ТРЕБОВАНИЯ К ПРОЕКТУ ПО КУРСУ «ВЕБ-РАЗРАБОТКА»

# Введение

Требуется разработать веб-приложение, которое будет включать в себя:

* Создание таблиц в базе данных СУБД PostgreSQL.
* Реализация запросов в БД.
* Реализация как серверной, так и клинской отрисовки HTML.
* Аутентификация по логину/паролю и авторизация на основе ролей для доступа к контенту.
* Реализация полноценного CRUD (создание/получение/изменение/удаление записей) из веб-интерфейса минимум для одной таблицы.

На защите будет демонстрироваться сам сайт, а также исходный код. Ожидается отсутствие непредвиденных ошибок на демонстрации.

Быть готовым ответить на вопросы:

* Какой css класс отвечает за такой-то элемент
* Как работает функция удаления записи в js
* Где происходит сохранение ролей в сессию
* Как именно сервер преобразует результаты запроса БД в JSON
* Показать, как реализована валидация данных на сервере
* Показать, как реализованы группировка в запросе страницы администратора
* Как реализована функция выбора активной строки
* Показать запрос изменения/удаления данных на сервер
* Продемонстрировать ошибку БД срабатывания уникального ключа
* Продемонстрировать ошибку БД срабатывания check
* Продемонстрировать ошибку БД (ввели строку вместо числа или длинную строку)
* Показать функцию, где js формирует выпадающие списки справочных таблиц
* Продемонстрировать разное поведение страницы администратора в зависимости от роли пользователя
* Продемонстрировать добавление/изменение/удаление данных
* Т.д.

# Требования к БД

Минимум 6 таблиц:

* Роли
* Пользователи
* Роли пользователей
* 2 справочных таблицы
* Основная таблица (с внешними ключами на справочные)

## 1.1 Таблица Роли

Название таблицы: roles.

Таблица 1 – Роли. Структура

|  |  |  |  |
| --- | --- | --- | --- |
| **Название** | **Тип данных** | **NOT NULL** | **Примечание** |
| id | serial | Да | Первичный ключ |
| code | character varying(40) | Да | Уникальность на (code) |
| name | character varying(40) | Да | Уникальность на (name) |

Таблица 2 – Роли. Заполнение

|  |  |
| --- | --- |
| **Название** | **Код** |
| Пользователь | USR |
| Администратор | ADM |
| Модератор | MOD |

Заполнить таблицу нужно в БД, меняться эти данные не будут. При желании можно добавить еще ролей кроме указанных трех.

## 1.2 Таблица Пользователи

Название таблицы: users.

Таблица 1 – Пользователи. Структура

|  |  |  |  |
| --- | --- | --- | --- |
| **Название** | **Тип данных** | **NOT NULL** | **Примечание** |
| id | serial | Да | Первичный ключ |
| name | character varying(40) | Да |  |
| login | character varying(40) | Да | Уникальность на (login) |
| pass | character varying(40) | Да |  |

Таблица 2 – Пользователи. Заполнение

|  |  |  |
| --- | --- | --- |
| **Имя** | **Логин** | **Пароль** |
| Пользователь 1 | user1 | <задать самостоятельно> |
| Пользователь 2 | user2 | <задать самостоятельно> |
| Пользователь 3 | user3 | <задать самостоятельно> |
| <Ваше имя> | < задать самостоятельно > | <задать самостоятельно> |

Заполнить таблицу нужно в БД, меняться эти данные не будут. При желании можно добавить еще пользователей кроме указанных четырех. Также при желании можно реализовать шифрование паролей.

## 1.3 Таблица Роли пользователей

Название таблицы: user\_roles.

Таблица 1 – Роли пользователей. Структура

|  |  |  |  |
| --- | --- | --- | --- |
| **Название** | **Тип данных** | **NOT NULL** | **Примечание** |
| id | serial | Да | Первичный ключ |
| role\_id | integer | Да | Внешний ключ на role.id |
| user\_id | integer | Да | Внешний ключ на user.id |

Таблица 2 – Роли пользователей. Заполнение

|  |  |
| --- | --- |
| **Логин** | **Роли** |
| user1 | <задать самостоятельно> |
| user2 | <задать самостоятельно> |
| user3 | <задать самостоятельно> |
| user\_task1 | Администратор |

Заполнить таблицу нужно в БД, меняться эти данные не будут. Минимум у двух пользователей должно быть не меньше двух рохей

## 1.4 Справочные таблицы

Минимум 2 справочные таблицы. Название таблиц задать самостоятельно

Таблица 1 – Дополнительная таблица. Структура

|  |  |  |  |
| --- | --- | --- | --- |
| **Название** | **Тип данных** | **NOT NULL** | **Примечание** |
| id | serial | Да | Первичный ключ |
| name | character varying(50) | Да |  |

При желании можно добавить еще столбцы или изменить название name. При желании можно добавить ключи уникальности или другие внешние ключи

Примеры заполнения указаны в таблице 1.8. Необходимо придумать свои варианты из 4-10 записей для каждой из двух таблиц. Это могут быть статусы, причины, категории, географические понятия и т.д. Заполнить таблицу нужно в БД, меняться эти данные не будут.

Таблица 2 – Примеры заполнения

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **name** |  | **name** |  | **name** |  | **name** |
| Челябинск |  | В работе |  | В наличии |  | Тяжелый |
| Казань |  | На проверке |  | Забракован |  | Легкий |
| Уфа |  | Требует доработки |  | Нет в наличии |  | Средний |
| Екатеринбург |  | Решен |  | Ожидается |  | Неизвестно |
|  |  |  |  |  |  |  |
| **name** |  | **name** |  | **name** |  | **name** |
| мм |  | Упаковка |  | Металл |  | Процессор |
| см |  | Флакон |  | Дерево |  | ОЗУ |
| м |  | Тара |  | Пластик |  | SSD |
| км |  | Контейнер |  | Камень |  | Блок питания |

## 1.5 Основная таблица

Название таблицы: <задать самостоятельно>.

Таблица 1 – Основная таблица. Структура

|  |  |  |  |
| --- | --- | --- | --- |
| **Название** | **Тип данных** | **NOT NULL** | **Примечание** |
| id | serial | Да | Первичный ключ |
| <…> | integer | <…> | Внешний ключ на первую справочную таблицу |
| <…> | integer | <…> | Внешний ключ на вторую справочную таблицу |
| <…> | <…> | <…> |  |
| <…> | <…> | <…> |  |
| <…> | <…> | <…> | <…> |

Требования:

* Придумать названия и типы колонок самостоятельно. Суммарно требуется 6-8 колонок. Можно больше.
* Должны присутствовать как минимум типы данных integer и character varying(<N>). При желании можно тип даты и другие.
* Как минимум одно поле NOT NULL помимо первичного ключа.
* Как минимум одно ограничение уникальности помимо первичного ключа.
* Как минимум одно ограничение checks.

Заполнять основную таблицу значениями через БД напрямую не обязательно, разве что для тестирования и отладки. Работа с этой таблицей будет вестись в веб-приложении.

Ниже указан пример для функционала аптеки:

Таблица 2 – Пример. Справочная таблица. Единицы измерения

|  |  |
| --- | --- |
| units | |
| **id** | **name** |
| 1 | мг. |
| 2 | шт. |
| 3 | мл. |
| 4 | л |

Таблица 3 – Пример. Справочная таблица. Производитель

|  |  |
| --- | --- |
| producer | |
| **id** | **name** |
| 1 | ФармЛенд |
| 2 | Фабрика |
| 3 | ФарГроуп |
| 4 | МедДев |

Таблица 4 – Пример. Основная таблица. Структура

|  |  |  |  |
| --- | --- | --- | --- |
| **Название** | **Тип данных** | **NOT NULL** | **Примечание** |
| id | serial | Да | Первичный ключ |
| units\_id | integer | Да | Внешний ключ на units.id |
| producer\_id | integer | Нет | Внешний ключ на producer.id |
| name | character varying(50) | Да | Название препарата |
| num | integer | Да | Кол-во в упаковке |
| date\_create | date | Нет | Дата производства |

Ограничения:

* Ограничение уникальности на совокупность полей (units\_id, name, date\_create).
* Ограничение check (num > 0).

Таблица 4 – Пример. Основная таблица. Заполнение

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **units \_id** | **producer\_id** | **name** | **num** | **date\_create** |
| 1 | 1 | Сироп от кашля | 7 | 01-11-2024 |
| 1 | 2 | Парацетамол | 3 |  |
| 3 |  | Кальций Д3 | 5 | 03-10-2024 |
| 2 | 4 | Витамин С | 4 |  |

# Требования к WEB-интерфейсу и Серверной части

На рисунках указаны примеры, делать 1 в 1 по цвету, форме, расположению, названию элементов не нужно, реализовывать согласно своим макетам.

Представлена общая структура, что должно быть.

Как минимум, это:

* Страница авторизации пользователя
* Страница с таблицей основной таблицы, где можно вызвать удаление, изменение, добавление записей. На этой же странице реализовать поля для ввода значений для изменения и добавления новых записей.
* Страница администратора

## 2.1 Страница авторизации

Путь к странице: authorization.php

Название страницы: Авторизация

Доступна неавторизированным пользователям. Если пользователь уже авторизован, перенаправлять на /index.php.

На странице располагается окно для ввода логина и пароля с двумя полями и кнопкой «Ок». Окно должно располагаться по центру страницы (по вертикали и горизонтали).

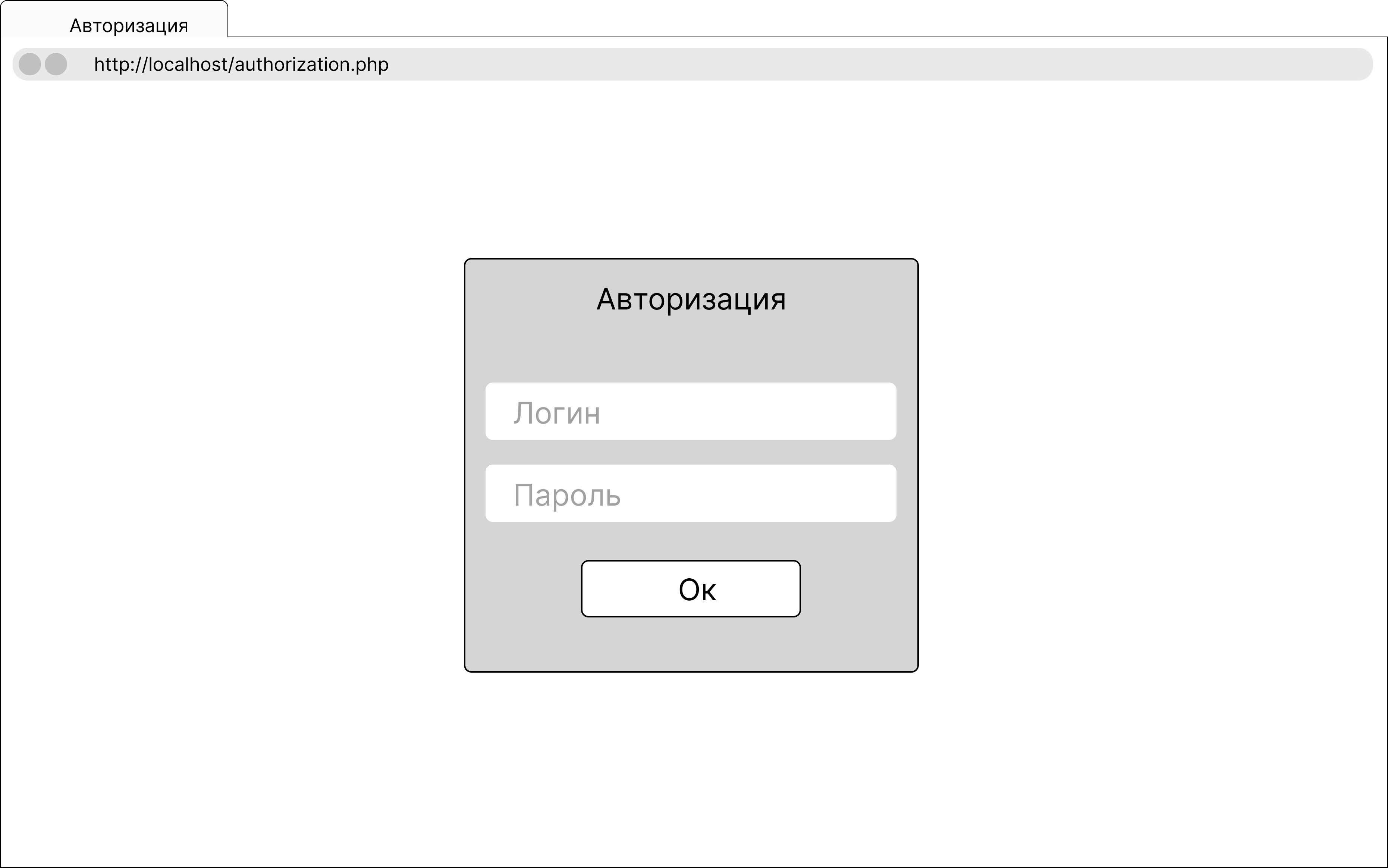


Рисунок 2.1 – Пример страницы авторизации

При нажатии на «Ок» идет запрос к серверу. Тип запроса: POST. Если логин и пароль совпадают, то сохранить необходимые для дальнейшей логики данные в сессию и перенаправить пользователя на страницу /index.php.

Если пароль или логин не совпадают, то вывести сообщение с ошибкой «Неправильный логин или пароль». Отправлять запрос на сервер можно как при помощи fetch-запроса (рисунок 2.2), так и при помощи запроса с перезагрузкой страницы (рисунок 2.3). Выбрать любой способ на свое усмотрение.

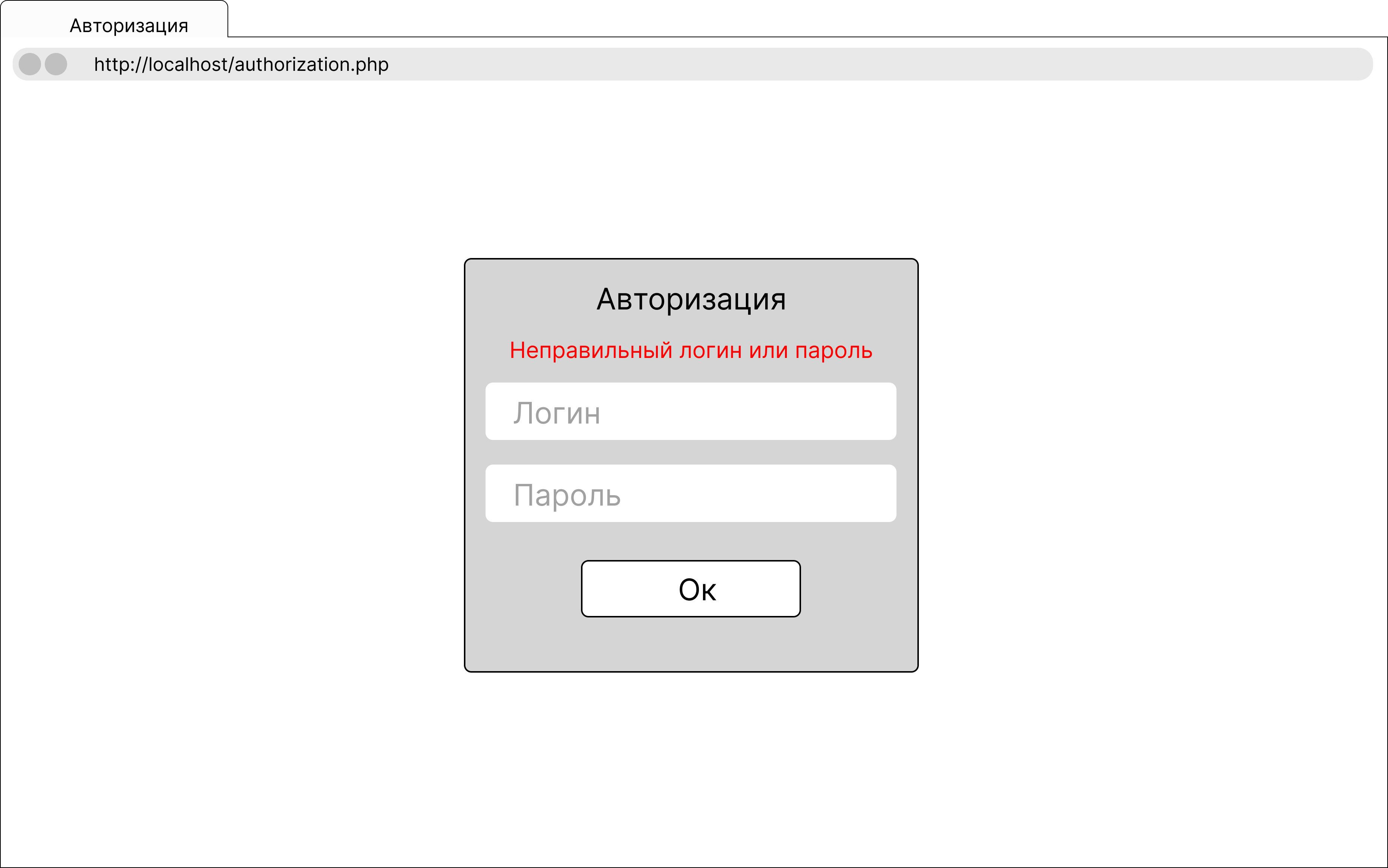


Рисунок 2.2 – Пример отправки логина и пароля при помощи fetch-запроса  
 (без перезагрузки страницы)

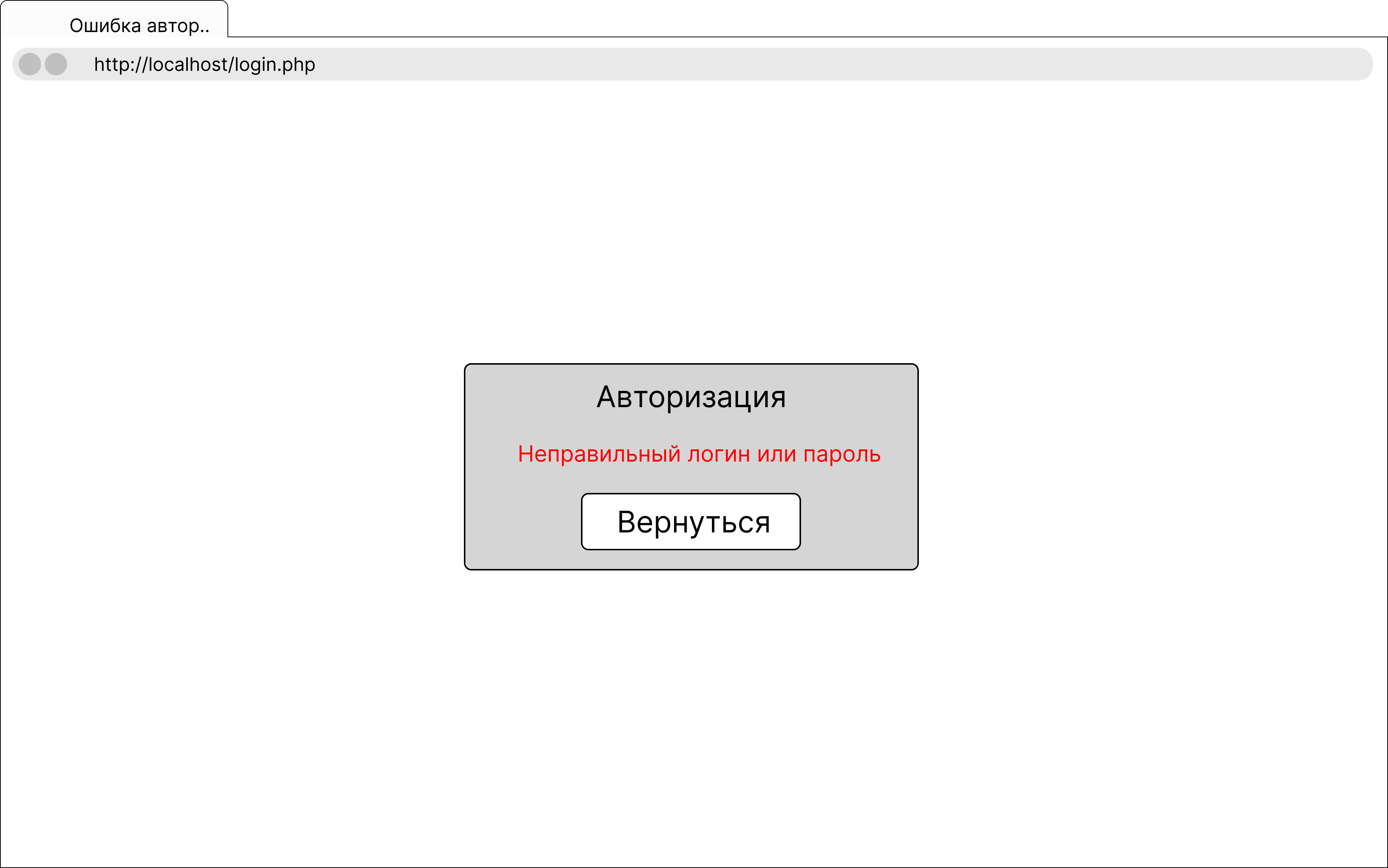


Рисунок 2.3 – Пример отправки логина и пароля при помощи запроса в <form>

## 2.2 Основная страница

Путь к странице: index.php

Доступна авторизированным пользователям. Если пользователь не авторизирован, перенаправлять на /authorization.php

Любое взаимодействие с сервером на данной странице не предполагает ее перезагрузку (кроме кнопки «Выход»). Используется клиентский рендеринг содержимого страницы при помощи fetch запросов к серверу.



Рисунок 2.4 – Пример внешнего вида страницы

* При нажатии на «Добавить» появляются поля ввода информации основной таблицы (рисунок 2.5). Заголовок этого окна «Добавление новой записи».
* При нажатии на «Отмена» окно становится пустым.
* При нажатии на «Добавить» на сервер отправляется fetch-запрос. Если есть ошибки – они выводятся пользователю в alert (рисунок 2.6).
* Если данные валидны и ошибок нет, то обновить данные в таблице, окно добавления становится пустым (рисунок 2.7).
* Выбор элементов справочных таблиц должен осуществляться через **выпадающие списки**. Данные получать через сервер по fetch и формировать эти выпадающие списки через js.

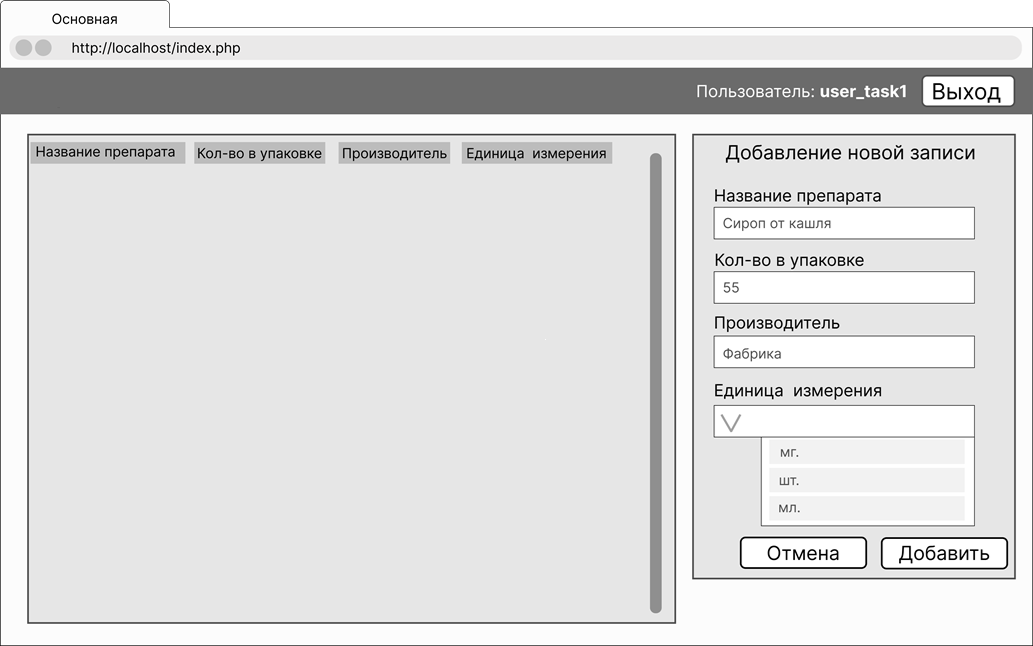


Рисунок 2.5 – Пример добавления данных

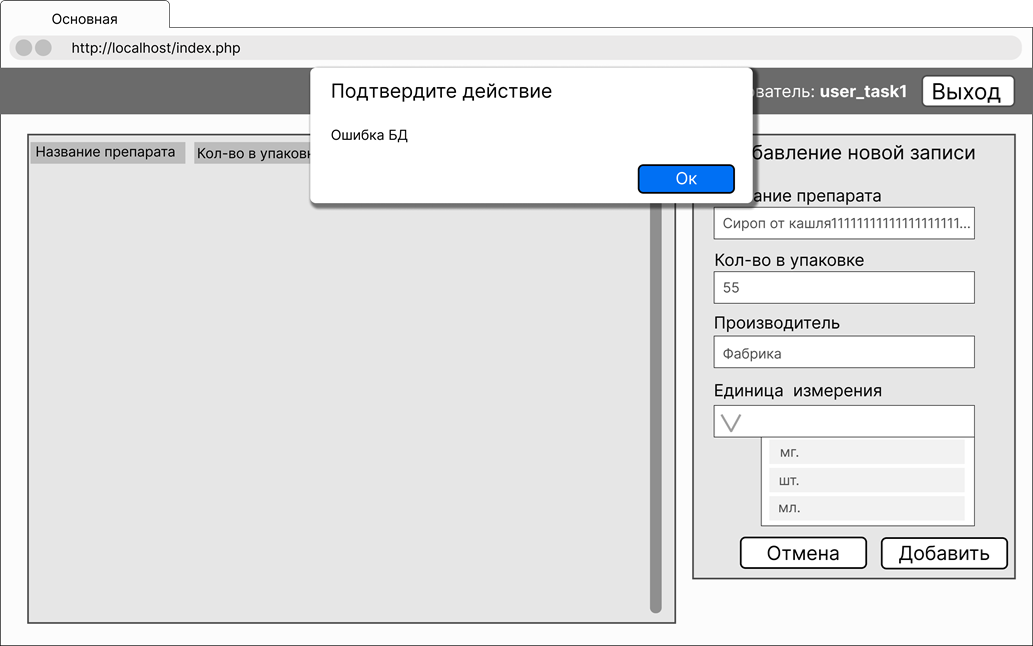


Рисунок 2.6 – Вывод ошибки пользователю

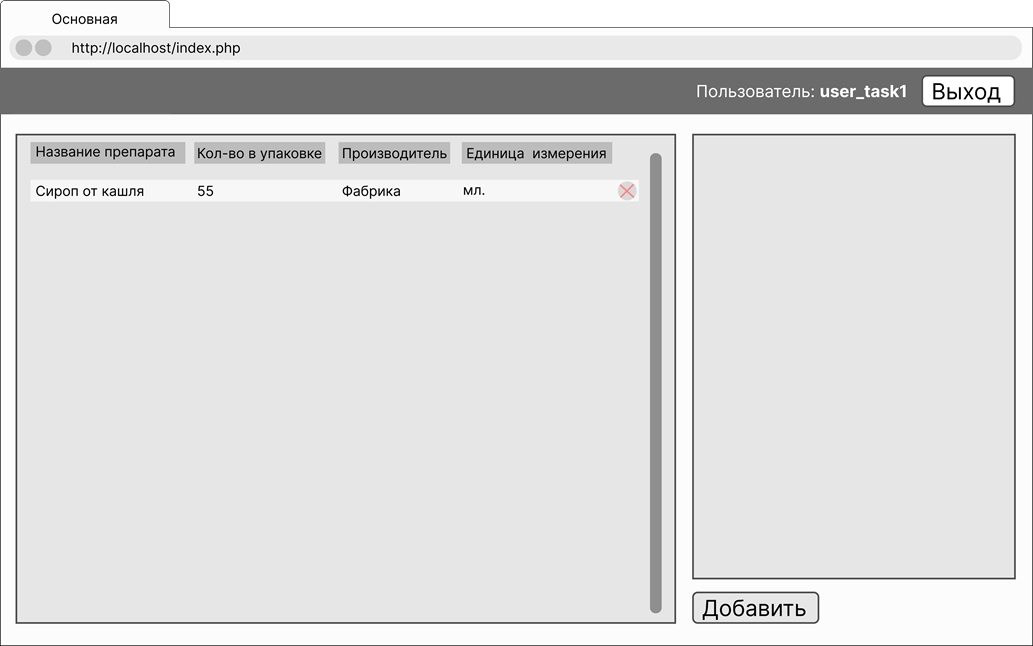


Рисунок 2.7 – Пример успешного добавления данных (поля ввода исчезли)

При нажатии на элемент таблицы он должен стать активным, и строка таблицы выделяется цветом. В окно подтягиваются данные выбранной записи. Название окна «Изменение записи» (Рисунок 2.8)

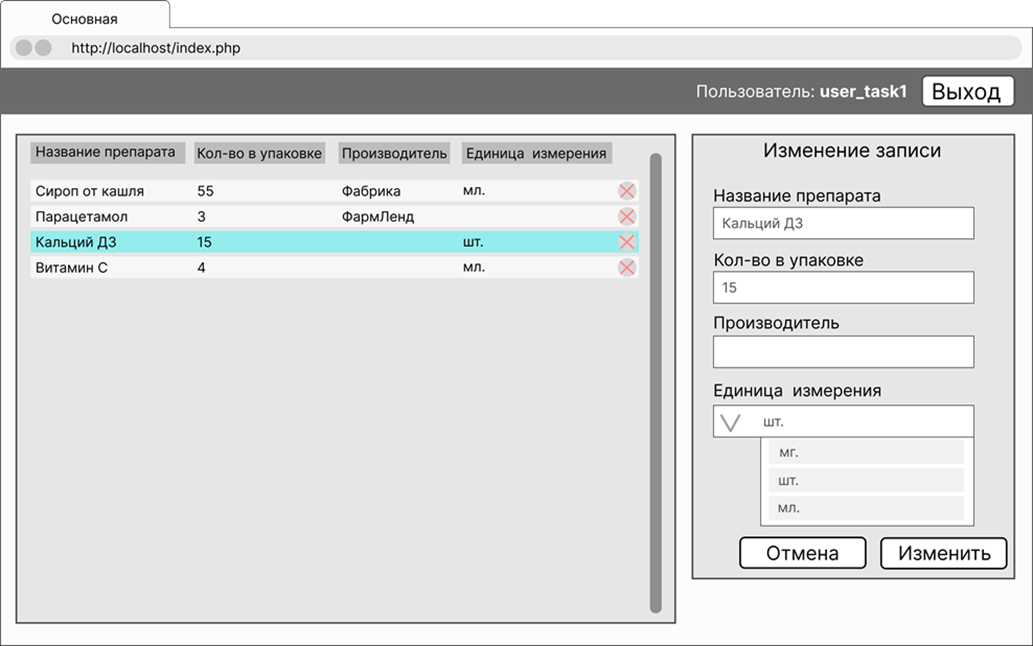


Рисунок 2.8 – Пример выбора строки и изменения данных

При нажатии на «Отмена» окно становится пустым, активная строка пропадает (рисунок 2.9).

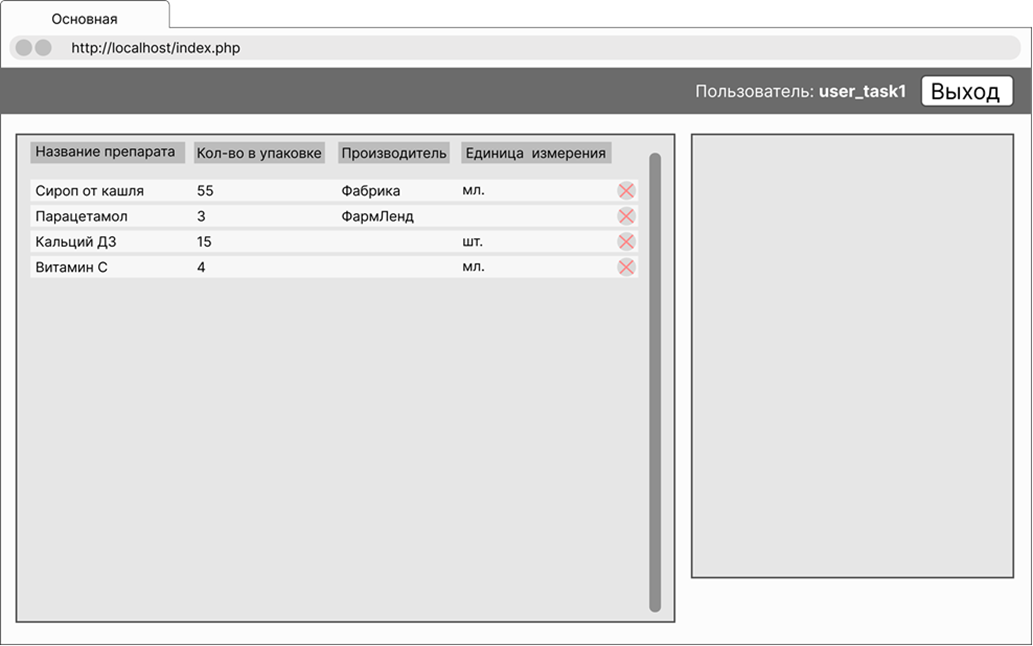


Рисунок 2.9 – Нет выделения строки

При нажатии на «Изменить» на сервер отправляется fetch-запрос. Если есть ошибки – они выводятся пользователю в alert.

Если данные валидны и ошибок нет, то обновить данные в таблице, окно добавления становится пустым, выделение активной строки пропадает (рисунок 2.9).

Удаление:

* При нажатии на иконку удаления  пользователю выводится сообщение (confirm) «Вы действительно хотите удалить запись?» (Рисунок 2.10).
* Если пользователь нажал «Ок», то на сервер отправляется запрос на удаление, таблица обновляется.
* Если удаляемая запись перед этим была выбрана и изменялась в окне справа, то окно должно стать пустым.

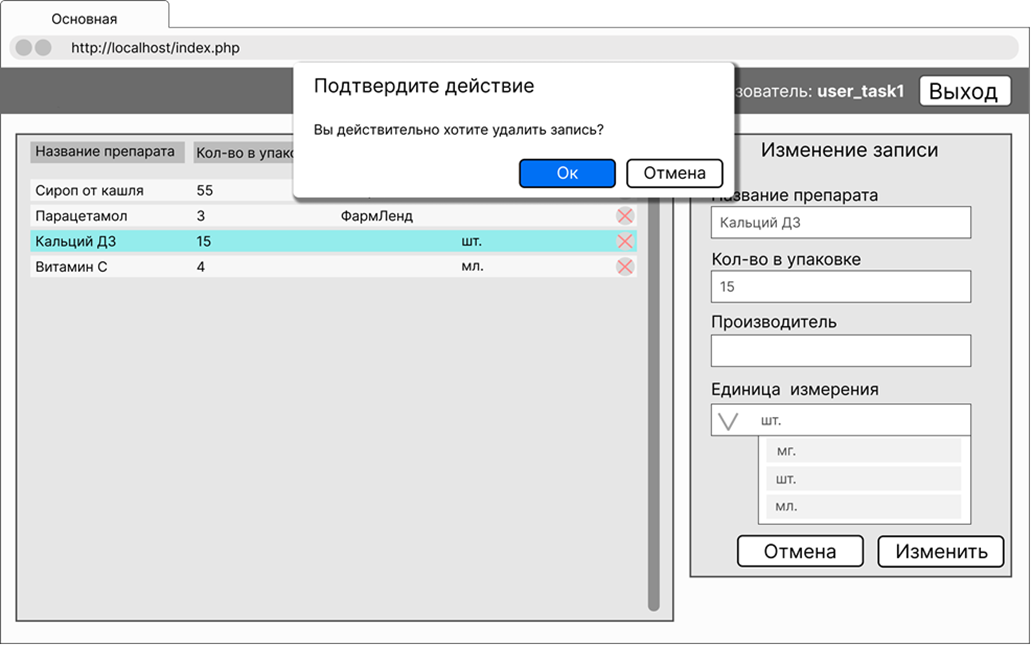


Рисунок 2.10 – Пример удаления записи

## 2.3 Требование к валидации данных

Минимум одно поле при добавлении и изменении должно быть подлежать валидации на сервере. Это может быть проверка на пустоту, проверка на тип, на равенство конкретному значению, больше/меньше и т.д. Ошибку необходимо вывести пользователю. Пример на рисунке 2.11.

Дополнительная валидация на стороне клиента – по желанию.

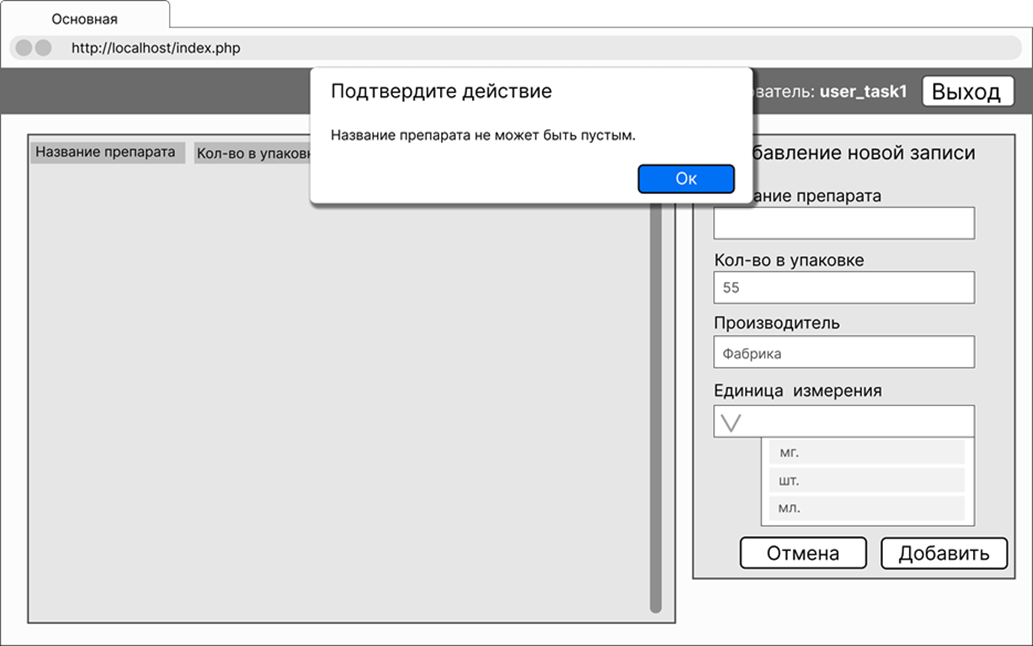


Рисунок 2.11 – Валидация данных

## 2.4 Страница администратора

Путь к странице: admin.php

Доступ по ролям: Администратор

Доступна авторизированным пользователям. Если пользователь не авторизирован, перенаправлять на /authorization.php

Страница формируется полностью на сервере методами PHP.

Сначала выводить список ролей пользователя по связи:

role.name, где role.id = user\_roles.role\_id и user\_roles.user\_id = user\_id из сессии пользователя. Сортировка по role.name по алфавиту. Можно сохранить роли в сессию сразу при авторизации.

Каждая роль выводится с новой строки, выводить в формате «- Название роли». Если название роли = «Администратор», то вывести жирным.

Далее выводить таблицу со всеми пользователями и их ролями. Стиль таблицы произвольный, главное – видимые границы строк и столбцов. В БД использовать запрос с группировкой. Должно быть 3 столбца: Логин, Имя и Роли. В столбец «Роли» вывести название всех ролей данного пользователя через запятую. Список должен совпасть со требованиями к БД. Сортировать на выбор либо по логину, либо по имени.

Далее вывести таблицу с двумя столбцами: Поле и Кол-во. Стиль таблицы произвольный, главное – видимые границы строк и столбцов. В данную таблицу вывести список записей name любой справочной таблицы и кол-во записей из основной таблицы по внешнему ключу. Использовать запрос с группировкой. Сортировать по столбику «Кол-во».

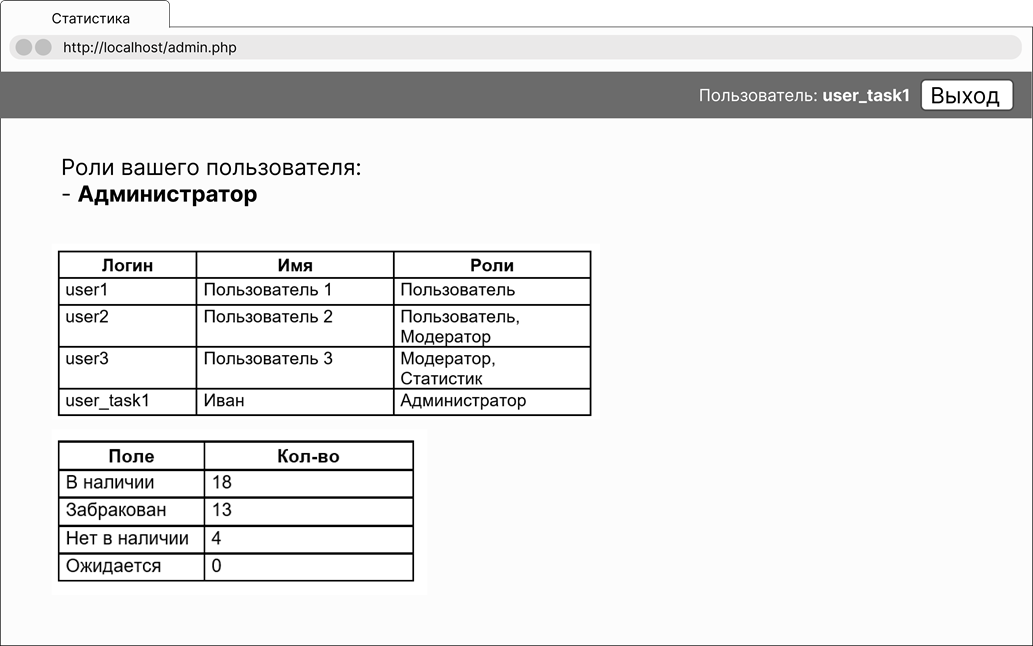


Рисунок 2.13 – Пример макета страницы

Если у пользователя нет нужной роли для доступа к admin.php, то выводить список ролей пользователя и сообщение «У вас недостаточно привилегий для просмотра данной страницы»

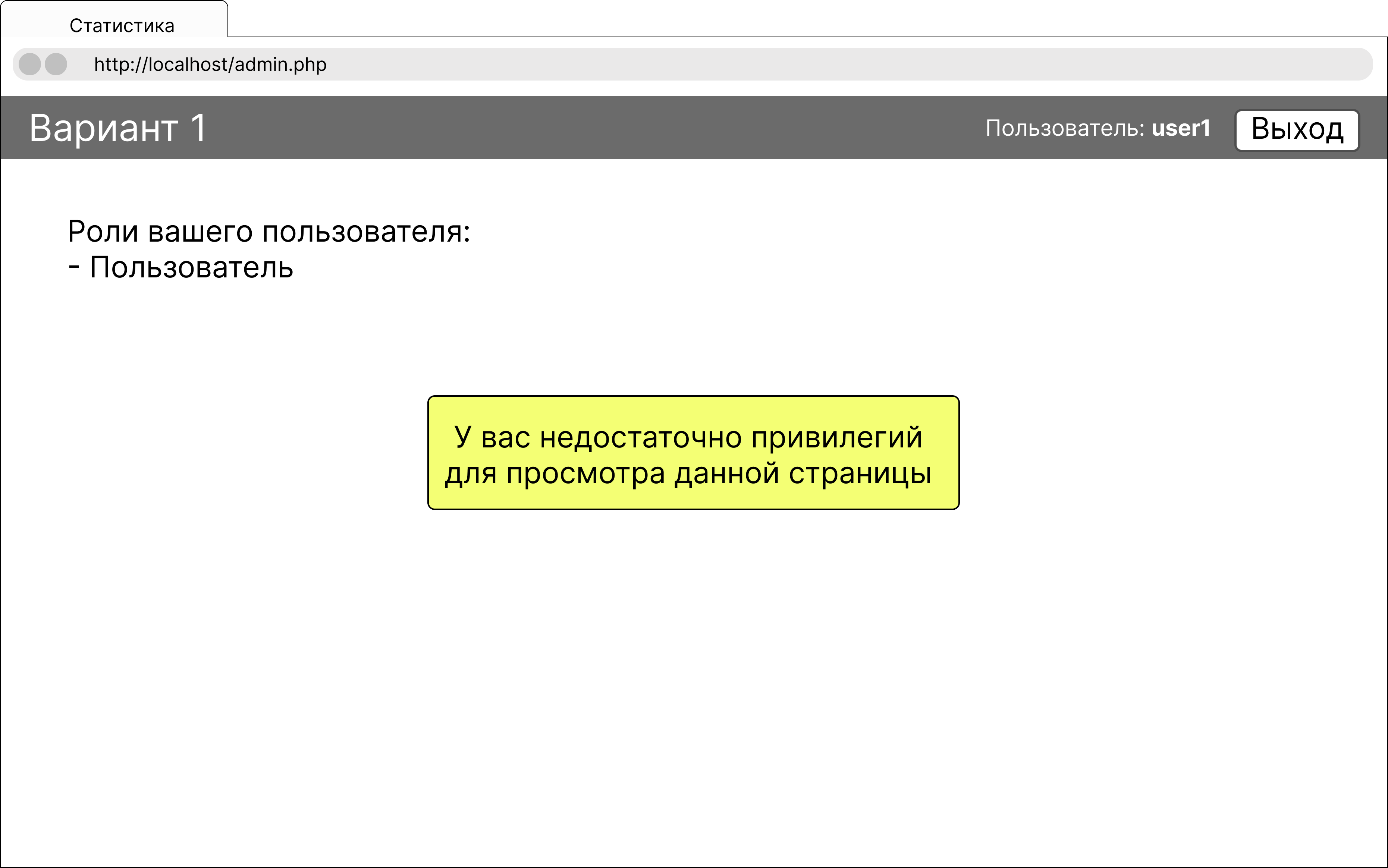


Рисунок 2.15 – Пример вывода недостаточно привилегий

## Шапка страницы

На страницах index.php и admin.php присутствует шапка в самом верху страницы.

Ширина – 100% страницы.

Справа выводиться логин пользователя user.name (сохранить в сессию и доставать оттуда) с жирным выделением.

Кнопка «Выход» сбрасывает сессию пользователя и перенаправляет на authorization.php (рисунок 2.16).

На странице authorization.php шапку выводить не нужно.

Стиль шапки придумать самостоятельно



Рисунок 2.16 – Шапка страницы