МИНИСТЕРСТВО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

МЕЖДУНАРОДНЫЙ ИНСТИТУТ КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ

ФАКУЛЬТЕТ ИНФОРМАЦИОННЫХ И ЭНЕРГЕТИЧЕСКИХ СИСТЕМ

КАФЕДРА информатики и вычислительной техники

КУРСОВОЙ ПРОЕКТ

по дисциплине " БАЗЫ ДАННЫХ"

ТЕМА: Программное приложение "Склад товара"

Рассчетно – пояснительная записка

Разработал студент гр.ИВТо-211 /Василенко Н.А. /

Руководитель /Ганцева Е.А. /

подпись,дата инициалы,фамилия

Нормоконтролер / /

подпись,дата инициалы,фамилия

Защищен \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

дата

Воронеж 2023

МЕЖДУНАРОДНЫЙ ИНСТИТУТ КОМПЬЮТЕРНЫХ

ТЕХНОЛОГИЙ

ФАКУЛЬТЕТ ИНФОРМАЦИОННЫХ СИСТЕМ

КАФЕДРА ИНФОРМАТИКИ И ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ

**ЗАДАНИЕ**

на курсовой проект

по дисциплине "Базы данных"

Тема работы: Программное приложение "Склад товара"

Студент группы ИВТо-211 Василенко Назар Александрович

фамилия, имя, отчество

Технические условия: Среда разработки программного обеспечения «Microsoft Visual Studio, Система управления реляционными базами данных «Microsoft SQL Server».

Содержание и объем проекта (графические работы, расчеты и прочее):

раcчетно - пояснительная записка – 33 страницы формата А4; поясняющий текст, 9 рисунков и 11 таблиц).

Срок защиты курсового проекта: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель / Ганцева Е.А. /

подпись, дата фамилия, инициалы

Задание принял студент гр.ИВТо-211 /Василенко Н.А. /

подпись, дата фамилия, инициалы

# **Замечания руководителя**

Содержание

**1. Программное обеспечение для курсового проекта**  
1.1 **MySQL**

* История развития MySQL
* Основные возможности и архитектура MySQL  
  1.2 **PHP и серверная разработка**
* Назначение PHP для взаимодействия с базой данных  
  1.3 **HTML/CSS/JS для клиентской части**
* Роль фронтенда в проекте

**2. Структура базы данных**  
2.1 **Описание основных таблиц**

* Таблица customers (Клиенты)
* Таблица equipment (Спортивный инвентарь)
* Таблица orders (Заказы)
* Таблица order\_items (Товары в заказах)
* Таблица payments (Платежи)  
  2.2 **Связи между таблицами**
* Диаграмма базы данных (ER-диаграмма)

**3. Реализация программы**  
3.1 **Элементы управления веб-приложения**

* Общий вид страницы
* Интерфейс просмотра инвентаря
* Управление заказами
* Работа с клиентами и платежами  
  3.2 **Функции программы**
* Добавление новых записей (товаров, клиентов, заказов)
* Удаление и редактирование записей  
  3.3 **Пример PHP-кода для работы с базой данных**
* Подключение к базе данных
* Выполнение SQL-запросов (SELECT, INSERT, UPDATE, DELETE)

**4. Пример фронтенд-кода на HTML/CSS/JS**

* Создание интерфейса для работы с таблицами

**Заключение**

Список литературы

### Введение

В настоящее время информационные технологии занимают важное место в различных сферах деятельности. Особо важную роль они играют в автоматизации учета и управления различными ресурсами. Создание и использование баз данных позволяет существенно упрощать эти процессы, предоставляя удобные инструменты для хранения, обработки и анализа данных.

Курсовая работа по дисциплине «Базы данных» посвящена теме автоматизации проката спортивного инвентаря. Данная тема была выбрана вследствие актуальности разработки эффективных информационных систем, которые облегчают учет оборота и управление ассортиментом спортивного оборудования.

В работе будут рассмотрены вопросы создания и администрирования базы данных, основанной на системе MySQL. Для реализации веб-интерфейса будет использоваться PHP, что позволит обеспечить интерактивное взаимодействие пользователей с системой.

Цель данной работы — разработка информационной системы для учета проката спортивного инвентаря, обеспечивающей удобное хранение данных, автоматизацию рутинных процедур и мониторинг работы системы.

Программное обеспечение для курсового проекта

В работе будут рассмотрены вопросы создания и администрирования базы данных, основанной на системе MySQL. MySQL является одной из самых популярных систем управления базами данных с открытым исходным кодом. Она обеспечивает надежное хранение данных, высокую производительность и масштабируемость. MySQL поддерживает язык структурированных запросов SQL, который предоставляет широкие возможности для работы с данными, включая их выборку, фильтрацию, обновление и удаление. Система широко применяется как в небольших проектах, так и в крупных корпоративных приложениях благодаря своей гибкости и возможности интеграции с другими технологиями.

Для реализации веб-интерфейса будет использоваться PHP, один из наиболее популярных языков серверного программирования. PHP обладает высокой совместимостью с MySQL, что делает его оптимальным выбором для разработки динамических веб-приложений. Этот язык предоставляет множество встроенных функций для работы с базами данных, упрощая выполнение таких задач, как подключение к базе данных, выполнение SQL-запросов и обработка результатов. Кроме того, PHP поддерживает различные веб-фреймворки, которые ускоряют процесс разработки и повышают качество создаваемых приложений.

### **1.1 История развития MySQL**

MySQL — это одна из самых популярных систем управления базами данных с открытым исходным кодом. Она была разработана в середине 1990-х годов компанией MySQL AB, основанной **Майклом Видениусом**, **Дэвидом Аксмарком** и **Алленом Ларссоном**. Первая версия MySQL появилась в **1995 году**, и с тех пор система продолжает активно развиваться.

На начальных этапах разработки MySQL была задумана как быстрая, надёжная и простая система для работы с реляционными базами данных, которая поддерживает **язык SQL**. Её популярность быстро росла благодаря двум важным факторам:

1. Открытый исходный код, который позволял разработчикам использовать и модифицировать систему бесплатно.
2. Высокая производительность и эффективность при обработке больших объёмов данных.

В **2008 году** компания MySQL AB была приобретена **Sun Microsystems**, а затем в **2010 году** Sun перешла под контроль **Oracle Corporation**, которая в настоящее время поддерживает и развивает MySQL.

### **Этапы развития MySQL:**

1. **1995 год** — Выпуск первой версии MySQL (версии 1.0).
2. **2000 год** — Появление версии 3.23 с поддержкой транзакций и улучшенной производительностью.
3. **2003 год** — Выпуск версии 4.0, добавляющей кэширование запросов.
4. **2005 год** — Версия 5.0, где появилась поддержка хранимых процедур, триггеров и представлений.
5. **2010 год** — Oracle выпускает версию 5.5 с улучшенной многопоточностью.
6. **2013 год** — Выпуск MySQL 5.6 с поддержкой репликации и оптимизацией запросов.
7. **2015 год** — MySQL 5.7 с улучшенной производительностью и JSON-типами данных.
8. **2018 год** — MySQL 8.0, в которой была внедрена поддержка CTE (общих табличных выражений), улучшенная безопасность и расширенные возможности индексов.

На данный момент MySQL активно развивается и используется в миллионах проектов по всему миру, включая крупные веб-приложения и корпоративные системы.

### **1.2 Основные возможности и архитектура MySQL**

MySQL является реляционной СУБД, в которой данные организованы в виде таблиц, состоящих из строк и столбцов. Основными возможностями MySQL являются:

1. **Высокая производительность и масштабируемость**  
   MySQL обеспечивает быструю обработку запросов даже на больших объёмах данных благодаря оптимизированным механизмам индексации и кэширования.
2. **Поддержка стандартного SQL**  
   MySQL поддерживает язык структурированных запросов (SQL), что позволяет легко взаимодействовать с данными — выполнять **SELECT**, **INSERT**, **UPDATE**, **DELETE** и другие операции.
3. **Поддержка транзакций**  
   MySQL поддерживает ACID-транзакции (атомарность, согласованность, изолированность и надёжность), что обеспечивает целостность данных даже при сбоях.
4. **Репликация данных**  
   MySQL позволяет создавать резервные копии и дублировать базы данных для повышения отказоустойчивости.
5. **Гибкая архитектура хранилищ**  
   MySQL предоставляет несколько **движков хранения данных**:
   * **InnoDB** — основной движок, поддерживающий транзакции и внешние ключи.
   * **MyISAM** — старый движок, обеспечивающий высокую производительность на чтение.
   * **Memory** — хранение данных в оперативной памяти для быстрого доступа.
   * **CSV, Archive, NDB и другие** — специализированные движки для различных сценариев использования.
6. **Безопасность данных**  
   MySQL поддерживает аутентификацию пользователей, шифрование соединений и гранулярное управление правами доступа.
7. **Масштабируемость и высокая доступность**  
   MySQL легко интегрируется с кластерными решениями для масштабирования и обеспечения отказоустойчивости.
8. **Поддержка JSON и NoSQL-функционала**  
   В версии MySQL 8.0 добавлена возможность хранения и обработки данных в формате JSON, что позволяет использовать MySQL в современных веб-приложениях.

### **1.2 PHP и серверная разработка**

**PHP** (Hypertext Preprocessor) — это популярный язык программирования для создания динамических веб-приложений и взаимодействия с базами данных. Он выполняется на стороне сервера и позволяет обрабатывать запросы пользователей, управлять данными и передавать результаты на клиентскую сторону. В данном проекте PHP используется для взаимодействия с базой данных MySQL и реализации бизнес-логики системы проката спортивного инвентаря.

#### **Назначение PHP для взаимодействия с базой данных**

1. **Подключение к базе данных**  
   PHP предоставляет встроенные функции и расширения для работы с MySQL (например, **mysqli** и **PDO**). С их помощью можно установить соединение с сервером баз данных, а также безопасно выполнять запросы.
2. **Выполнение операций с базой данных**  
   Через PHP можно отправлять SQL-запросы к базе данных:
   * **SELECT** — для извлечения данных (например, списка доступного инвентаря);
   * **INSERT** — для добавления новых записей (например, новых заказов или клиентов);
   * **UPDATE** — для обновления данных (например, изменения статуса заказа);
   * **DELETE** — для удаления записей (например, отмены заказа).
3. **Обработка данных**  
   PHP позволяет обрабатывать результаты SQL-запросов и форматировать их для передачи на клиентскую часть. Например, можно преобразовать данные в формат **JSON**, который легко обрабатывается с помощью JavaScript на фронтенде.
4. **Валидация и безопасность**  
   Для обеспечения надёжной работы системы PHP обрабатывает и валидирует данные от пользователей, предотвращая ошибки и потенциальные угрозы, такие как **SQL-инъекции**. Для этого используются подготовленные запросы (**Prepared Statements**) и фильтрация данных.
5. **Реализация бизнес-логики**  
   В проекте PHP реализует всю логику приложения, такую как:
   * Регистрация и авторизация пользователей;
   * Обработка заказов (добавление, редактирование, просмотр);
   * Управление платежами;
   * Контроль статусов и доступности инвентаря.

Таким образом, PHP играет ключевую роль в проекте как серверный язык, обеспечивающий связь между базой данных и пользовательским интерфейсом, а также обрабатывающий запросы и реализующий логику системы.

### **1.3 HTML/CSS/JS для клиентской части**

**HTML**, **CSS** и **JavaScript** — это основные технологии для создания и управления пользовательским интерфейсом веб-приложений. Они позволяют организовать удобное и интуитивно понятное взаимодействие пользователя с системой проката спортивного инвентаря.

#### **Роль фронтенда в проекте**

1. **HTML (HyperText Markup Language)**  
   HTML используется для создания структуры веб-страниц. В данном проекте с помощью HTML формируются основные элементы интерфейса:
   * Форма для регистрации и входа пользователей;
   * Таблицы для отображения списка доступного инвентаря и заказов;
   * Кнопки и ссылки для управления данными (добавление, редактирование, удаление).
2. **CSS (Cascading Style Sheets)**  
   CSS отвечает за внешний вид веб-приложения. Оно используется для стилизации интерфейса, чтобы сделать его более привлекательным и удобным для пользователя. В проекте CSS выполняет следующие функции:
   * Определяет цвета, шрифты и отступы;
   * Создаёт адаптивный дизайн, чтобы приложение корректно отображалось на разных устройствах;
   * Улучшает визуальное восприятие данных через оформление таблиц, кнопок и форм.
3. **JavaScript (JS)**  
   JavaScript добавляет интерактивность и динамичность веб-приложению. В проекте JS выполняет следующие задачи:
   * **Обновление данных без перезагрузки страницы** (с использованием AJAX/Fetch API);
   * Валидация данных на клиентской стороне (например, проверка полей форм перед отправкой на сервер);
   * Обработка событий (например, кликов по кнопкам для добавления или удаления записей);
   * Улучшение пользовательского опыта за счёт динамического отображения данных, анимаций и уведомлений.

#### **Примеры интерфейсов фронтенда**

В рамках проекта будут реализованы следующие ключевые интерфейсы:

* **Страница просмотра инвентаря**: Таблица с информацией о доступном спортивном инвентаре.
* **Форма добавления заказа**: Поля для выбора инвентаря, указания количества и сроков проката.
* **Личный кабинет клиента**: Данные пользователя и список его заказов.
* **Форма регистрации и авторизации**: Поля для ввода логина и пароля.

Таким образом, **HTML**, **CSS** и **JavaScript** обеспечивают создание удобного и функционального пользовательского интерфейса, который взаимодействует с сервером через PHP и предоставляет пользователю доступ к данным системы проката.

### 1. **Таблица** equipment (Инвентарь)

Хранит информацию о спортивном инвентаре.

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| equipment\_id | INT (PK, AUTO\_INCREMENT) | Уникальный идентификатор инвентаря |
| name | VARCHAR(255) | Название инвентаря |
| category | VARCHAR(255) | Категория (например, лыжи, велосипеды) |
| size | VARCHAR(50) | Размер (если применимо) |
| price\_per\_day | DECIMAL(10, 2) | Стоимость аренды за день |
| quantity | INT | Доступное количество |
| status | ENUM('available', 'out\_of\_stock', 'repair') | Статус инвентаря |

### 2. **Таблица** customers (Клиенты)

Хранит данные о клиентах.

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| customer\_id | INT (PK, AUTO\_INCREMENT) | Уникальный идентификатор клиента |
| first\_name | VARCHAR(255) | Имя |
| last\_name | VARCHAR(255) | Фамилия |
| phone | VARCHAR(15) | Телефон |
| email | VARCHAR(255) | Электронная почта |
| address | VARCHAR(255) | Адрес |

### 3. **Таблица** orders (Заказы)

Хранит информацию о заказах.

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| order\_id | INT (PK, AUTO\_INCREMENT) | Уникальный идентификатор заказа |
| customer\_id | INT (FK) | Идентификатор клиента |
| order\_date | DATETIME | Дата оформления заказа |
| return\_date | DATETIME | Ожидаемая дата возврата |
| total\_price | DECIMAL(10, 2) | Общая стоимость заказа |
| status | ENUM('active', 'completed', 'canceled') | Статус заказа |

### 4. **Таблица** order\_items (Состав заказа)

Хранит информацию о конкретных единицах инвентаря, входящих в заказ.

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| order\_item\_id | INT (PK, AUTO\_INCREMENT) | Уникальный идентификатор позиции |
| order\_id | INT (FK) | Идентификатор заказа |
| equipment\_id | INT (FK) | Идентификатор инвентаря |
| quantity | INT | Количество |
| price | DECIMAL(10, 2) | Стоимость за единицу |

### 5. **Таблица** payments (Платежи)

Хранит данные об оплате.

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| payment\_id | INT (PK, AUTO\_INCREMENT) | Уникальный идентификатор платежа |
| order\_id | INT (FK) | Идентификатор заказа |
| payment\_date | DATETIME | Дата платежа |
| amount | DECIMAL(10, 2) | Сумма оплаты |
| payment\_method | ENUM('cash', 'card', 'online') | Способ оплаты |

### Связи между таблицами:

1. **Связь между customers и orders**: Один клиент может иметь много заказов.
2. **Связь между orders и order\_items**: Один заказ состоит из нескольких позиций.
3. **Связь между order\_items и equipment**: Каждая позиция заказа связана с определенным инвентарем.
4. **Связь между orders и payments**: Один заказ может быть оплачен несколькими платежами (например, предоплата и доплата).

### **2.2 Диаграмма базы данных**

На диаграмме представлена структура базы данных для системы проката спортивного инвентаря. Она включает в себя следующие основные таблицы

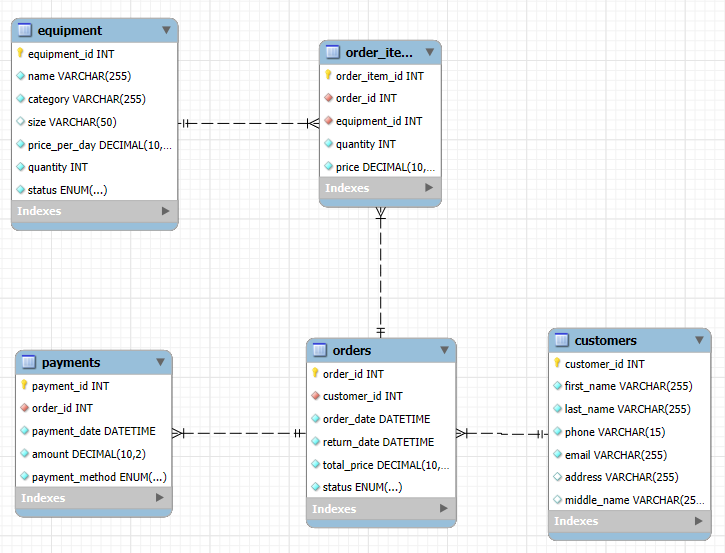


Рисунок №. Диаграмма базы данных

Общий вид страницы

На изображении представлена **административная панель** для управления заказами в системе проката спортивного инвентаря. Интерфейс разделён на несколько функциональных блоков для удобного взаимодействия с данными.

1. **Панель навигации**  
   В верхней части располагаются кнопки:
   * **Добавление заказа** (активная вкладка);
   * **Добавление товара**;
   * **Добавление клиента**.  
     Эти вкладки позволяют переключаться между основными разделами административной панели.
2. **Поиск по заказам**  
   Поле для ввода **ФИО клиента** и кнопка **«Найти»** позволяют быстро отфильтровать заказы по фамилии клиента.
3. **Добавление нового заказа**  
   Блок содержит форму для внесения новых заказов с полями:
   * **Имя клиента**, **Фамилия клиента**, **Отчество клиента**;
   * **Выбор товара** из выпадающего списка;
   * **Количество** единиц товара;
   * **Дата заказа** и **дата возврата** (с использованием календаря для удобного выбора дат);
   * **Общая стоимость** заказа.  
     После заполнения всех полей нажатием на кнопку **«Добавить заказ»** данные сохраняются в базе данных.
4. **Список заказов**  
   В нижней части страницы представлен **табличный список заказов**, включающий столбцы:
   * **ID** — уникальный идентификатор заказа;
   * **ФИО клиента**;
   * **Дата заказа** и **дата возврата**;
   * **Стоимость** — общая сумма заказа;
   * **Товар** — наименование прокатного инвентаря;
   * **Количество** единиц товара.
   * **Действия** — кнопка **«Удалить»** для удаления заказа из базы данных.

Такой интерфейс обеспечивает удобное управление заказами, добавление новых записей и мониторинг текущих заказов, что делает работу администратора простой и эффективной.

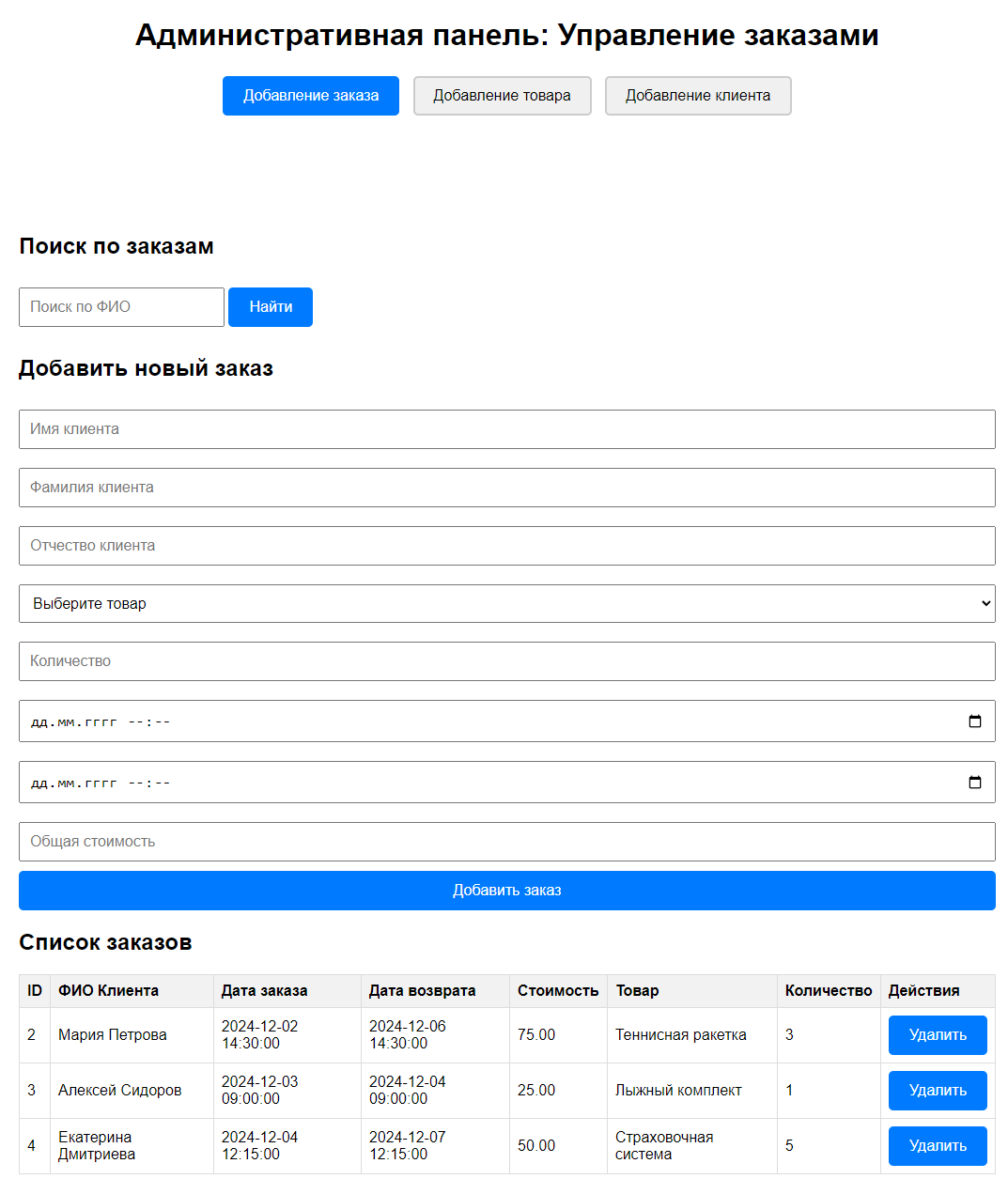


Рисунок №. Общий вид страницы

Интерфейс управления товарами

На данной странице представлена функциональность для управления **спортивным инвентарём** в системе проката. Интерфейс разделён на две основные секции: **добавление нового товара** и **список существующих товаров**.

#### **Добавление нового товара**

В верхней части страницы расположена форма для добавления новых единиц спортивного инвентаря. Поля формы включают:

* **Название товара** — наименование спортивного инвентаря (например, «Горный велосипед»);
* **Категория** — принадлежность товара к определённой группе (например, «Велосипеды», «Зимний спорт»);
* **Размер** — габариты товара (например, «Средний», «Большой»);
* **Цена за день** — стоимость аренды товара в сутки;
* **Количество** — доступное количество единиц товара;
* **Статус** — текущее состояние товара (доступный, отсутствующий, на ремонте).

После заполнения всех полей администратор может нажать кнопку **«Добавить товар»**, чтобы сохранить информацию в базе данных.

#### **Список товаров**

В нижней части страницы отображается таблица с перечнем доступного спортивного инвентаря. Каждая строка содержит следующую информацию:

* **ID** — уникальный идентификатор товара;
* **Название** — наименование инвентаря;
* **Категория** — группа, к которой принадлежит товар;
* **Размер** — размеры или габариты инвентаря;
* **Цена за день** — стоимость проката одной единицы товара в сутки;
* **Количество** — доступное количество единиц инвентаря на складе;
* **Статус** — текущее состояние товара:
  + **available** — доступен для проката;
  + **out\_of\_stock** — временно отсутствует;
  + **repair** — находится на ремонте.

### **Назначение страницы**

Страница позволяет администраторам:

1. Добавлять новые товары в базу данных с учётом всех параметров.
2. Просматривать текущий список инвентаря, контролировать его количество и состояние.
3. Оперативно управлять запасами для обеспечения бесперебойной работы сервиса проката.

Такой интерфейс обеспечивает простоту и удобство управления складом спортивного инвентаря.

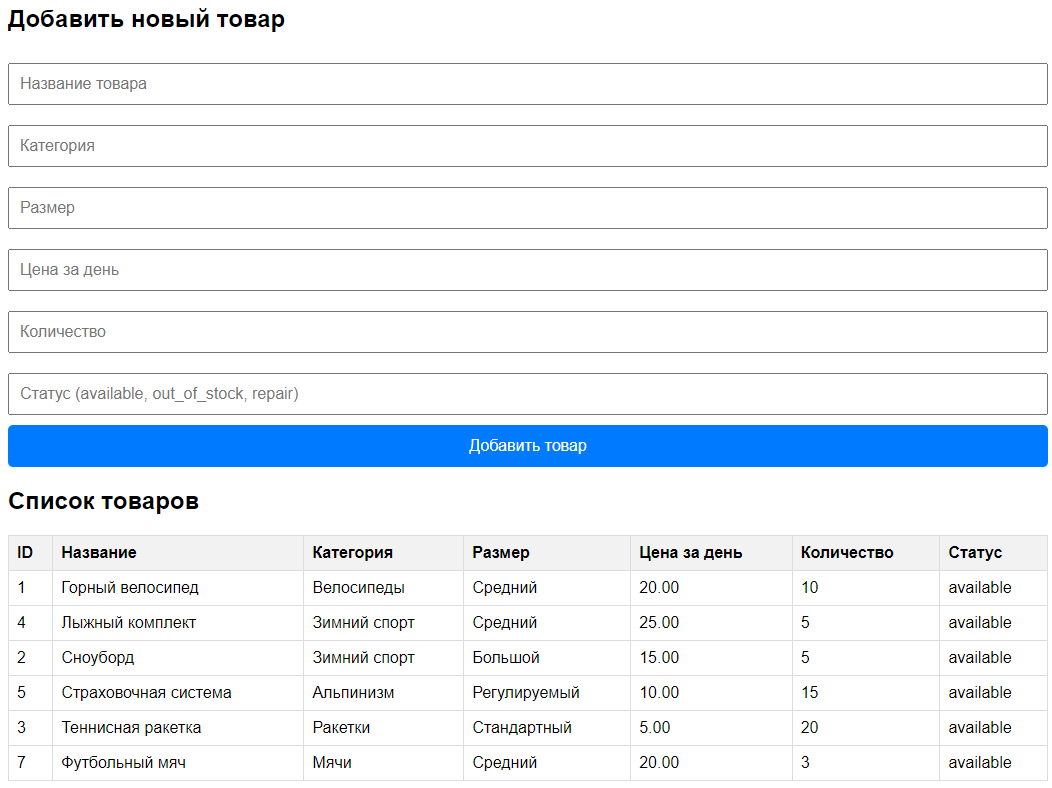


Рисунок №. Вкладка добавление товара

### **3.3 Интерфейс управления клиентами**

На данной странице представлена функциональность для добавления и управления клиентами системы проката спортивного инвентаря. Интерфейс разделён на две части: **форма добавления нового клиента** и **список клиентов**.

#### **Добавление нового клиента**

В верхней части страницы расположена форма для добавления информации о новом клиенте. Поля формы включают:

* **Имя** — имя клиента;
* **Фамилия** — фамилия клиента;
* **Отчество** — отчество клиента;
* **Телефон** — контактный номер телефона;
* **Email** — электронная почта клиента;
* **Адрес** — полный адрес проживания клиента.

После заполнения всех полей администратор может нажать кнопку **«Добавить клиента»**, чтобы сохранить данные клиента в базе данных. Это позволяет оперативно пополнять список новых пользователей.

#### **Список клиентов**

Ниже формы представлена таблица со списком всех зарегистрированных клиентов. Каждая запись содержит:

* **ID** — уникальный идентификатор клиента;
* **Фамилия**, **Имя**, **Отчество** — полное имя клиента;
* **Телефон** — контактный номер клиента;
* **Email** — адрес электронной почты;
* **Адрес** — место жительства клиента.

### **Назначение страницы**

Данный интерфейс позволяет администратору:

1. **Добавлять новых клиентов** в систему, заполняя их контактные и личные данные.
2. **Просматривать текущий список клиентов**, получая доступ к важной информации о каждом пользователе.
3. **Контролировать актуальность данных** клиентов для эффективного взаимодействия и работы системы.

Таким образом, страница упрощает управление клиентской базой и обеспечивает точный учёт пользователей в системе проката.

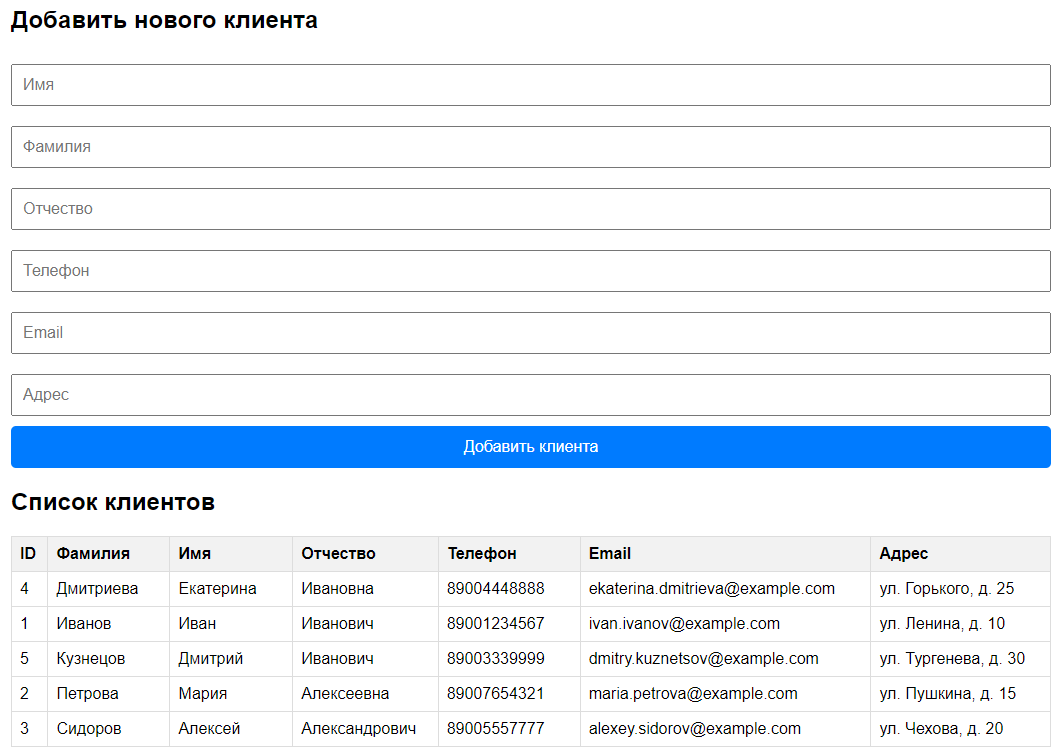


Рисунок №.

### **3.3 Пример PHP-кода для работы с базой данных**

В этом разделе представлены основные шаги и структуры для реализации работы с базой данных MySQL с использованием **PHP**. Примеры кода показывают подключение к базе данных, выполнение SQL-запросов для добавления, удаления, редактирования и получения данных.

#### **Подключение к базе данных**

Для начала работы необходимо установить соединение с базой данных MySQL. Это реализуется с использованием файла с конфигурацией подключения (db.php).

<?php

$host = 'localhost';

$dbname = 'accounting\_of\_sports\_equipment';

$username = 'root';

$password = 'b.5647382910-D';

try {

    $pdo = new PDO("mysql:host=$host;dbname=$dbname;charset=utf8", $username, $password);

    $pdo->setAttribute(PDO::ATTR\_ERRMODE, PDO::ERRMODE\_EXCEPTION);

} catch (PDOException $e) {

    die("Ошибка подключения к базе данных: " . $e->getMessage());

}

?>

#### **Добавление новых записей**

Добавление новых данных в таблицу выполняется с помощью **INSERT**. Пример кода будет отличаться в зависимости от добавляемых данных:

* Добавление нового клиента;
* Добавление нового товара;
* Добавление заказа.

Добавление нового клиента происходит следующим образом:

**Сюда нужно вставить код для выполнения запроса INSERT**  
(например, для добавления нового клиента в таблицу *customers* или нового товара в таблицу *equipment* с использованием подготовленных запросов).

#### **Удаление записей**

Удаление данных из таблицы осуществляется с помощью **DELETE** по уникальному идентификатору записи (id). Это позволяет удалить конкретного клиента, товар или заказ.

**Сюда нужно вставить код для выполнения запроса DELETE**  
(например, для удаления записи из таблицы *orders* по ID заказа).

#### **Редактирование записей**

Редактирование существующих данных выполняется с помощью **UPDATE**. Этот запрос изменяет данные в определённой строке таблицы, например, обновляет количество товара или статус заказа.

**Сюда нужно вставить код для выполнения запроса UPDATE**  
(например, для изменения количества доступного товара в таблице *equipment* или обновления контактных данных клиента в таблице *customers*).

#### **Получение данных**

Для вывода данных из базы используется **SELECT**. Это позволяет получить списки клиентов, товаров или заказов, а также отфильтровать их по определённым критериям.

**Сюда нужно вставить код для выполнения запроса SELECT**  
(например, для отображения списка всех заказов с их параметрами или списка клиентов с полным ФИО и контактами).

Таким образом, блоки кода позволят реализовать основные функции работы с базой данных: добавление, удаление, редактирование и отображение записей для системы проката спортивного инвентаря.