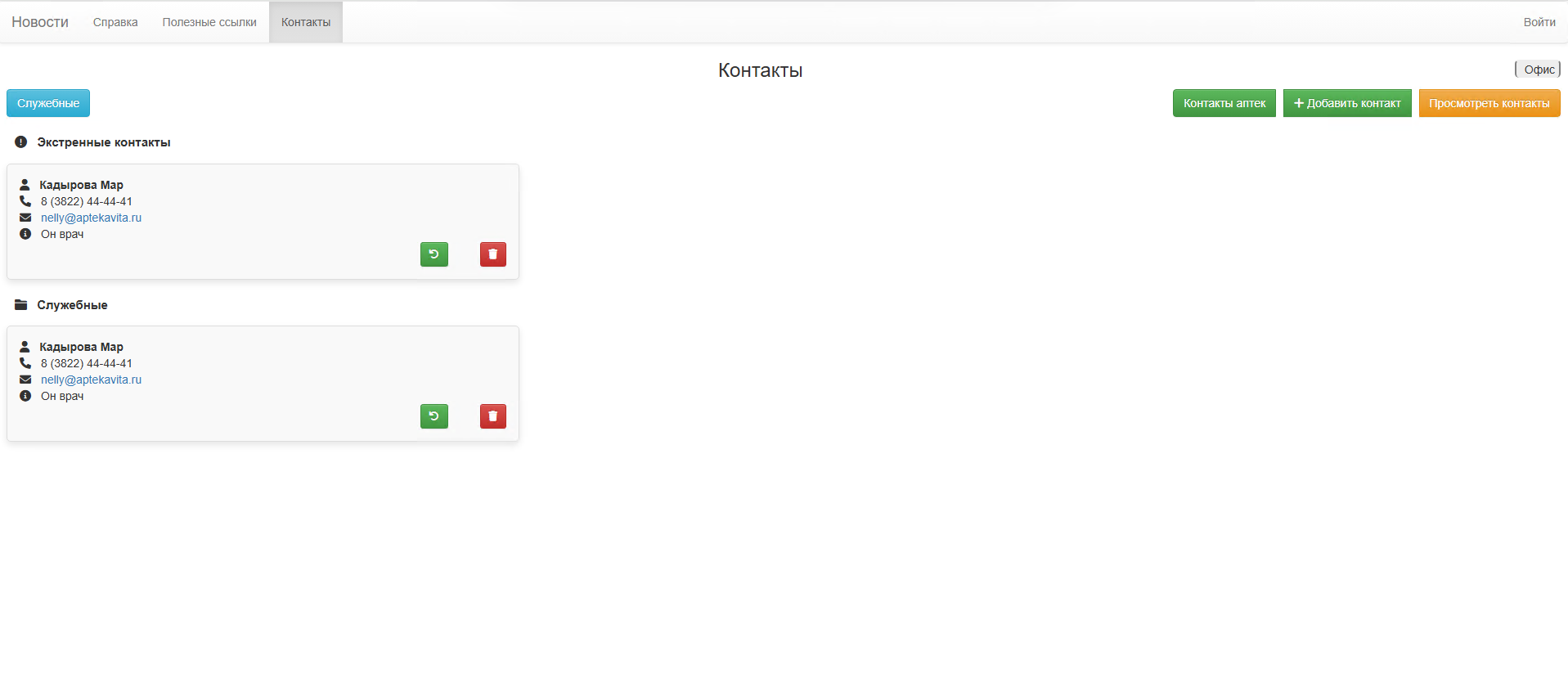


Рисунок 67 Страница “Контакты” возвращение контакта

На странице "Контакты" отображаются все добавленные контакты, сгруппированные по категориям. Реализована отдельная секция для экстренных контактов, которая отображается только при наличии соответствующих записей. Пользователю доступна навигация по категориям с помощью кнопок-якорей, а также функции редактирования, удаления, возврата и окончательного удаления контактов, реализован режим отображения удалённых контактов с возможностью их восстановления. Часть кода представлена на Листинг 8

Листинг 8 – Блок кода страницы “Контакты”

<div class="cont">  
 <h3>Контакты <?php if(isset($\_GET['partnerid'])){  
 echo $partnerName;  
 }?></h3>  
 <div class="d-flex justify-content-between align-items-center flex-wrap" style="margin-bottom: 10px;">  
 <div class="btn-group">  
 <?php foreach ($\_cat\_kont as $id => $category): ?>  
 <?php   
 $hasContacts = false;  
 foreach ($regularContacts as $one) {  
 if ($one['cat\_kont'] == $id) {  
 $hasContacts = true;  
 break;  
 }  
 }  
 if ($hasContacts): ?>  
 <button class="btn btn-info" onclick="scrollToCategory('category-<?= $id ?>')">  
 <?= htmlspecialchars($category['name']) ?>  
 </button>  
 <?php endif; ?>  
 <?php endforeach; ?>  
 </div>  
 <div class="d-flex">  
 <div class="btn-group">  
 <a href="allkontakts.php" style="margin-right: 10px;" class="btn btn-success add-btn">Контакты аптек</a>  
 <a href="formkontakts.php<?php if ($ispartner) { echo "?partner=" . $\_GET['partnerid']; } ?>" style="margin-right: 10px;" class="btn btn-success add-btn">  
 <i class="fas fa-plus" ></i> Добавить контакт  
 </a>  
 <?php if ($isReturn): ?>  
 <a href="kontakts.php<?= $retlinkV ?>" class="btn btn-warning">Просмотреть контакты</a>  
 <?php else: ?>  
 <a href="kontakts.php?ret=1<?= $retlink ?>" class="btn btn-warning">Вернуть контакт</a>  
 <?php endif; ?>  
 </div>  
 </div>  
 </div>  
 <?php if (count($urgentContacts) > 0): ?>  
 <div class="category-header">  
 <div class="category-name"><i class="fas fa-exclamation-circle"></i> Экстренные контакты</div>  
 </div>  
 <div class="contact-columns">  
 <?php foreach ($urgentContacts as $one): ?>  
 <div class="contact-column">  
 <div class="contact-item">  
 <div class="contact-info">  
 <strong><i class="fas fa-user"></i> <?= htmlspecialchars($one['name\_kont']) ?></strong><br>  
 <i class="fas fa-phone"></i> <?= htmlspecialchars($one['text\_kont']) ?><br>  
 <i class="fas fa-envelope"></i> <a href="mailto:<?= htmlspecialchars($one['email\_kont']) ?>"><?= htmlspecialchars($one['email\_kont']) ?></a><br>  
 <i class="fas fa-info-circle"></i> <?= htmlspecialchars($one['opis\_kont']) ?>  
 </div>  
 <div class="contact-actions text-right">  
 <?php if ($\_cat\_kont[$one['cat\_kont']]['can\_add'] && !$isReturn): ?>  
 <a href="formkontakts.php?cat\_kont=<?= $one['cat\_kont'] ?>&id=<?= $one['id\_kont'] ?><?php if ($ispartner) { echo "&partner=" . $\_GET['partnerid']; } ?>" class="btn btn-warning btn-sm" style="margin-right: 30px;">  
 <i class="fas fa-edit"></i>  
 </a>  
 <button onclick="hideContact(<?= $one['id\_kont'] ?>)" class="btn btn-danger btn-sm">  
 <i class="fas fa-trash"></i>  
 </button>  
 <?php elseif ($isReturn): ?>  
 <button onclick="returnContact(<?= $one['id\_kont'] ?>)" style="margin-right: 30px;" class="btn btn-success btn-sm">  
 <i class="fas fa-undo"></i>  
 </button>  
 <button onclick="deleteContact(<?= $one['id\_kont'] ?>)" class="btn btn-danger btn-sm">  
 <i class="fas fa-trash"></i>  
 </button>  
 <?php endif; ?>  
 </div>  
 </div>  
 </div>  
 <?php endforeach; ?>  
 </div>  
 <?php endif; ?>  
 <?php   
 $currentCategory = null;  
 foreach ($regularContacts as $one):   
 // Если категория изменилась, выводим заголовок категории  
 if ($one['cat\_kont'] != $currentCategory):   
 // Закрываем предыдущий контейнер контактов  
 if ($currentCategory !== null) {  
 echo '</div>';  
 }  
 $currentCategory = $one['cat\_kont'];  
 ?>  
 <div id="category-<?= $one['cat\_kont'] ?>" class="category-header">  
 <div class="category-name">  
 <i class="fas fa-folder"></i> <?= $\_cat\_kont[$currentCategory]['name'] ?>  
 </div>  
 </div>  
 <div class="contact-columns">  
 <?php endif; ?>  
 <div class="contact-column">  
 <div class="contact-item">  
 <div class="contact-info">  
 <strong><i class="fas fa-user"></i> <?= htmlspecialchars($one['name\_kont']) ?></strong><br>  
 <i class="fas fa-phone"></i> <?= htmlspecialchars($one['text\_kont']) ?><br>  
 <i class="fas fa-envelope"></i> <a href="mailto:<?= htmlspecialchars($one['email\_kont']) ?>"><?= htmlspecialchars($one['email\_kont']) ?></a><br>  
 <i class="fas fa-info-circle"></i> <?= htmlspecialchars($one['opis\_kont']) ?>  
 </div>  
 <div class="contact-actions text-right">  
 <?php if ($\_cat\_kont[$one['cat\_kont']]['can\_add'] && !$isReturn): ?>  
 <a href="formkontakts.php?cat\_kont=<?= $one['cat\_kont'] ?>&id=<?= $one['id\_kont'] ?><?php if ($ispartner) { echo "&partner=" . $\_GET['partnerid']; } ?>" class="btn btn-warning btn-sm" style="margin-right: 20px;">  
 <i class="fas fa-edit"></i>  
 </a>  
 <button onclick="hideContact(<?= $one['id\_kont'] ?>)" class="btn btn-danger btn-sm">  
 <i class="fas fa-trash"></i>  
 </button>  
 <?php elseif ($isReturn): ?>  
 <button onclick="returnContact(<?= $one['id\_kont'] ?>)" style="margin-right: 30px;" class="btn btn-success btn-sm">  
 <i class="fas fa-undo"></i>  
 </button>  
 <button onclick="deleteContact(<?= $one['id\_kont'] ?>)" class="btn btn-danger btn-sm">  
 <i class="fas fa-trash"></i>  
 </button>  
 <?php endif; ?>  
 </div>  
 </div>  
 </div>  
 <?php endforeach; ?>  
 </div>  
</div>

Для добавления нового контакта предусмотрена кнопка "Добавить контакт", которая открывает форму создания (см. Рисунок 68).

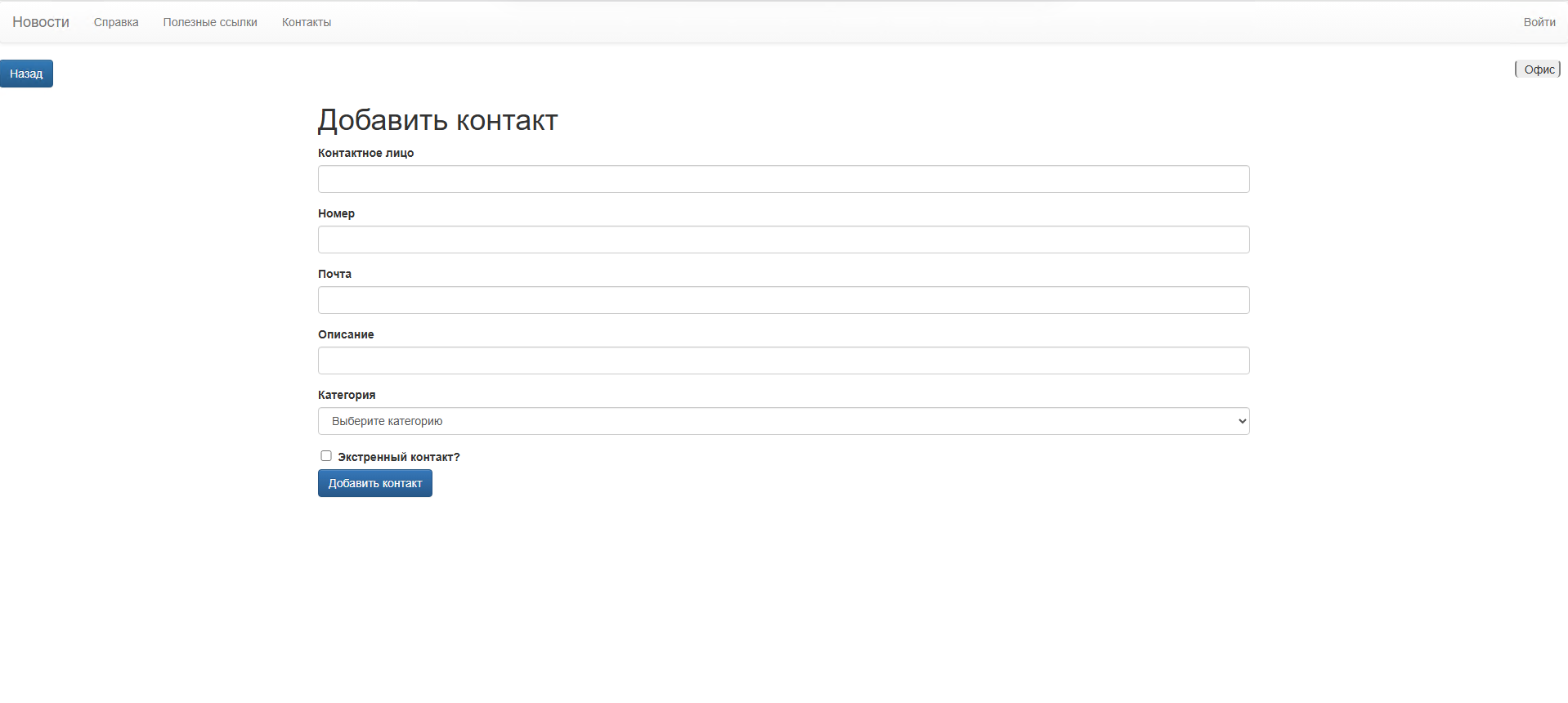


Рисунок 68 Страница добавления контактов

Также есть возможность просмотреть информацию об остальных аптеках нажав на кнопку “Контакты аптек” (см. Рисунок 69).

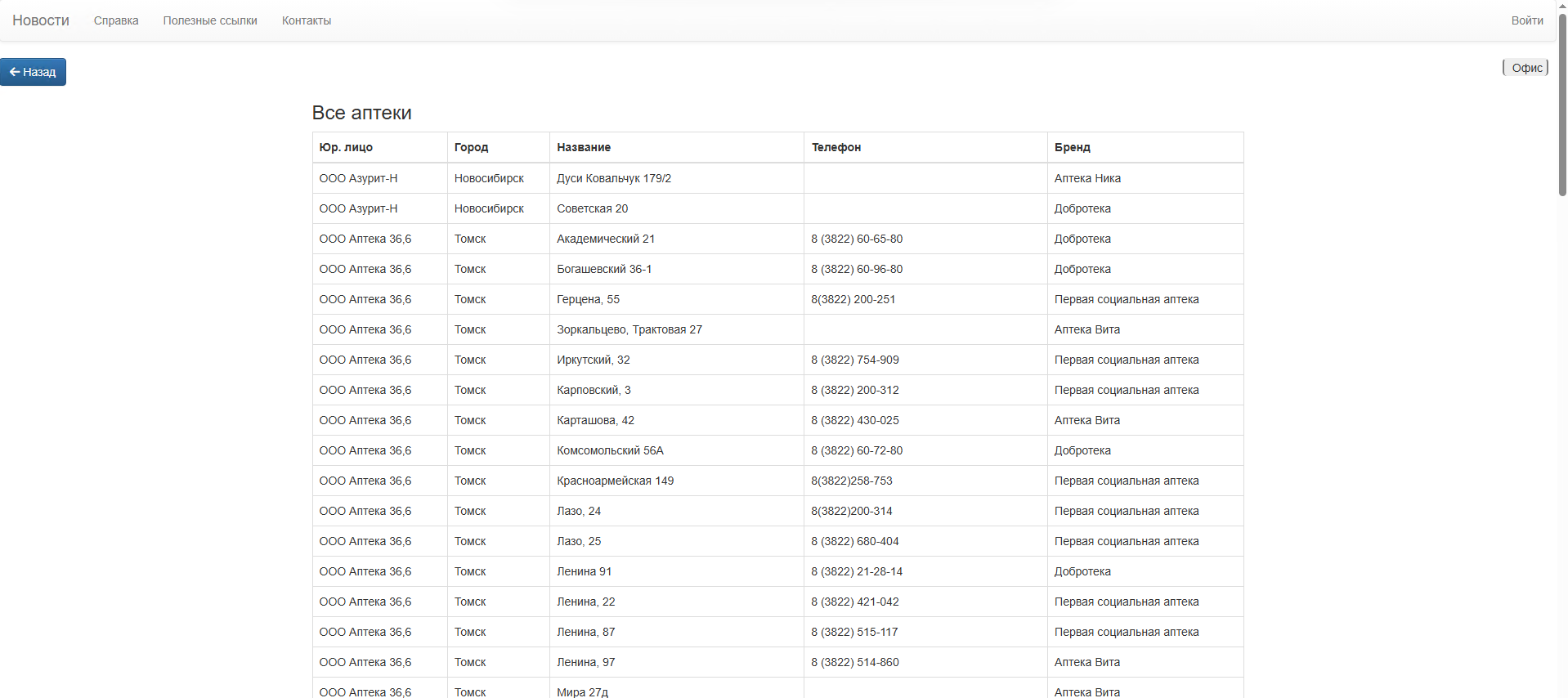


Рисунок 69 Страница просмотра всех аптек

## Административная часть

При заходе на страницы, защищенные авторизацией если пользователь не авторизован его перебрасывает на страницу для авторизации (см. Рисунок 70).

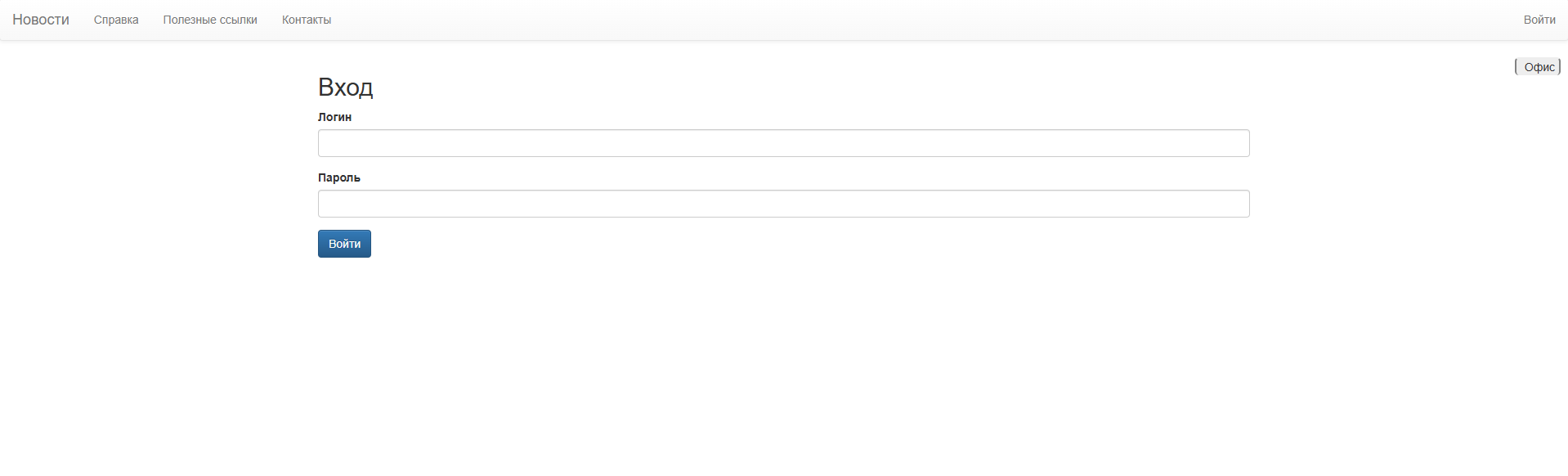


Рисунок 70 Страница “Вход”

На странице “Вход” реализован механизм авторизации пользователя с проверкой логина и пароля в базе данных. В случае успешной аутентификации пользователь перенаправляется на страницу, с которой был направлен к авторизации, либо на главную страницу по умолчанию.

Предусмотрена базовая обработка ошибок, а также визуальный вывод сообщений о неверных данных. Также осуществляется предотвращение повторного входа уже авторизованного пользователя.

Листинг 9 демонстрирует полный код страницы “Вход”.

Листинг 9 – Код страницы “Вход”

<?php  
require\_once "bd\_connect1.php";  
require "allfunc.php";  
// Если пользователь уже авторизован — редирект на сохранённую страницу  
if (isset($\_SESSION['user\_id'])) {  
 header("Location: " . ($\_SESSION['previous\_page'] ?? 'index.php'));  
 exit;  
}  
// Сохраняем предыдущую страницу  
if (!isset($\_SESSION['previous\_page']) && !empty($\_SERVER['HTTP\_REFERER']) && strpos($\_SERVER['HTTP\_REFERER'], 'login.php') === false) {  
 $\_SESSION['previous\_page'] = $\_SERVER['HTTP\_REFERER'];  
} elseif (!isset($\_SESSION['previous\_page']) && strpos($\_SERVER['REQUEST\_URI'], 'login.php') === false) {  
 $\_SESSION['previous\_page'] = $\_SERVER['REQUEST\_URI'];  
}  
$connect = mysqli\_connect(*HOST*, *USER*, *PW*, *DB*);  
mysqli\_set\_charset($connect, "UTF8");  
if ($\_SERVER["REQUEST\_METHOD"] == "POST") {  
 $login = trim($\_POST["login"]);  
 $password = trim($\_POST["password"]);  
 if (!empty($login) && !empty($password)) {  
 $sql = "SELECT id, login, name, password FROM users WHERE login = ? AND password = ?";  
 if ($stmt = mysqli\_prepare($connect, $sql)) {  
 mysqli\_stmt\_bind\_param($stmt, "ss", $login, $password);  
 if (mysqli\_stmt\_execute($stmt)) {  
 mysqli\_stmt\_store\_result($stmt);  
 if (mysqli\_stmt\_num\_rows($stmt) == 1) {  
 mysqli\_stmt\_bind\_result($stmt, $id, $login, $name, $password);  
 if (mysqli\_stmt\_fetch($stmt)) {  
 $\_SESSION["user\_id"] = $id;  
 $\_SESSION["login"] = $login;  
 $\_SESSION["name"] = $name;  
 // Проверяем, есть ли сохранённая страница  
 $redirectPage = $\_SESSION['previous\_page'] ?? 'index.php';  
 unset($\_SESSION['previous\_page']); // Удаляем, чтобы не зациклить  
 header("Location: " . $redirectPage);  
 exit;  
 } else {  
 $login\_err = "Неправильный логин или пароль";  
 }  
 } else {  
 $login\_err = "Неправильный логин или пароль";  
 }  
 } else {  
 echo "Что-то пошло не так. Пожалуйста, повторите попытку позже.";  
 }  
 mysqli\_stmt\_close($stmt);  
 }  
 } else {  
 $login\_err = "Пожалуйста, введите логин и пароль";  
 }  
}  
?>  
<!DOCTYPE html>  
<html lang="ru">  
<head>  
 <meta charset="UTF-8">  
 <meta name="viewport" content="width=device-width, initial-scale=1.0">  
 <title>Вход</title>  
 <link rel="stylesheet" href="/css/bootstrap.min.css">  
</head>  
<body>  
 <?php  
 $navbarType = "user";   
 include 'topnavbar.php';  
 ?>  
 <div class="container">  
 <h2>Вход</h2>  
 <form action="<?php echo htmlspecialchars($\_SERVER["PHP\_SELF"]); ?>" method="post">  
 <div class="form-group">  
 <label>Логин</label>  
 <input type="text" name="login" class="form-control" required>  
 </div>  
 <div class="form-group">  
 <label>Пароль</label>  
 <input type="password" name="password" class="form-control" required>  
 </div>  
 <div class="form-group">  
 <input type="submit" class="btn btn-primary" value="Войти">  
 </div>  
 <?php  
 if (!empty($login\_err)) {  
 echo '<div class="alert alert-danger">' . $login\_err . '</div>';  
 }  
 ?>  
 </form>  
 </div>  
</body>  
</html>

Страница "Прайс-листы" предназначена для управления прайс-листами и функционально не изменялась. Были изменены дизайн кнопок и оформление таблицы (см. Рисунок 71). На этой странице можно добавить аптеку, нажав кнопку "Добавить" — откроется страница добавления аптеки (см. Рисунок 72).

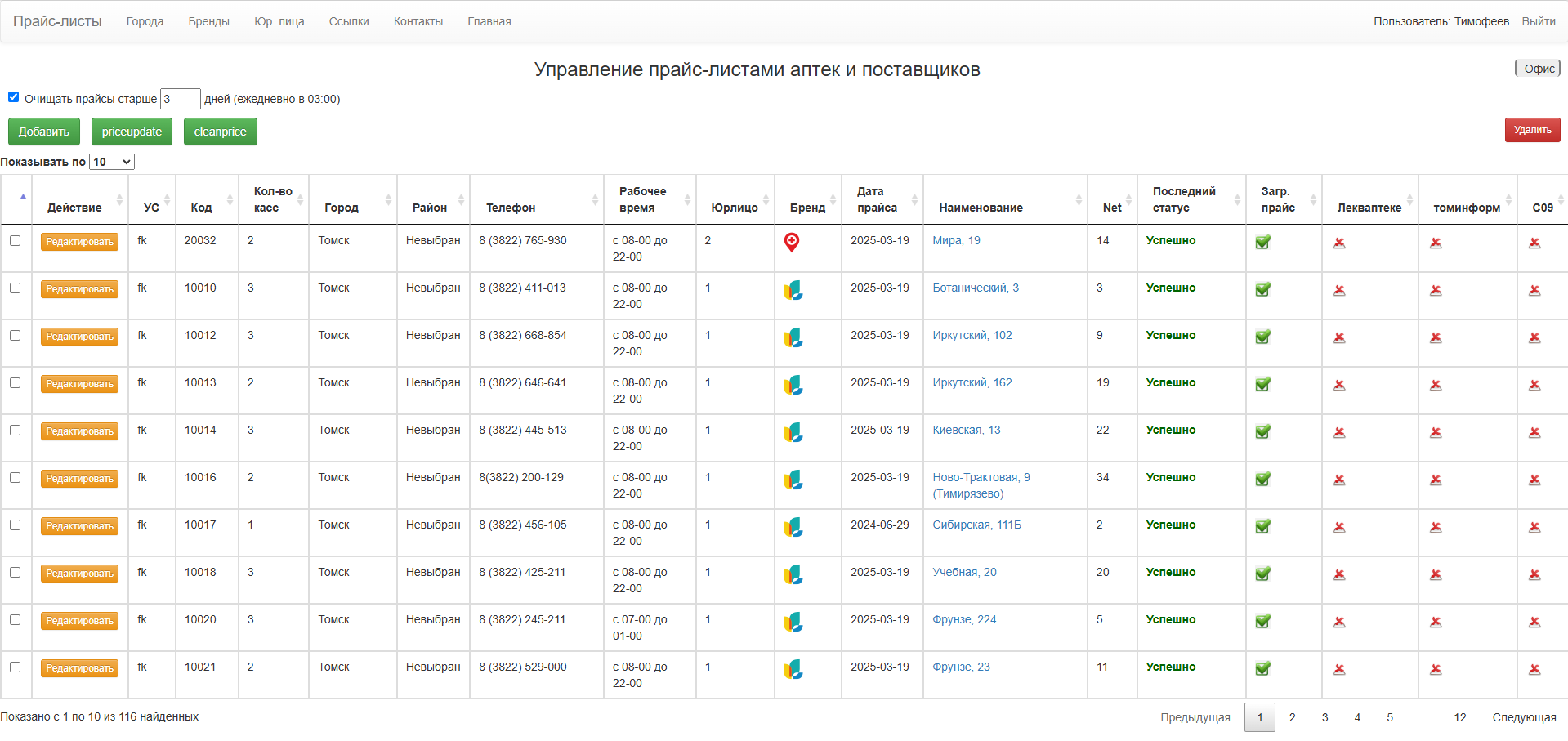


Рисунок 71 Страница “Прайс-листы”

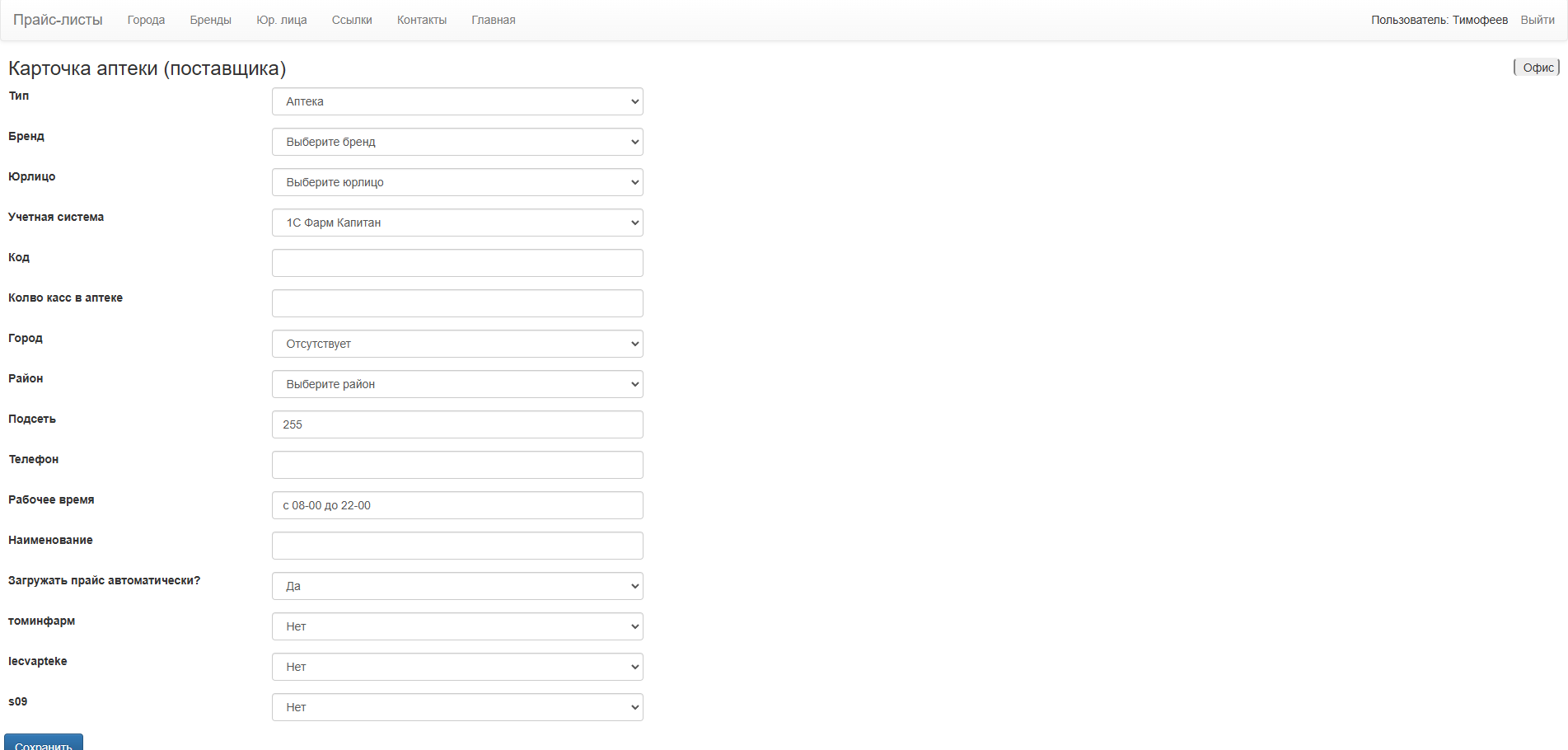


Рисунок 72 Страница создания аптеки

Страницы "Города", "Бренды" и "Юр. лица" выполнены по одному принципу. В качестве примера выбрана страница "Города" (см. Рисунок 73).

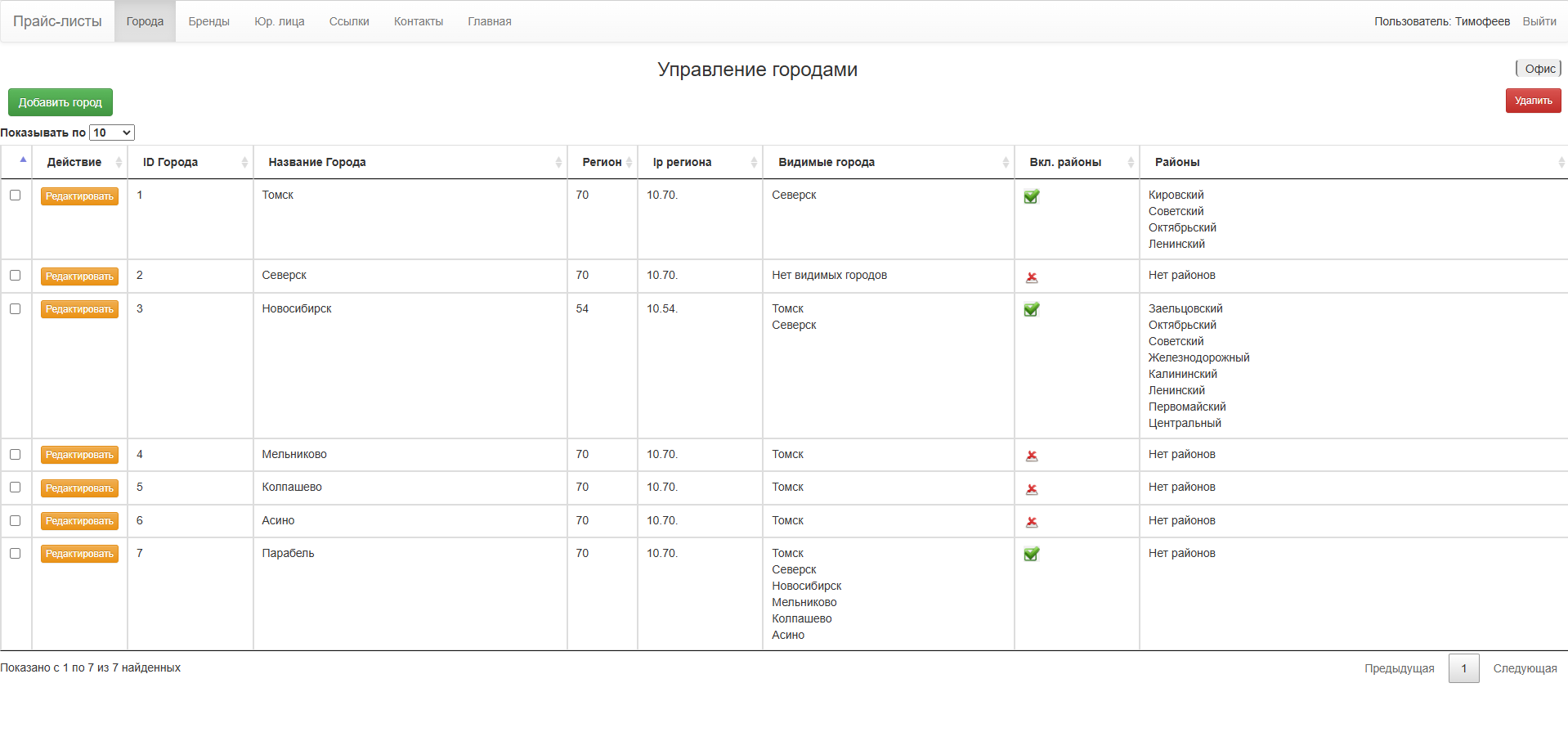


Рисунок 73 Страница “Города”

На странице “Города” реализовано управление списком городов: отображение, добавление, редактирование и удаление вместе с районами. Используется таблица с сортировкой и сохранением состояния. Доступ ограничен авторизованными пользователями.

Листинг 10 – Код страницы “Города”

<?php  
 // Подключение файла с настройками подключения к базе данных  
 require\_once "bd\_connect1.php";  
 // Подключение файла с дополнительными функциями  
 require "allfunc.php";  
 checkAuth();  
 // Установка соединения с базой данных  
 $connect = mysqli\_connect(*HOST*, *USER*, *PW*, *DB*);  
 // Установка кодировки соединения  
 mysqli\_set\_charset($connect, "UTF8");  
 // SQL-запрос для выборки городов и связанных с ними районов и видимых городов  
 $query = "SELECT c.\*, GROUP\_CONCAT(d.name SEPARATOR '<br/> ') AS district\_names,  
 (SELECT GROUP\_CONCAT(vc.namecity SEPARATOR '<br/> ')   
 FROM cities vc WHERE FIND\_IN\_SET(vc.idcity, c.visiblecities)) AS visible\_city\_names  
 FROM cities c  
 LEFT JOIN district d ON c.idcity = d.cityid  
 GROUP BY c.idcity";  
 // Выполнение запроса и получение результата  
 $result = mysqli\_query($connect, $query);  
 // Получение количества строк в результате запроса  
 $num\_rowscity = mysqli\_num\_rows($result);  
 // Инициализация массива для хранения городов  
 $cities = array();  
 // Цикл для получения данных из результата запроса и добавления их в массив $cities  
 while ($row = mysqli\_fetch\_assoc($result)) {  
 $cities[] = $row;  
 }  
 // Если есть города, то дополнительно получаем данные из результата запроса и добавляем их в массив $cities  
 if ($num\_rowscity > 0) {  
 while ($rescity = mysqli\_fetch\_assoc($result)) {  
 if ($rescity === false) {  
 die("Ошибка при получении данных: " . mysqli\_error($connect));  
 }  
 $cities[] = $rescity;   
 }  
 }  
 // Освобождение памяти, занятой результатом запроса  
 mysqli\_free\_result($result);  
 // Проверка, была ли отправлена форма с параметром submit  
 if (isset($\_GET['submit'])) {  
 // Проверка, был ли отправлен параметр modify  
 if (isset($\_GET['modify'])){  
 // Получение идентификаторов городов для удаления  
 $cities\_for\_delete = implode(",", $\_GET['modify']);  
 // Установка соединения с базой данных  
 $connect = mysqli\_connect(*HOST*, *USER*, *PW*, *DB*);  
 mysqli\_set\_charset($connect, "utf8");  
 // Проверка, была ли нажата кнопка "Удалить"  
 if ($\_GET['submit'] == 'Удалить') {  
 // SQL-запрос для удаления районов, связанных с выбранными городами  
 $delete\_districts\_query = "DELETE FROM district WHERE cityid IN ($cities\_for\_delete)";  
 mysqli\_query($connect, $delete\_districts\_query);  
 // SQL-запрос для удаления выбранных городов  
 $delete\_cities\_query = "DELETE FROM cities WHERE idcity IN ($cities\_for\_delete)";  
 mysqli\_query($connect, $delete\_cities\_query);  
 // Перенаправление на страницу управления городами  
 header("Location:/".*CITIESPHP*);  
 exit;  
 }  
 }  
 }  
 // Закрытие соединения с базой данных  
 mysqli\_close($connect);  
?>  
<!DOCTYPE html>  
<html lang="ru">  
<head>  
 <meta charset="UTF-8">  
 <meta name="viewport" content="width=device-width, initial-scale=1.0">  
 <title>Управление городами</title>  
 <link rel="stylesheet" href="/css/avstyles.css">  
 <link rel="stylesheet" href="/css/bootstrap.min.css">  
 <link rel="stylesheet" href="/css/bootstrap-theme.min.css">  
 <link rel="stylesheet" type="text/css" href="/css/jquery.dataTables.css">  
 <style>  
 .btn-default, table {  
 margin: 10px;  
 border: medium;  
 }  
 </style>  
</head>  
<body>  
<?php   
$navbarType = "admin";   
include 'topnavbar.php';  
?>  
<form>  
<h3 style="text-align: center">Управление городами</h3>  
<form id="mod" method="GET" action="cities.php">  
 <input type="submit" style="margin-left: 10px; margin-bottom: 10px;" class="btn btn-success add-btn" name="submit" value="Добавить город" formaction="formcity.php">  
 <input type="submit" class="btn btn-sm btn-danger p-4" name="submit" value="Удалить" style="float: right; margin-right: 10px; margin-bottom: 10px;">  
</form>  
<table id="cities" class="table table-bordered table-condensed">  
 <thead>  
 <tr>  
 <th style="width: 0;"></th>  
 <th style="width: 0;" >Действие</th>  
 <th style="width: 8%;" >ID Города</th>  
 <th style="width: 20%;">Название Города</th>  
 <th style="width: 0;">Регион</th>  
 <th style="width: 8%;">Ip региона</th>  
 <th style="width: 16%;">Видимые города</th>  
 <th style="width: 8%;">Вкл. районы</th>  
 <th>Районы</th>  
 </tr>  
 </thead>  
 <tbody>  
 <?php  
 foreach ($cities as $city) {  
 echo "<tr>";  
 echo '<td><input type="checkbox" name="modify[]" value="' . $city['idcity'] . '" form="mod"></td>';  
 echo '<td><a href="formcity.php?idcity=' . $city["idcity"] . '" class="btn btn-warning btn-xs">Редактировать</td>';  
 echo '<td>' . $city['idcity'] . '</td>';  
 echo "<td>" . $city['namecity'] . "</td>";  
 echo "<td>" . $city['regioncity'] . "</td>";  
 echo "<td>" . $city['ipcity'] . "</td>";  
 echo "<td>" . (empty($city['visible\_city\_names']) ? 'Нет видимых городов' : $city['visible\_city\_names']) . "</td>";  
 if ($city['isenabled']==1)  
 {  
 $val = "<img src='images/da.jpg' width=20 height=20>";  
 }  
 else  
 {  
 $val = "<img src='images/net.jpg' width=20 height=20>";  
 }  
 echo "<td>$val</td>";   
 echo "<td>" . (empty($city['district\_names']) ? 'Нет районов' : $city['district\_names']) . "</td>";  
 echo "</tr>";  
 }  
 ?>  
 </tbody>  
</table>  
<div class="idid"></div>  
<script src="/js/jquery-3.2.1.js"></script>  
<script src="/js/bootstrap.min.js"></script>  
<script type="text/javascript" charset="utf8" src="/js/jquery.dataTables.min.js"></script>  
<script>  
 $(function(){  
 $('#cities').DataTable({  
 "searching": false,  
 "stateSave": true,  
 "language":{  
 info:"Показано с \_START\_ по \_END\_ из \_TOTAL\_ найденных",  
 "paginate": {  
 "first": "Первая",  
 "last": "Последняя",  
 "next": " Следующая",  
 "previous": "Предыдущая "  
 },  
 "infoEmpty": "Записей не найдено",  
 "emptyTable": "Записей не найдено",  
 "lengthMenu": "Показывать по \_MENU\_ "  
 }  
 });  
 });  
</script>  
</body>  
</html>

На странице "Контакты" представлена таблица со всеми контактами всех аптек (см. Рисунок 74). Код страницы представлен на Листинг 11. При нажатии на кнопку "Управление категориями" таблица с контактами заменяется на таблицу с категориями и соответствующими кнопками (см. Рисунок 75). Код страницы представлен на Листинг 12

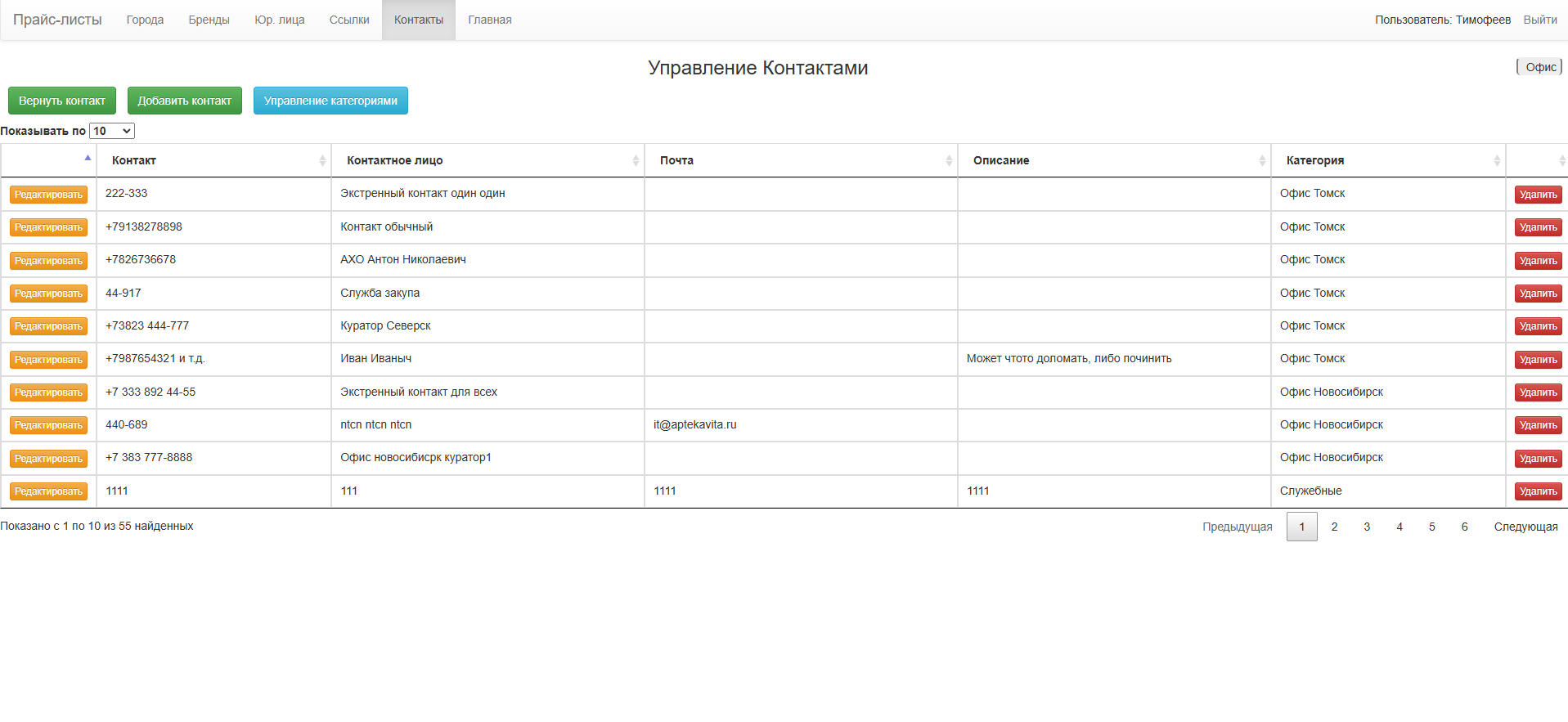


Рисунок 74 Страница “Управление контактами”

Листинг 11 – Код страницы “Управление контактами”

<?php  
 require\_once "bd\_connect1.php";  
 require "allfunc.php";  
 checkAuth();  
 $connect = mysqli\_connect(*HOST*, *USER*, *PW*, *DB*);  
 mysqli\_set\_charset($connect, "UTF8");  
 $isadmin = 0;  
 $isReturn = isset($\_GET['ret']);  
 if(isset($\_SESSION['login'])){  
 if($\_SESSION['login'] == 'admin'){  
 $isadmin = 1;  
 }  
 }  
 if($isReturn)  
 {  
 $querycity = "SELECT \* FROM `" . *KONTAKTTABLE* . "` WHERE see\_kont=1 ORDER BY cat\_kont, urgent\_kont DESC";  
 } else  
 {  
 $querycity = "SELECT \* FROM `" . *KONTAKTTABLE* . "` WHERE see\_kont=0 ORDER BY cat\_kont, urgent\_kont DESC ";  
 }  
 $resultkont = mysqli\_query($connect, $querycity);  
 if (!$resultkont) {  
 die("Ошибка запроса: " . mysqli\_error($connect));  
 }  
 $konts = [];  
 while ($rescity = mysqli\_fetch\_assoc($resultkont)) {  
 $konts[] = $rescity;  
 }  
 mysqli\_free\_result($resultkont);  
 // Получаем все категории и их флаг can\_employees\_add  
 $query\_cat\_kont = "SELECT \* FROM " . *CATKONTAKTTABLE*;  
 $result\_cat\_kont = mysqli\_query($connect, $query\_cat\_kont);  
 if (!$result\_cat\_kont) {  
 die("Ошибка запроса: " . mysqli\_error($connect));  
 }  
 $\_cat\_kont = [];  
 while ($res\_cat\_kont = mysqli\_fetch\_assoc($result\_cat\_kont)) {  
 $\_cat\_kont[$res\_cat\_kont["id\_cat\_kont"]] = [  
 "name" => $res\_cat\_kont["name\_cat\_kont"],  
 "can\_add" => (bool) $res\_cat\_kont["can\_employees\_add"]  
 ];  
 }  
 mysqli\_free\_result($result\_cat\_kont);  
 if (isset($\_GET['delete\_id'])) {  
 $delete\_id = intval($\_GET['delete\_id']);  
 $query = "UPDATE `" . *KONTAKTTABLE* . "` SET see\_kont = 1 WHERE id\_kont = $delete\_id";  
 mysqli\_query($connect, $query);  
 header("Location: /admkontakts.php");  
 exit;  
 }  
 if (isset($\_GET['return\_id'])) {  
 $return\_id = intval($\_GET['return\_id']);  
 $query = "UPDATE `" . *KONTAKTTABLE* . "` SET see\_kont = 0 WHERE id\_kont = $return\_id";  
 mysqli\_query($connect, $query);  
 header("Location: /admkontakts.php?ret=1");  
 exit;  
 }  
 mysqli\_close($connect);  
?>  
<!DOCTYPE html>  
<html lang="ru">  
<head>  
 <meta charset="UTF-8">  
 <meta name="viewport" content="width=device-width, initial-scale=1.0">  
 <title>Управление Контактами</title>  
 <link rel="stylesheet" href="/css/bootstrap.min.css">  
 <script src="/js/jquery-3.2.1.js"></script>  
 <script>  
 function deleteContact(id) {  
 if (confirm("Вы уверены, что хотите удалить этот контакт?")) {  
 *window*.*location*.href = "?delete\_id=" + id;  
 }  
 }  
 function returnContact(id) {  
 if (confirm("Вы уверены, что хотите вернуть этот контакт?")) {  
 *window*.*location*.href = "?return\_id=" + id;  
 }  
 }  
 </script>  
</head>  
<body>  
<?php   
$navbarType = "admin";   
include 'topnavbar.php';  
?>  
 <h3 style="text-align: center">Управление Контактами</h3>  
 <?php   
 if($isReturn){  
 echo '<a href="admkontakts.php" class="btn btn-success" style="margin-left: 10px; margin-bottom: 10px;">Просмотреть контакты</a>';  
 }  
 else{  
 echo '<a href="admkontakts.php?ret=1" class="btn btn-success" style="margin-left: 10px; margin-bottom: 10px;">Вернуть контакт</a>';  
 }  
 ?>  
 <a href="admformkontakts.php" class="btn btn-success" style="margin-left: 10px; margin-bottom: 10px;">Добавить контакт</a>  
 <a href="admcatkont.php" style="margin-left: 10px; margin-bottom: 10px;" class="btn btn-info add-btn">  
 <i class="fas fa-plus" ></i> Управление категориями  
 </a>  
 <table id="konts" class="table table-bordered">  
 <thead>  
 <tr>  
 <th></th>  
 <th>Контакт</th>  
 <th>Контактное лицо</th>  
 <th>Почта</th>  
 <th>Описание</th>  
 <th>Категория</th>  
 <th></th>  
 </tr>  
 </thead>  
 <tbody>  
 <?php $cc = 0; foreach ($konts as $one) { ?>  
 <tr>  
 <?php if (!$isReturn) { ?>  
 <td><a href="admformkontakts.php?cat\_kont=<?= $one['cat\_kont'] ?>&id=<?= $one['id\_kont'] ?>" class="btn btn-warning btn-xs" >Редактировать</a></td>  
 <?php } else { ?>  
 <td><button onclick="returnContact(<?= $one['id\_kont'] ?>)" class="btn btn-danger btn-xs">Вернуть</button></td>  
 <?php } ?>  
 <td width=15%><?= htmlspecialchars($one['text\_kont']) ?></td>  
 <td width=20%><?= htmlspecialchars($one['name\_kont']) ?></td>  
 <td width=20%><?= htmlspecialchars($one['email\_kont']) ?></td>  
 <td width=20%><?= htmlspecialchars($one['opis\_kont']) ?></td>  
 <td width=15%><?= htmlspecialchars($\_cat\_kont[$one['cat\_kont']]['name']) ?></td>  
 <?php if (!$isReturn) { ?>  
 <td><button onclick="deleteContact(<?= $one['id\_kont'] ?>)" class="btn btn-danger btn-xs">Удалить</button></td>  
 <?php }?>  
 </tr>  
 <?php $cc = $one['cat\_kont']; ?>  
 <?php } ?>  
 </tbody>  
 </table>  
 <div class="idid"></div>  
 <script src="/js/jquery-3.2.1.js"></script>  
 <script src="/js/bootstrap.min.js"></script>  
 <script type="text/javascript" charset="utf8" src="/js/jquery.dataTables.min.js"></script>  
 <script>  
 $(function(){  
 $('#konts').DataTable({  
 "searching": false,  
 "stateSave": true,  
 "language":{  
 info:"Показано с \_START\_ по \_END\_ из \_TOTAL\_ найденных",  
 "paginate": {  
 "first": "Первая",  
 "last": "Последняя",  
 "next": " Следующая",  
 "previous": "Предыдущая "  
 },  
 "infoEmpty": "Записей не найдено",  
 "emptyTable": "Записей не найдено",  
 "lengthMenu": "Показывать по \_MENU\_ "  
 }  
 });  
 });  
 </script>  
</body>  
</html>



Рисунок 75 Страница “Контакты” управление категориями

Листинг 12 – Код страницы “Управление категориями контактов”

<?php   
 require\_once "bd\_connect1.php";  
 require "allfunc.php";  
 checkAuth();  
 $connect = mysqli\_connect(*HOST*, *USER*, *PW*, *DB*);  
 mysqli\_set\_charset($connect, "UTF8");  
 $isadmin = 0;  
 if(isset($\_SESSION['login'])){  
 if($\_SESSION['login'] == 'admin'){  
 $isadmin = 1;  
 }  
 }  
 $querycity = "SELECT \* FROM `" . *CATKONTAKTTABLE* . "`";  
 $resultkont = mysqli\_query($connect, $querycity);  
 if (!$resultkont) {  
 die("Ошибка запроса: " . mysqli\_error($connect));  
 }  
 $catkonts = [];  
 while ($rescity = mysqli\_fetch\_assoc($resultkont)) {  
 $catkonts[] = $rescity;  
 }  
 mysqli\_free\_result($resultkont);  
 // Получаем все категории и их флаг can\_employees\_add  
 $query\_cat\_kont = "SELECT \* FROM " . *CATKONTAKTTABLE*;  
 $result\_cat\_kont = mysqli\_query($connect, $query\_cat\_kont);  
 if (!$result\_cat\_kont) {  
 die("Ошибка запроса: " . mysqli\_error($connect));  
 }  
 mysqli\_free\_result($result\_cat\_kont);  
 if (isset($\_GET['delete\_id'])) {  
 $delete\_id = intval($\_GET['delete\_id']);  
 $query = "DELETE FROM `" . *CATKONTAKTTABLE* . "` WHERE id\_cat\_kont = $delete\_id";  
 mysqli\_query($connect, $query);  
 header("Location: /admcatkont.php");  
 exit;  
 }  
 mysqli\_close($connect);  
?>  
<!DOCTYPE html>  
<html lang="ru">  
<head>  
 <meta charset="UTF-8">  
 <meta name="viewport" content="width=device-width, initial-scale=1.0">  
 <title>Управление Контактами</title>  
 <link rel="stylesheet" href="/css/bootstrap.min.css">  
 <script src="/js/jquery-3.2.1.js"></script>  
 <script>  
 function deleteContact(id) {  
 if (confirm("Вы уверены, что хотите удалить этот контакт?")) {  
 *window*.*location*.href = "?delete\_id=" + id;  
 }  
 }  
 </script>  
</head>  
<body>  
<?php   
$navbarType = "admin";   
include 'topnavbar.php';  
?>  
 <h3 style="text-align: center">Управление Контактами</h3>  
 <a href="admformcatkont.php" class="btn btn-success" style="margin-left: 10px; margin-bottom: 10px;">Добавить категорию</a>  
 <a href="admkontakts.php" style="margin-left: 10px; margin-bottom: 10px;" class="btn btn-info add-btn">  
 <i class="fas fa-plus" ></i> Управление контактами  
 </a>  
 <table id="catkonts" class="table table-bordered">  
 <thead>  
 <tr>  
 <th></th>  
 <th>Название категории</th>  
 <th>Cтатус</th>  
 <th>Могут сотрудники добавлять</th>  
 <th>Города которые видят</th>  
 <th></th>  
 </tr>  
 </thead>  
 <tbody>  
 <?php foreach ($catkonts as $one) { ?>  
 <tr>  
 <td><a href="admformcatkont.php?id=<?= $one['id\_cat\_kont'] ?>" class="btn btn-warning btn-xs" >Редактировать</a></td>  
 <td width=15%><?= htmlspecialchars($one['name\_cat\_kont']) ?></td>  
 <td width=20%><?= htmlspecialchars($one['status\_cat\_kont']) ?></td>  
 <td width=20%><?= htmlspecialchars($one['can\_employees\_add']) ?></td>  
 <td width=20%><?= htmlspecialchars($one['city\_cat\_kont']) ?></td>  
 <td><button onclick="deleteContact(<?= $one['id\_cat\_kont'] ?>)" class="btn btn-danger btn-xs">Удалить</button></td>  
 </tr>  
   
 <?php } ?>  
 </tbody>  
 </table>  
 <div class="idid"></div>  
 <script src="/js/jquery-3.2.1.js"></script>  
 <script src="/js/bootstrap.min.js"></script>  
 <script type="text/javascript" charset="utf8" src="/js/jquery.dataTables.min.js"></script>  
 <script>  
 $(function(){  
 $('#catkonts').DataTable({  
 "searching": false,  
 "stateSave": true,  
 "language":{  
 info:"Показано с \_START\_ по \_END\_ из \_TOTAL\_ найденных",  
 "paginate": {  
 "first": "Первая",  
 "last": "Последняя",  
 "next": " Следующая",  
 "previous": "Предыдущая "  
 },  
 "infoEmpty": "Записей не найдено",  
 "emptyTable": "Записей не найдено",  
 "lengthMenu": "Показывать по \_MENU\_ "  
 }  
 });  
 });  
 </script>  
</body>  
</html>

# РУЧНОЕ ТЕСТИРОВАНИЕ ВЕБ-ПОРТАЛА

В процессе разработки веб-интерфейса было проведено ручное тестирование основных пользовательских сценариев. Целью тестирования была проверка корректности работы интерфейса, логики взаимодействия с данными, адаптивности сайта и устойчивости к возможным пользовательским ошибкам.

| **№** | **Тестируемый сценарий** | **Входные данные** | **Этапы тестирования** | **Ожидаемый результат** | **Факт. результат** | **Статус** |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | Авторизация пользователя (существующий пользователь) | Валидные логин/пароль:  Логин: admin  Пароль: admin | Зашел на страницу, защищенный авторизацией.  Переадресация на страницу авторизации.  Ввел данные авторизации.  Зашел на страницу, защищенную авторизацией. | Успешный вход в страницу, защищенную авторизацией. | Успешный вход в страницу, защищенную авторизацией. | Пройден |
| 1.1 | Авторизация пользователя  (несуществующий пользователь + SQL инъекция) | Невалидные логин/пароль:  Логин: log  Пароль: pass' OR '1'='1 | Зашел на страницу, защищенный авторизацией.  Переадресация на страницу авторизации.  Ввел данные авторизации.  Сообщение об ошибке. | Блокировка входа, сообщение об ошибке | Блокировка входа, сообщение об ошибке | Пройден |
| 2 | Поиск препаратов (существующий препарат) | Существующий препарат, например: парацетамол | Зашел на страницу “Справка” в строку поиска препаратов ввел данные.  Появилась таблица с информацией. | Отображение списка препаратов | Отображение списка препаратов | Пройден |
| 2.1 | Поиск препаратов  (несуществующий препарат) | Несуществующий препарат, например: asdf | Зашел на страницу “Справка” в строку поиска препаратов ввел данные.  Появилось сообщение "Совпадений не найдено". | Сообщение "Совпадений не найдено" | Сообщение "Совпадений не найдено" | Пройден |
| 3 | Создание новости (со всеми данными) | Корректные данные, например: Заголовок: test  Краткое описание: test  Полное описание: test | Зашел на страницу. “Новости” нажал кнопку “Добавить” ввел данные нажал на кнопку “Добавить пост”.  Новость появилась в общем списке новостей. | Новая новость сохраняется в БД | Новая новость сохраняется в БД | Пройден |
| 3.1 | Создание новости (без данных) | Пустые обязательные поля | Зашел на страницу “Новости” нажал кнопку “Добавить”, не вводя данные нажал на кнопку “Добавить пост”.  Появляется сообщение о пропуске полей. | Блокировка сохранения, ошибка валидации | Блокировка сохранения, ошибка валидации | Пройден |
| 4 | Участие в опросе (успешное) | Выбран вариант ответа | Зашел на страницу “Новости” нажал на новость с опросом.  Выбрал варианты ответа, нажал на кнопку “Ответить”  Сообщение “Опрос успешно завершен” | Ответ записан в БД | Ответ записан в БД | Пройден |
| 4.1 | Участие в опросе (неуспешное) | Ответ не выбран | Зашел на страницу “Новости” нажал на новость с опросом.  Не выбирая варианты ответа, нажал на кнопку “Ответить”  Появляется сообщение о пропуске полей. | Предупреждение "Выберите вариант" | Предупреждение "Выберите вариант" | Пройден |
| 5 | Добавление комментария | Текст комментария, например:  Спасибо за информацию. | Зашел на страницу “Новости” нажал на новость.  В поле комментария ввел текст нажал кнопку “Отправить”  Комментарий появился под новостью | Комментарий отображается в списке | Комментарий отображается в списке | Пройден |
| 6 | Просмотр логов активности | Запрос логов (админ) | Авторизировался. Зашел на страницу “Новости” нажал на новость.  В новости нажал кнопку “Просмотреть лог записи”.  Открылась страница со всеми логами, связанными с этой новостью. | Отображение списка событий | Отображение списка событий | Пройден |
| 7 | Создание контакта | Ввод данных, например;  Контактное лицо: Тимофеев Александр Викторович  Номер: +7 999 999 99 99  Почта: oldmail@mail.ru  Описание: Test  Категория: Внутринние | Зашел на страницу “Контакты”.  Нажал на кнопку “Добавить контакт”.  Ввел данные контакта и нажал на кнопку “ Добавить контакт”.  Контакт появился в общем списке контактов. | Контакт записан в БД | Контакт записан в БД | Пройден |
| 8 | Редактирование контакта | Изменение данных, например:  Номер: +7 999 444 44 44  Почта:  newmail@mail.ru | Зашел на страницу “Контакты”.  Нажал у нужного контакта кнопку редактирования.  В открывшейся странице заменяю старые данные на новые и наживаю на кнопку “Сохранить изменения”.  Контакт изменился в списке. | Обновленные данные сохраняются | Обновленные данные сохраняются | Пройден |
| 9 | Навигация по меню | Клик по пункту меню | Нажал на пункт “Новости” открылась страница с новостями  Нажал на пункт “Полезные ссылки” открылась страница с полезными ссылками | Корректный переход на страницу | Корректный переход на страницу | Пройден |
| 10 | Адаптивность интерфейса | Разрешения: 1920x1080, 1280x1024, 1152x864 | Нажимаю на клавишу F12 и нажимаю комбинацию клавиш Ctrl + Shift + M в открывшимся меню ввожу нужные значения разрешения и проверяю расположение элементов | Корректное отображение на всех разрешениях | Корректное отображение на всех разрешениях | Пройден |

**Выводы:**

* Все CRUD-операции (создание, чтение, обновление, удаление) работают корректно.
* Интерфейс устойчив к некорректным действиям пользователя.
* Доработка не требуется.

Система готова к эксплуатации.

# РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

Внутренний веб-портал аптечной сети предназначен для централизованного доступа сотрудников к ключевым рабочим ресурсам, включая поиск препаратов, внутренние новости, полезные ссылки и контактную информацию.

Основные функции портала включают:

* Поиск лекарственных средств с фильтрацией по городам и брендам аптек;
* Просмотр и комментирование новостей, включая важные объявления и опросы;
* Доступ к полезным ссылкам и связанным с ними учетным данным;
* Управление контактами сотрудников и партнеров.

**1. Введение**

Данное руководство предназначено для сотрудников аптечной сети, использующих внутренний веб-портал для работы. Портал предоставляет доступ к важной информации, инструментам поиска препаратов, новостям, контактам и другим ресурсам, необходимым для ежедневной работы.

**2. Начало работы**

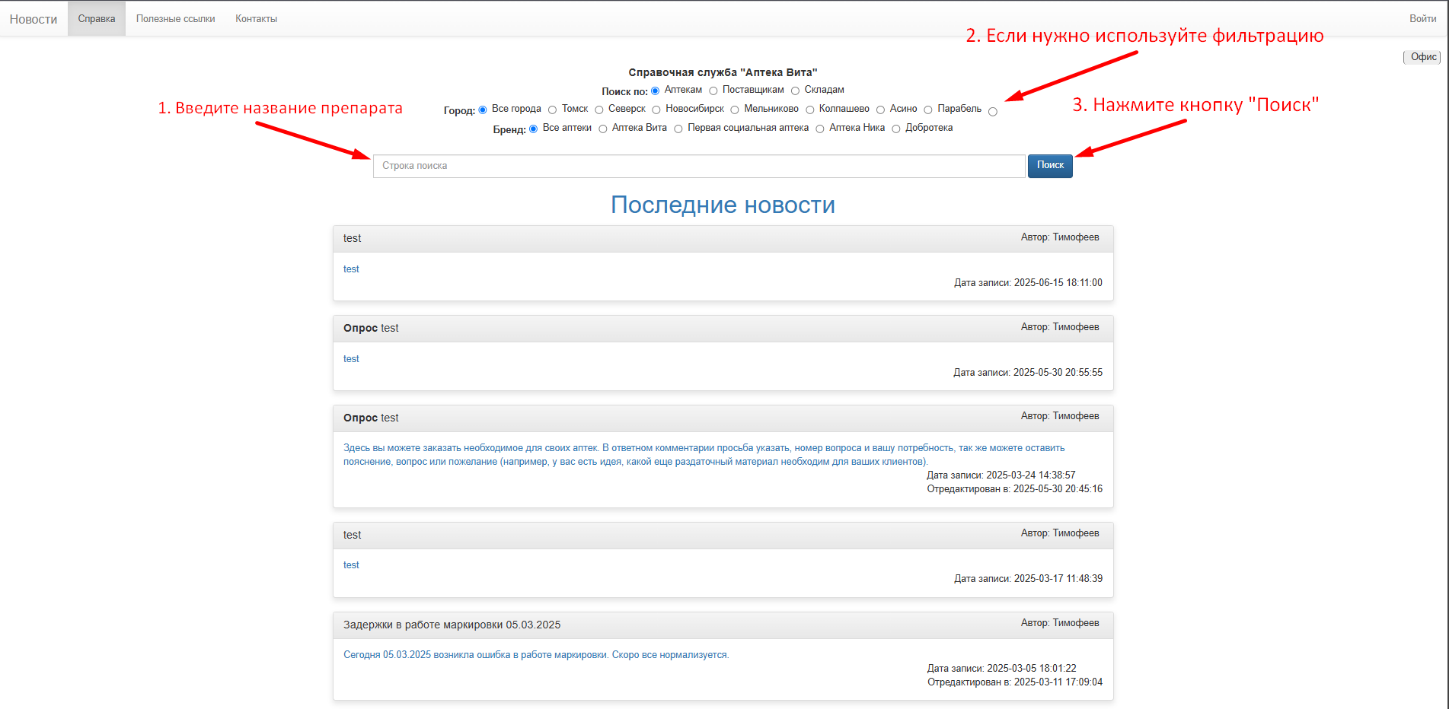
**2.1. Доступ к порталу**

* Откройте браузер и перейдите по адресу внутреннего портала
* Обычные пользователи идентифицируются автоматически по IP-адресу
* Администраторы входят с помощью логина и пароля

**3. Основные разделы портала**

**3.1. Справка (Главная страница)**

* **Последние новости**:
  + Просматривайте краткую информацию о новостях
  + Нажмите на заголовок новости для просмотра полного текста
* **Поиск препаратов**:
  + Введите название препарата в строку поиска
  + Используйте фильтры по городам, районам и брендам аптек для уточнения результатов



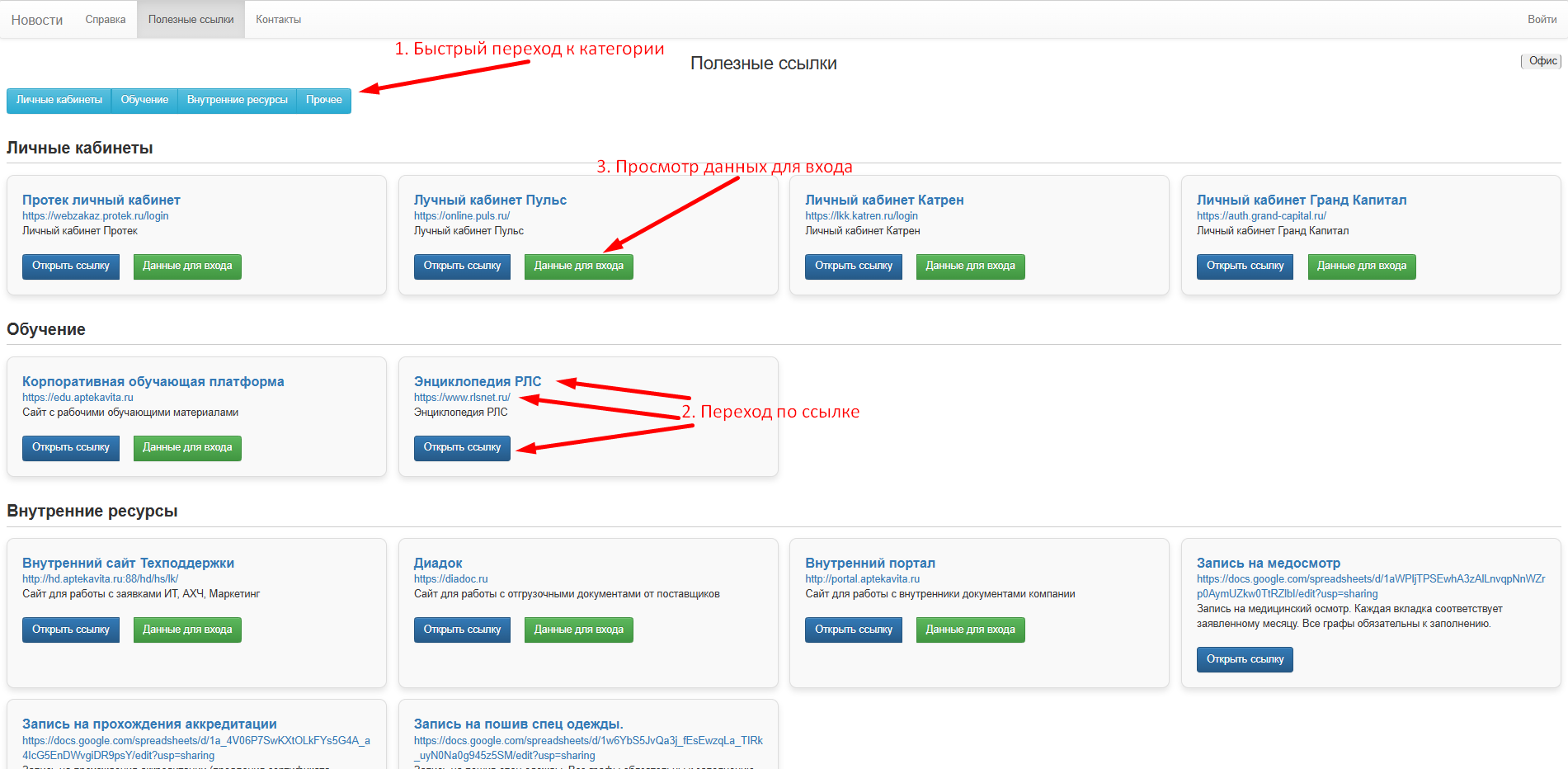
* + Результаты поиска отображаются в таблице ниже

**3.2. Новости**

* Просматривайте список всех новостей
* Фильтруйте новости по типам: "Все", "Новости", "Опросы", "Инструкции"
* Важные новости помечены специальными метками ("Актуально", "Важный")
* Для просмотра полного содержания новости нажмите на её заголовок или краткое описание.
* Оставляйте комментарии к новостям (если разрешено)

**3.3. Полезные ссылки**

* Просматривайте список полезных ресурсов, сгруппированных по категориям
* Быстро переходите по ссылкам, нажав на них
* Для доступа к учетным данным нажмите "Данные для входа" рядом с нужной ссылкой



**3.4. Контакты**

* Просматривайте список контактов, сгруппированных по категориям
* Используйте кнопки-якоря для быстрого перехода к нужной категории
* Экстренные контакты отображаются в отдельном разделе

**4. Работа с данными для входа**

**4.1. Просмотр данных**

1. Перейдите в раздел "Полезные ссылки"
2. Нажмите "Данные для входа" рядом с нужной ссылкой
3. Просматривайте список логинов и паролей

**4.2. Добавление новых данных**

1. На странице данных для входа нажмите "Добавить"
2. Заполните форму: логин, пароль, описание
3. Нажмите "Сохранить"

**4.3. Редактирование данных**

1. На странице данных для входа нажмите "Редактировать" рядом с нужной записью
2. Внесите изменения
3. Нажмите "Сохранить"

**4.4. Удаление данных**

1. На странице данных для входа нажмите "Удалить" рядом с нужной записью
2. Подтвердите удаление

**5. Работа с комментариями к новостям**

1. Откройте новость, к которой хотите оставить комментарий
2. Введите текст комментария в соответствующее поле
3. При необходимости прикрепите файлы
4. Нажмите "Отправить"

**6. Участие в опросах**

1. Откройте новость с опросом(подробнее)
2. Ответьте на вопросы (текстовый ответ, выбор одного или нескольких вариантов)
3. Нажмите "Ответить"

**7. Техническая поддержка**

В случае возникновения проблем:

1. Проверьте подключение к сети
2. Очистите кэш браузера
3. Попробуйте перезагрузить страницу
4. Если проблема сохраняется, обратитесь к системному администратору

# РУКОВОДСТВО АДМИНИСТРАТОРА

Внутренний портал предназначен для централизованного управления информацией, коммуникации между сотрудниками и упрощения рабочих процессов в аптечной сети.

**Описание основного функционала**

* Управление новостями и опросами
* Контроль контактов и полезных ссылок
* Администрирование городов, брендов и юридических лиц
* Работа с прайс-листами

**1. Вход в систему**

* Перейдите на страницу входа нажав на кнопку “Вход” в меню навигации.
* Введите логин и пароль, предоставленные системным администратором.
* Нажмите кнопку "Войти".

**2. Основные функции администратора**

**2.1 Управление новостями**

* Просмотр новостей: Все новости отображаются на главной странице и в разделе "Новости".
* Добавление новости:
  1. Нажмите кнопку "Добавить" в разделе новостей.
  2. Заполните форму: заголовок, краткое и полное описание.
  3. При необходимости прикрепите файлы или фотографии.
  4. Установите параметры важности, видимости и возможность комментирования.
  5. Нажмите "Добавить пост".
* Редактирование новости: Нажмите кнопку "Редактировать" на странице новости, которую вы хотите редактировать.
* Удаление новости: Нажмите кнопку "Удалить" на странице со всеми новостями.

**2.2 Управление полезными ссылками**

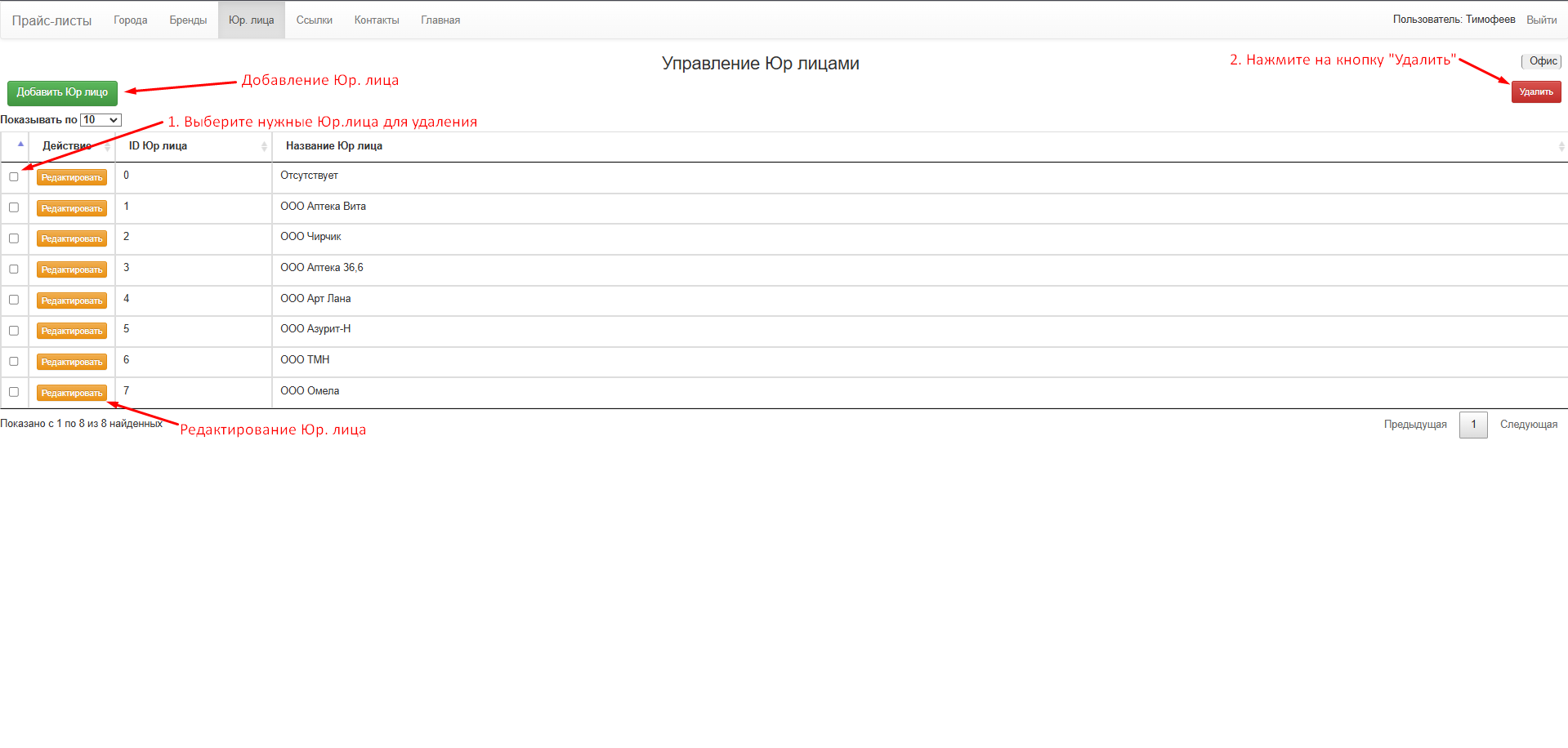
* **Добавление ссылки**:
  1. Перейдите в раздел "Ссылки".
  2. Нажмите "Добавить ссылку".
  3. Заполните название, URL и описание.
  4. Укажите, могут ли сотрудники добавлять данные для входа.
* **Редактирование/удаление**: Используйте соответствующие кнопки рядом с каждой ссылкой.

**2.3 Управление контактами**

* **Добавление контакта**:
  1. Перейдите в раздел "Контакты".
  2. Нажмите "Добавить контакт".
  3. Заполните информацию о контакте (имя, телефон, email, описание).
  4. Выберите категорию контакта.
* Управление категориями контактов: Доступно через кнопку "Управление категориями".

**2.4 Управление городами, брендами и юридическими лицами**

* Работа с этими разделами осуществляется аналогично:
  1. Перейдите в соответствующий раздел.
  2. Используйте кнопки "Добавить", "Редактировать", "Удалить".
  3. Для городов можно также управлять районами.



**2.5 Управление прайс-листами**

* В разделе "Прайс-листы" можно добавлять, редактировать и удалять прайс-листы аптек.

**3. Работа с пользователями**

* Просмотр активности: Через журналы действий ("Просмотреть лог записи" в нужной новости).
* Просмотр ответов на опросы: Кнопка "Просмотреть ответы" в новости с опросом.
* Управление комментариями: Возможность удаления неподобающих комментариев.

**4. Безопасность**

* Никому не передавайте свои учетные данные.
* Своевременно удаляйте неактуальные новости и контакты.

**5. Выход из системы**

* Всегда нажимайте кнопку "Выход" после завершения работы.
* Не оставляйте активную сессию на общедоступных компьютерах.

**6. Техническая поддержка**

При возникновении проблем:

1. Попробуйте обновить страницу (F5).
2. Очистите кеш браузера.
3. Если проблема остаётся, обратитесь к системному администратору, указав:
   * Какие действия выполняли
   * Текст ошибки (если была)
   * Время возникновения проблемы

# ЗАКЛЮЧЕНИЕ

В ходе выполнения дипломного проекта была успешно достигнута поставленная цель - модернизация и редизайн существующего веб-портала для сотрудников аптечной сети. В результате проведённой работы удалось значительно улучшить функциональность, удобство использования и визуальное оформление системы.

Были выполнены все поставленные задачи:

1. Разработать техническое задание веб-портала.
2. Разработать прототип интерфейса с учётом удобства и минимизации ошибок при работе.
3. Добавить разделы для отображения внутренних новостей, полезных ссылок, данных для входа к ним и контактной информации.
4. Разработать ручные тесты и провести тестирование веб-портала.

Проведённое тестирование продемонстрировало стабильную работу всех модулей портала, включая CRUD-операции, систему авторизации и механизмы обработки данных. Особое внимание было уделено адаптивности интерфейса и его корректному отображению на различных устройствах.

Таким образом, дипломный проект успешно завершён, а разработанное решение готово к внедрению в производственную среду. Дальнейшее развитие системы может включать расширение функциональности и внедрение дополнительных средств автоматизации рабочих процессов.

# СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Руководство по MySql / Режим доступа: <https://metanit.com/sql/mysql/>, открытый.
2. Руководство по PHP / Режим доступа: <https://metanit.com/php/tutorial/>, открытый.
3. Учебник JavaScript / Режим доступа: <https://learn.javascript.ru/>, открытый.
4. Документация Bootstrap / Режим доступа: <https://getbootstrap.com/docs/5.3>, открытый.
5. Учебник HTML и CSS / Режим доступа: <https://code.mu/ru/markup/book/supreme/>, открытый.
6. JavaScript. Полное руководство. 7-е изд. — СПб.: Диалектика, 2023. — 1088 с.: ил.
7. PHP и MySQL. Разработка веб-приложений. 6-е изд. — СПб.: БХВ-Петербург, 2022. — 752 с.: ил.
8. Тестирование программного обеспечения. Базовый курс. 3-е изд. — СПб.: Питер, 2022. — 464 с.: ил. — (Серия «Библиотека программиста»).
9. UI/UX дизайн. Принципы создания интерфейсов. — М.: Манн, Иванов и Фербер, 2021. — 368 с.: ил.
10. Web-программирование для всех. — СПб.: БХВ-Петербург, 2008. — 560 с.: ил.

# ПРИЛОЖЕНИЕ 1

Sql код базы с тестовыми данными.

SET SQL\_MODE = "NO\_AUTO\_VALUE\_ON\_ZERO";

START TRANSACTION;

SET time\_zone = "+00:00";

/\*!40101 SET @OLD\_CHARACTER\_SET\_CLIENT=@@CHARACTER\_SET\_CLIENT \*/;

/\*!40101 SET @OLD\_CHARACTER\_SET\_RESULTS=@@CHARACTER\_SET\_RESULTS \*/;

/\*!40101 SET @OLD\_COLLATION\_CONNECTION=@@COLLATION\_CONNECTION \*/;

/\*!40101 SET NAMES utf8mb4 \*/;

CREATE TABLE `brand` (

`id` int NOT NULL,

`name` varchar(255) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

INSERT INTO `brand` (`id`, `name`) VALUES

(1, 'Аптека Вита'),

(2, 'Первая социальная аптека'),

(3, 'Аптека Ника'),

(4, 'Добротека');

CREATE TABLE `cities` (

`id` int NOT NULL,

`name` varchar(255) NOT NULL,

`ip` varchar(255) NOT NULL,

`is\_enabled` tinyint(1) NOT NULL,

`visible\_cities` int NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

INSERT INTO `cities` (`id`, `name`, `ip`, `is\_enabled`, `visible\_cities`) VALUES

(1, 'Томск', '10.70.', 1, 2),

(2, 'Северск', '10.70.', 0, 1),

(3, 'Новосибирск', '10.54.', 1, 2),

(4, 'Мельниково', '10.70.', 0, 1),

(5, 'Колпашево', '10.70.', 0, 1),

(6, 'Асино', '10.70.', 0, 1),

(7, 'Парабель', '10.70.', 1, 6);

CREATE TABLE `comments` (

`id` int NOT NULL,

`post\_id` int NOT NULL,

`user\_id` int NOT NULL,

`text` mediumtext,

`photo` varchar(255) DEFAULT NULL,

`date\_created` date NOT NULL,

`is\_public` tinyint(1) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

INSERT INTO `comments` (`id`, `post\_id`, `user\_id`, `text`, `photo`, `date\_created`, `is\_public`) VALUES

(69, 64, 1, '.', 'news/comments/93\_665ff5da728ee.sql', '2024-06-05', 0),

(70, 64, 1, '', 'news/comments/93\_665ff605997fc.sql', '2024-06-05', 0),

(71, 64, 2, 'da', NULL, '2024-06-05', 1),

(72, 64, 2, 'da', NULL, '2024-06-05', 1),

(73, 64, 2, 'da', NULL, '2024-06-06', 1),

(74, 64, 1, 'eqw', NULL, '2024-06-10', 1),

(75, 64, 2, '', NULL, '2024-06-10', 1),

(76, 64, 2, 'xj&', NULL, '2024-06-10', 0),

(77, 64, 1, 'ad', NULL, '2024-06-10', 0),

(78, 64, 1, 'da', NULL, '2024-06-10', 0);

CREATE TABLE `district` (

`id` int NOT NULL,

`name` varchar(255) NOT NULL,

`city\_id` int NOT NULL,

`is\_enabled` tinyint(1) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

INSERT INTO `district` (`id`, `name`, `city\_id`, `is\_enabled`) VALUES

(1, 'Кировский', 1, 1),

(2, 'Советский', 1, 1),

(3, 'Октябрьский', 1, 1),

(4, 'Ленинский', 1, 1),

(5, 'Заельцовский', 3, 1),

(6, 'Октябрьский', 3, 1),

(7, 'Советский', 3, 1),

(8, 'Железнодорожный', 3, 1),

(9, 'Калининский', 3, 1),

(10, 'Ленинский', 3, 1),

(11, 'Первомайский', 3, 1),

(12, 'Центральный', 3, 1);

CREATE TABLE `firm` (

`id` int NOT NULL,

`name` varchar(255) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

INSERT INTO `firm` (`id`, `name`) VALUES

(1, 'ООО Аптека Вита'),

(2, 'ООО Чирчик'),

(3, 'ООО Аптека 36,6'),

(4, 'ООО Арт Лана'),

(5, 'ООО Азурит-Н'),

(6, 'ООО ТМН'),

(7, 'ООО Омела');

CREATE TABLE `importance` (

`id` int NOT NULL,

`name` varchar(255) NOT NULL,

`code` varchar(255) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

INSERT INTO `importance` (`id`, `name`, `code`) VALUES

(1, 'Current', 'current'),

(2, 'Important', 'important'),

(3, 'Temporary', 'temporary');

CREATE TABLE `kontakts` (

`id` int NOT NULL,

`type\_kontakt\_id` int NOT NULL,

`urgent` tinyint(1) NOT NULL,

`visibility` tinyint(1) NOT NULL,

`name` varchar(255) NOT NULL,

`phone` varchar(255) NOT NULL,

`email` varchar(255) NOT NULL,

`description` mediumtext NOT NULL,

`author` int NOT NULL,

`last\_editor` int NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

INSERT INTO `kontakts` (`id`, `type\_kontakt\_id`, `urgent`, `visibility`, `name`, `phone`, `email`, `description`, `author`, `last\_editor`) VALUES

(3, 2, 0, 0, 'ntcn ntcn ntcn', '440-689', 'it@aptekavita.ru', '', 2, 2),

(4, 1, 1, 0, 'Экстренный контакт один один', '222-333', '', ' ', 2, 2),

(5, 2, 1, 0, 'Экстренный контакт для всех', '+7 333 892 44-55', '', '', 2, 2),

(6, 1, 0, 0, 'Контакт обычный', '+79138278898', '', '', 2, 2),

(7, 1, 0, 0, 'АХО Антон Николаевич', '+7826736678', '', '', 2, 2),

(8, 1, 0, 0, 'Служба закупа', '44-917', '', '', 2, 2),

(9, 2, 0, 0, 'Офис новосибисрк куратор1', '+7 383 777-8888', '', '', 2, 2),

(10, 1, 0, 0, 'Куратор Северск', '+73823 444-777', '', '', 2, 2);

CREATE TABLE `links` (

`id` int NOT NULL,

`name` varchar(255) NOT NULL,

`url` mediumtext NOT NULL,

`description` mediumtext NOT NULL,

`type\_link\_id` int NOT NULL,

`can\_employees\_add` tinyint(1) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

INSERT INTO `links` (`id`, `name`, `url`, `description`, `type\_link\_id`, `can\_employees\_add`) VALUES

(1, 'Внутренний сайт Техподдержки', 'http://hd.aptekavita.ru:88/hd/hs/lk/', 'Сайт для работы с заявками ИТ, АХЧ, Маркетинг', 3, 1),

(2, 'Диадок', 'https://diadoc.ru', 'Сайт для работы с отгрузочными документами от поставщиков', 3, 1),

(3, 'Внутренний портал', 'http://portal.aptekavita.ru', 'Сайт для работы с внутренники документами компании', 3, 1),

(4, 'Корпоративная обучающая платформа', 'https://edu.aptekavita.ru', 'Сайт с рабочими обучающими материалами', 2, 1),

(5, 'Инкассация Сбербанк', 'https://encashment.sberbank.ru/', 'Заполнение инкассации сбербанк', 4, 1),

(6, 'Протек личный кабинет', 'https://webzakaz.protek.ru/login', 'Личный кабинет Протек', 1, 1);

CREATE TABLE `link\_pas` (

`id` int NOT NULL,

`link\_id` int NOT NULL,

`login` varchar(255) DEFAULT NULL,

`password` varchar(255) DEFAULT NULL,

`description` mediumtext,

`author` int NOT NULL,

`last\_editor` int NOT NULL,

`is\_visible` tinyint(1) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

INSERT INTO `link\_pas` (`id`, `link\_id`, `login`, `password`, `description`, `author`, `last\_editor`, `is\_visible`) VALUES

(1, 1, 'user001', 'Qz7x!pL9aV', '', 1, 1, 1),

(2, 2, 'pharma\_1', '4bYt\*GkM2r', '', 1, 1, 0),

(3, 3, 'worker12', 'Fm!89xZvQp', '', 1, 1, 1),

(4, 4, 'pharmacy23', '3Kj#PqRt9L', '', 1, 1, 0),

(5, 5, 'employeeX', 'Bv8\*LnQz72', '', 1, 1, 1),

(6, 6, 'staff2024', 'TzQp!46MkY', '', 1, 1, 0);

CREATE TABLE `logs` (

`id` int NOT NULL,

`post\_id` int NOT NULL,

`user\_id` int NOT NULL,

`date\_action` date NOT NULL,

`action` varchar(255) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

INSERT INTO `logs` (`id`, `post\_id`, `user\_id`, `date\_action`, `action`) VALUES

(19, 64, 1, '2024-06-04', 'Посмотрел'),

(20, 64, 1, '2024-06-04', 'Просмотрел'),

(21, 64, 1, '2024-06-04', 'Просмотрел'),

(22, 64, 1, '2024-06-04', 'Просмотрел'),

(23, 64, 1, '2024-06-04', 'Просмотрел'),

(24, 64, 1, '2024-06-04', 'Редактировал запись'),

(25, 64, 1, '2024-06-04', 'Просмотрел'),

(26, 64, 1, '2024-06-04', 'Просмотрел'),

(27, 64, 1, '2024-06-04', 'Просмотрел'),

(28, 64, 1, '2024-06-04', 'Просмотрел');

CREATE TABLE `partners` (

`id` int NOT NULL,

`city\_id` int NOT NULL,

`brand\_id` int NOT NULL,

`district\_id` int DEFAULT NULL,

`firm\_id` int NOT NULL,

`type\_partner\_id` int NOT NULL,

`name` varchar(255) NOT NULL,

`code` varchar(255) NOT NULL,

`phone` varchar(255) NOT NULL,

`net` varchar(255) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

INSERT INTO `partners` (`id`, `city\_id`, `brand\_id`, `district\_id`, `firm\_id`, `type\_partner\_id`, `name`, `code`, `phone`, `net`) VALUES

(25, 1, 2, NULL, 2, 1, 'Мира, 19', '20032', '8 (3822) 765-930', '14'),

(26, 1, 1, NULL, 1, 1, 'Ботанический, 3', '10010', '8 (3822) 411-013', '3'),

(28, 1, 1, NULL, 1, 1, 'Иркутский, 102', '10012', '8 (3822) 668-854', '9'),

(29, 1, 1, NULL, 1, 1, 'Иркутский, 162', '10013', '8 (3822) 646-641', '19'),

(30, 1, 1, NULL, 1, 1, 'Киевская, 13', '10014', '8 (3822) 445-513', '22');

CREATE TABLE `posts` (

`id` int NOT NULL,

`author` int NOT NULL,

`last\_editor` int NOT NULL,

`title` varchar(255) NOT NULL,

`short\_description` varchar(255) NOT NULL,

`full\_description` mediumtext NOT NULL,

`attachment\_fotos` varchar(255) DEFAULT NULL,

`attachment\_files` varchar(255) DEFAULT NULL,

`importance\_id` int NOT NULL,

`importance\_date` date DEFAULT NULL,

`allow\_comments` tinyint(1) NOT NULL,

`comment\_visibility` tinyint(1) NOT NULL,

`visibility` tinyint(1) NOT NULL,

`type\_post\_id` int NOT NULL,

`date\_created` date NOT NULL,

`date\_updated` date NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

INSERT INTO `posts` (`id`, `author`, `last\_editor`, `title`, `short\_description`, `full\_description`, `attachment\_fotos`, `attachment\_files`, `importance\_id`, `importance\_date`, `allow\_comments`, `comment\_visibility`, `visibility`, `type\_post\_id`, `date\_created`, `date\_updated`) VALUES

(1, 2, 2, 'Обновление сайта справка Аптека Вита', 'Сегодня сделали небольшое обновление для справки. Добавили на него несколько новых модулей...', '<p>Добрый день.</p><p>Сегодня сделали обновление для сайта справки добавили несколько новых модулей...</p>', '[]', '[\"news/posts/1\_file\_667b80d3a7457.pdf\"]', 2, NULL, 0, 1, 1, 1, '2024-06-13', '2025-03-11'),

(2, 1, 2, 'Краткий стандарт поведения сотрудника аптеки 2024', 'Уважаемые коллеги, прикрепила краткий стандарт поведения сотрудника аптеки.', '<p>Уважаемые коллеги, прикрепила краткий стандарт поведения сотрудника аптеки для наглядности.</p>', '[]', '[]', 1, NULL, 1, 0, 1, 1, '2024-06-14', '2025-03-11'),

(3, 1, 2, 'Осторожно! Мошенники!', 'Самые частые виды мошенничества.', '<p>Уважаемые коллеги, будьте внимательны со звонками или сообщениями с незнакомых номеров.</p>', '[]', '[]', 2, NULL, 1, 1, 1, 1, '2024-06-18', '2025-03-11'),

(4, 2, 2, 'Добавлен раздел с Полезные ссылки', 'Добавлен раздел по сохранению полезных ссылок', '<p>На сайте добавлен раздел в котором можно сохранять нужные в работе ссылки...</p>', '[]', '[]', 1, NULL, 1, 1, 1, 1, '2024-06-27', '2025-03-11'),

(64, 2, 2, 'Замена фискального накопителя в кассе в вашей аптеке', 'Замена фискального накопителя в кассе в вашей аптеке', '<p>ЗАВТРА (10.07.2024) планируем замену ФН в Зоркальцево...</p>', '[]', '[]', 3, '2024-07-11', 1, 1, 1, 1, '2024-07-09', '2025-03-11');

CREATE TABLE `surveys` (

`id` int NOT NULL,

`post\_id` int NOT NULL,

`question` varchar(255) NOT NULL,

`type\_answer\_id` int NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

INSERT INTO `surveys` (`id`, `post\_id`, `question`, `type\_answer\_id`) VALUES

(1, 64, 'Вопрос 1', 2),

(2, 64, 'Вопрос 2', 2),

(3, 64, 'Вопрос 3', 3),

(4, 64, 'Вопрос 4', 1);

CREATE TABLE `survey\_answers` (

`id` int NOT NULL,

`survey\_id` int NOT NULL,

`answer` varchar(255) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

INSERT INTO `survey\_answers` (`id`, `survey\_id`, `answer`) VALUES

(4, 1, 'Ответ к вопросу 1 №1'),

(5, 1, 'Ответ к вопросу 1 №2'),

(6, 1, 'Ответ к вопросу 1 №3'),

(7, 1, 'Ответ к вопросу 1 №4'),

(8, 2, 'Ответ к вопросу 2 №1'),

(9, 2, 'Ответ к вопросу 2 №2'),

(10, 2, 'Ответ к вопросу 2 №3'),

(11, 2, 'Ответ к вопросу 2 №4'),

(12, 3, 'Ответ к вопросу 3 №1'),

(13, 3, 'Ответ к вопросу 3 №2'),

(14, 3, 'Ответ к вопросу 3 №3'),

(15, 3, 'Ответ к вопросу 3 №4');

CREATE TABLE `type\_answer` (

`id` int NOT NULL,

`name` varchar(255) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

INSERT INTO `type\_answer` (`id`, `name`) VALUES

(1, 'text'),

(2, 'radio'),

(3, 'checkbox');

CREATE TABLE `type\_kontakt` (

`id` int NOT NULL,

`name` varchar(255) NOT NULL,

`visibility` tinyint(1) NOT NULL,

`can\_employees\_add` tinyint(1) NOT NULL,

`city\_id` int NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

INSERT INTO `type\_kontakt` (`id`, `name`, `visibility`, `can\_employees\_add`, `city\_id`) VALUES

(1, 'Офис Томск', 0, 0, 1),

(2, 'Офис Новосибирск', 0, 0, 3),

(3, 'Поставщики', 0, 0, 1),

(4, 'Служебные', 1, 1, 1),

(5, 'Внутренние', 1, 1, 1),

(10, 'Медицинский представитель', 1, 1, 1);

CREATE TABLE `type\_link` (

`id` int NOT NULL,

`name` varchar(255) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

INSERT INTO `type\_link` (`id`, `name`) VALUES

(1, 'Личные кабинеты'),

(2, 'Обучение'),

(3, 'Внутренние ресурсы'),

(4, 'Прочее');

CREATE TABLE `type\_partner` (

`id` int NOT NULL,

`name` varchar(255) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

INSERT INTO `type\_partner` (`id`, `name`) VALUES

(1, 'Аптека'),

(2, 'Склад'),

(3, 'Поставщик');

CREATE TABLE `type\_post` (

`id` int NOT NULL,

`name` varchar(255) NOT NULL,

`code` varchar(255) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

INSERT INTO `type\_post` (`id`, `name`, `code`) VALUES

(1, 'Новость', 'news'),

(2, 'Объявление', 'announcement'),

(3, 'Опрос', 'survey');

CREATE TABLE `users` (

`id` int NOT NULL,

`name` varchar(255) NOT NULL,

`login` varchar(255) NOT NULL,

`password` varchar(255) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

INSERT INTO `users` (`id`, `name`, `login`, `password`) VALUES

(1, 'Ит отдел', 'admin', 'admin'),

(2, 'Тимофеев', 'da', 'da');

CREATE TABLE `user\_answers` (

`id` int NOT NULL,

`user\_id` int NOT NULL,

`survey\_answer\_id` int DEFAULT NULL,

`survey\_id` int NOT NULL,

`answer` mediumtext

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

INSERT INTO `user\_answers` (`id`, `user\_id`, `survey\_answer\_id`, `survey\_id`, `answer`) VALUES

(1, 2, 4, 1, NULL),

(2, 1, NULL, 4, 'Ответ на Вопрос 4'),

(3, 1, 4, 2, NULL),

(4, 2, NULL, 4, 'Ответ на Вопрос 4');

ALTER TABLE `brand`

ADD PRIMARY KEY (`id`),

ADD UNIQUE KEY `id` (`id`);

ALTER TABLE `cities`

ADD PRIMARY KEY (`id`),

ADD UNIQUE KEY `id` (`id`),

ADD KEY `cities\_index\_0` (`visible\_cities`);

ALTER TABLE `comments`

ADD PRIMARY KEY (`id`),

ADD UNIQUE KEY `id` (`id`),

ADD KEY `comments\_index\_0` (`post\_id`),

ADD KEY `comments\_index\_1` (`user\_id`);

ALTER TABLE `district`

ADD PRIMARY KEY (`id`),

ADD UNIQUE KEY `id` (`id`),

ADD KEY `district\_index\_0` (`city\_id`);

ALTER TABLE `firm`

ADD PRIMARY KEY (`id`),

ADD UNIQUE KEY `id` (`id`);

ALTER TABLE `importance`

ADD PRIMARY KEY (`id`),

ADD UNIQUE KEY `id` (`id`);

ALTER TABLE `kontakts`

ADD PRIMARY KEY (`id`),

ADD UNIQUE KEY `id` (`id`),

ADD KEY `kontakts\_index\_0` (`type\_kontakt\_id`),

ADD KEY `kontakts\_index\_1` (`author`),

ADD KEY `kontakts\_index\_2` (`last\_editor`);

ALTER TABLE `links`

ADD PRIMARY KEY (`id`),

ADD UNIQUE KEY `id` (`id`),

ADD KEY `links\_index\_0` (`type\_link\_id`);

ALTER TABLE `link\_pas`

ADD PRIMARY KEY (`id`),

ADD UNIQUE KEY `id` (`id`),

ADD KEY `link\_pas\_index\_0` (`link\_id`),

ADD KEY `link\_pas\_index\_1` (`author`),

ADD KEY `link\_pas\_index\_2` (`last\_editor`);

ALTER TABLE `logs`

ADD PRIMARY KEY (`id`),

ADD UNIQUE KEY `id` (`id`),

ADD KEY `post\_id` (`post\_id`),

ADD KEY `user\_id` (`user\_id`);

ALTER TABLE `partners`

ADD PRIMARY KEY (`id`),

ADD UNIQUE KEY `id` (`id`),

ADD KEY `partners\_index\_0` (`city\_id`),

ADD KEY `partners\_index\_1` (`brand\_id`),

ADD KEY `partners\_index\_2` (`district\_id`),

ADD KEY `partners\_index\_3` (`firm\_id`),

ADD KEY `partners\_index\_4` (`type\_partner\_id`);

ALTER TABLE `posts`

ADD PRIMARY KEY (`id`),

ADD UNIQUE KEY `id` (`id`),

ADD KEY `posts\_index\_0` (`author`),

ADD KEY `posts\_index\_1` (`last\_editor`),

ADD KEY `posts\_index\_2` (`importance\_id`),

ADD KEY `posts\_index\_3` (`type\_post\_id`);

ALTER TABLE `surveys`

ADD PRIMARY KEY (`id`),

ADD UNIQUE KEY `id` (`id`),

ADD KEY `surveys\_index\_0` (`post\_id`),

ADD KEY `surveys\_index\_1` (`type\_answer\_id`);

ALTER TABLE `survey\_answers`

ADD PRIMARY KEY (`id`),

ADD UNIQUE KEY `id` (`id`),

ADD KEY `survey\_answers\_index\_0` (`survey\_id`);

ALTER TABLE `type\_answer`

ADD PRIMARY KEY (`id`),

ADD UNIQUE KEY `id` (`id`);

ALTER TABLE `type\_kontakt`

ADD PRIMARY KEY (`id`),

ADD UNIQUE KEY `id` (`id`),

ADD KEY `type\_kontakt\_index\_0` (`city\_id`);

ALTER TABLE `type\_link`

ADD PRIMARY KEY (`id`),

ADD UNIQUE KEY `id` (`id`);

ALTER TABLE `type\_partner`

ADD PRIMARY KEY (`id`),

ADD UNIQUE KEY `id` (`id`);

ALTER TABLE `type\_post`

ADD PRIMARY KEY (`id`),

ADD UNIQUE KEY `id` (`id`);

ALTER TABLE `users`

ADD PRIMARY KEY (`id`),

ADD UNIQUE KEY `id` (`id`);

ALTER TABLE `user\_answers`

ADD PRIMARY KEY (`id`),

ADD UNIQUE KEY `id` (`id`),

ADD KEY `user\_answers\_index\_0` (`user\_id`),

ADD KEY `user\_answers\_index\_1` (`survey\_answer\_id`),

ADD KEY `user\_answers\_index\_2` (`survey\_id`);

ALTER TABLE `brand`

MODIFY `id` int NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=5;

ALTER TABLE `cities`

MODIFY `id` int NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=8;

ALTER TABLE `comments`

MODIFY `id` int NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=79;

ALTER TABLE `district`

MODIFY `id` int NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=13;

ALTER TABLE `firm`

MODIFY `id` int NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=9;

ALTER TABLE `importance`

MODIFY `id` int NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=4;

ALTER TABLE `kontakts`

MODIFY `id` int NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=11;

ALTER TABLE `links`

MODIFY `id` int NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=7;

ALTER TABLE `link\_pas`

MODIFY `id` int NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=18;

ALTER TABLE `logs`

MODIFY `id` int NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=29;

ALTER TABLE `partners`

MODIFY `id` int NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=31;

ALTER TABLE `posts`

MODIFY `id` int NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=65;

ALTER TABLE `surveys`

MODIFY `id` int NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=5;

ALTER TABLE `survey\_answers`

MODIFY `id` int NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=16;

ALTER TABLE `type\_answer`

MODIFY `id` int NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=4;

ALTER TABLE `type\_kontakt`

MODIFY `id` int NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=11;

ALTER TABLE `type\_link`

MODIFY `id` int NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=5;

ALTER TABLE `type\_partner`

MODIFY `id` int NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=4;

ALTER TABLE `type\_post`

MODIFY `id` int NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=4;

ALTER TABLE `users`

MODIFY `id` int NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=3;

ALTER TABLE `user\_answers`

MODIFY `id` int NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=5;

ALTER TABLE `cities`

ADD CONSTRAINT `cities\_ibfk\_1` FOREIGN KEY (`visible\_cities`) REFERENCES `cities` (`id`) ON UPDATE CASCADE;

ALTER TABLE `comments`

ADD CONSTRAINT `comments\_ibfk\_1` FOREIGN KEY (`post\_id`) REFERENCES `posts` (`id`) ON UPDATE CASCADE,

ADD CONSTRAINT `comments\_ibfk\_2` FOREIGN KEY (`user\_id`) REFERENCES `users` (`id`) ON UPDATE CASCADE;

ALTER TABLE `district`

ADD CONSTRAINT `district\_ibfk\_1` FOREIGN KEY (`city\_id`) REFERENCES `cities` (`id`) ON UPDATE CASCADE;

ALTER TABLE `kontakts`

ADD CONSTRAINT `kontakts\_ibfk\_1` FOREIGN KEY (`type\_kontakt\_id`) REFERENCES `type\_kontakt` (`id`) ON UPDATE CASCADE,

ADD CONSTRAINT `kontakts\_ibfk\_2` FOREIGN KEY (`author`) REFERENCES `users` (`id`) ON UPDATE CASCADE,

ADD CONSTRAINT `kontakts\_ibfk\_3` FOREIGN KEY (`last\_editor`) REFERENCES `users` (`id`) ON UPDATE CASCADE;

ALTER TABLE `links`

ADD CONSTRAINT `links\_ibfk\_1` FOREIGN KEY (`type\_link\_id`) REFERENCES `type\_link` (`id`) ON UPDATE CASCADE;

ALTER TABLE `link\_pas`

ADD CONSTRAINT `link\_pas\_ibfk\_1` FOREIGN KEY (`link\_id`) REFERENCES `links` (`id`) ON UPDATE CASCADE,

ADD CONSTRAINT `link\_pas\_ibfk\_2` FOREIGN KEY (`author`) REFERENCES `users` (`id`) ON UPDATE CASCADE,

ADD CONSTRAINT `link\_pas\_ibfk\_3` FOREIGN KEY (`last\_editor`) REFERENCES `users` (`id`) ON UPDATE CASCADE;

ALTER TABLE `logs`

ADD CONSTRAINT `logs\_ibfk\_1` FOREIGN KEY (`post\_id`) REFERENCES `posts` (`id`) ON UPDATE CASCADE,

ADD CONSTRAINT `logs\_ibfk\_2` FOREIGN KEY (`user\_id`) REFERENCES `users` (`id`) ON UPDATE CASCADE;

ALTER TABLE `partners`

ADD CONSTRAINT `partners\_ibfk\_1` FOREIGN KEY (`city\_id`) REFERENCES `cities` (`id`) ON UPDATE CASCADE,

ADD CONSTRAINT `partners\_ibfk\_2` FOREIGN KEY (`brand\_id`) REFERENCES `brand` (`id`) ON UPDATE CASCADE,

ADD CONSTRAINT `partners\_ibfk\_3` FOREIGN KEY (`district\_id`) REFERENCES `district` (`id`) ON UPDATE CASCADE,

ADD CONSTRAINT `partners\_ibfk\_4` FOREIGN KEY (`firm\_id`) REFERENCES `firm` (`id`) ON UPDATE CASCADE,

ADD CONSTRAINT `partners\_ibfk\_5` FOREIGN KEY (`type\_partner\_id`) REFERENCES `type\_partner` (`id`) ON UPDATE CASCADE;

ALTER TABLE `posts`

ADD CONSTRAINT `posts\_ibfk\_1` FOREIGN KEY (`author`) REFERENCES `users` (`id`) ON UPDATE CASCADE,

ADD CONSTRAINT `posts\_ibfk\_2` FOREIGN KEY (`last\_editor`) REFERENCES `users` (`id`) ON UPDATE CASCADE,

ADD CONSTRAINT `posts\_ibfk\_3` FOREIGN KEY (`importance\_id`) REFERENCES `importance` (`id`) ON UPDATE CASCADE,

ADD CONSTRAINT `posts\_ibfk\_4` FOREIGN KEY (`type\_post\_id`) REFERENCES `type\_post` (`id`) ON UPDATE CASCADE;

ALTER TABLE `surveys`

ADD CONSTRAINT `surveys\_ibfk\_1` FOREIGN KEY (`post\_id`) REFERENCES `posts` (`id`) ON UPDATE CASCADE,

ADD CONSTRAINT `surveys\_ibfk\_2` FOREIGN KEY (`type\_answer\_id`) REFERENCES `type\_answer` (`id`) ON UPDATE CASCADE;

ALTER TABLE `survey\_answers`

ADD CONSTRAINT `survey\_answers\_ibfk\_1` FOREIGN KEY (`survey\_id`) REFERENCES `surveys` (`id`) ON UPDATE CASCADE;

ALTER TABLE `type\_kontakt`

ADD CONSTRAINT `type\_kontakt\_ibfk\_1` FOREIGN KEY (`city\_id`) REFERENCES `cities` (`id`) ON UPDATE CASCADE;

ALTER TABLE `user\_answers`

ADD CONSTRAINT `user\_answers\_ibfk\_1` FOREIGN KEY (`survey\_id`) REFERENCES `surveys` (`id`) ON UPDATE CASCADE,

ADD CONSTRAINT `user\_answers\_ibfk\_2` FOREIGN KEY (`survey\_answer\_id`) REFERENCES `survey\_answers` (`id`) ON UPDATE CASCADE,

ADD CONSTRAINT `user\_answers\_ibfk\_3` FOREIGN KEY (`user\_id`) REFERENCES `users` (`id`) ON UPDATE CASCADE;

COMMIT;

/\*!40101 SET CHARACTER\_SET\_CLIENT=@OLD\_CHARACTER\_SET\_CLIENT \*/;

/\*!40101 SET CHARACTER\_SET\_RESULTS=@OLD\_CHARACTER\_SET\_RESULTS \*/;

/\*!40101 SET COLLATION\_CONNECTION=@OLD\_COLLATION\_CONNECTION \*/;