# Информационная система сервиса аренды оборудования

Соломенников Николай 22204

## Введение

Сервис аренды оборудования позволяет клиентам временно пользоваться инструментами, техникой или устройствами без необходимости их покупки. Это удобно для тех, кому оборудование нужно на короткий срок.

Клиент ищет оборудование, бронирует его на нужные даты и забирает в пункте выдачи. Менеджер оформляет аренду, выдает оборудование, контролирует его состояние при возврате и взаимодействует с клиентами. Администратор управляет каталогом оборудования, настраивает тарифы и штрафы, а также анализирует отчеты по доходам и популярности оборудования.

## Назначение приложения

Приложение нужно для автоматизации процессов управления арендой оборудования. Оно поможет компании, занимающейся арендой, эффективно управлять запасами оборудования, оформлять аренду, отслеживать возвраты и состояние оборудования, а также вести финансовую отчетность.

**Целевые пользователи:**

* **Администраторы**: управление каталогом оборудования, регулировка цен, просмотр отчетов.
* **Менеджеры по аренде**: взаимодействие с клиентами, оформление аренды, выдача оборудования, контроль сроков возврата.
* **Клиенты**: получение информации о наличии и стоимости оборудования, бронирование.

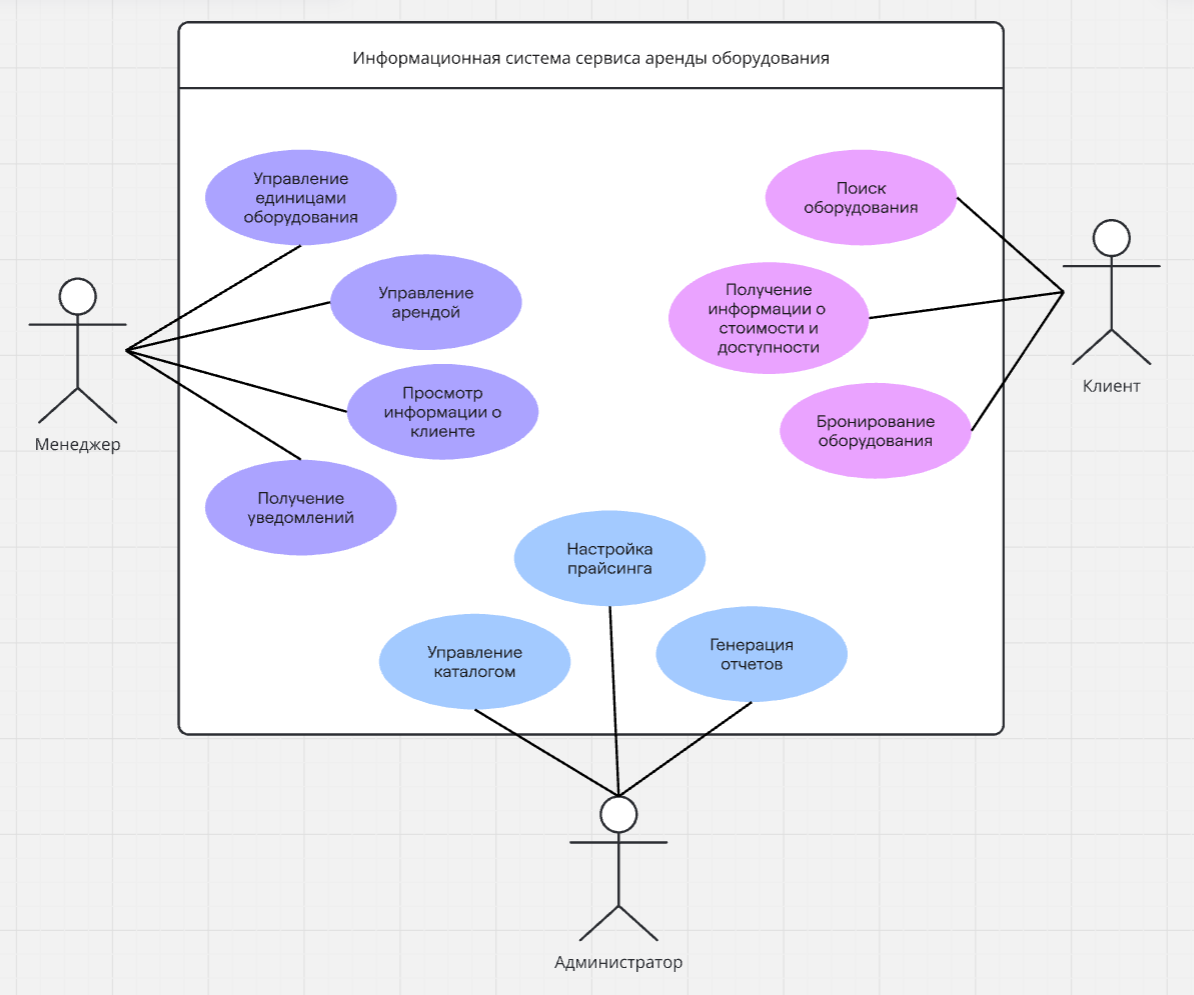
## Полный цикл арендования оборудования

1. **Бронирование оборудования**:
   1. Клиент выбирает оборудование на сайте.
   2. Указывает: ФИО, контактный телефон и email, желаемые даты аренды (начало и конец).
   3. Система проверяет доступность оборудования на указанные даты.
   4. Если оборудование доступно, клиенту предоставляется информация о стоимости аренды, залоге и условиях.
   5. Клиент подтверждает бронирование. Система резервирует оборудование на указанный срок.
2. **Подготовка к выдаче оборудования**:
   1. Менеджер в пункте выдачи получает уведомление о новом бронировании.
   2. Оборудование подготавливается к выдаче (проверяется его состояние, комплектность).
   3. Если клиент не появляется в назначенное время, менеджер связывается с ним для уточнения:
      1. Если клиент отказывается от аренды, бронь отменяется, оборудование снова становится доступным.
      2. Если клиент подтверждает задержку, бронь продлевается на определенное время.
3. **Оформление аренды**:
   1. Клиент приходит в пункт выдачи.
   2. Менеджер проверяет документы клиента.
   3. Клиент подписывает договор аренды, в котором указаны: сроки аренды, стоимость аренды, размер залога, условия возврата и штрафы за повреждения или просрочку.
   4. Клиент вносит залог и оплачивает аренду.
   5. Менеджер выдает оборудование клиенту.
4. **Использование оборудования**
   1. Клиент использует оборудование в течение оговоренного срока.
   2. Если клиенту требуется продлить аренду, он связывается с менеджером:
      1. Менеджер проверяет доступность оборудования на дополнительные дни.
      2. Если оборудование доступно, аренда продлевается, и клиент вносит дополнительную оплату.
      3. Если оборудование недоступно, клиент должен вернуть его в первоначально оговоренный срок.
5. **Возврат оборудования**:
   1. Клиент возвращает оборудование в пункт выдачи.
   2. Менеджер осматривает оборудование на предмет повреждений:
      1. Если повреждений нет, залог возвращается клиенту.
      2. Если есть повреждения, менеджер оценивает их стоимость и вычитает ее из залога. Остаток залога возвращается клиенту.
   3. Менеджер фиксирует возврат оборудования в системе, обновляя его статус на "доступно".
   4. Если клиент не вернул оборудование в срок, менеджер связывается с ним для напоминания:
      1. Если клиент возвращает оборудование с задержкой, начисляется штраф за просрочку.
      2. Если клиент не выходит на связь, могут быть предприняты дополнительные меры (например, обращение в правоохранительные органы).
6. **Завершение аренды**:
   1. Менеджер закрывает аренду в системе:
      1. Фиксирует итоговую сумму оплаты (с учетом штрафов, если они были).
      2. Обновляет историю аренд клиента.
      3. Обновляет состояние оборудования (например, отправляет его на обслуживание, если это необходимо).
   2. Клиенту предоставляется чек или отчет о завершении аренды.

## Бизнес-процессы системы

* **Оформление аренды**:
  + Поиск доступного оборудования.
  + Оформление договора аренды.
  + Оплата и внесение залога.
  + Получение оборудования.
* **Возврат оборудования**:
  + Проверка состояния оборудования.
  + Начисление штрафов за повреждения или просрочку.
  + Возврат залога.
  + Обновление статуса оборудования.
* **Взаимодействие с клиентами**:
  + Ответ на вопросы клиентов.
  + Оформление аренды менеджером.
  + Напоминание о сроках возврата.
* **Учет оборудования**:
  + Добавление, обновление и удаление информации об оборудовании.
  + Учет состояния оборудования (доступно, забронировано, в аренде, на обслуживании).
  + Учет технического обслуживания и ремонта.
* **Отчетность**:
  + Формирование отчетов по продажам.
  + Анализ популярности оборудования.
  + Просмотр истории оборота оборудованием.
  + Учёт рейтинга клиентов (количество заказов, был ли ущерб оборудованию или задержки).

## Бизнес-процессы для каждого типа пользователей

**Администраторы**:

* Управление каталогом оборудования (добавление / удаление карточек оборудования, редактирование описания).
* Настройка прайсинга (тарифы, штрафы).
* Генерация отчетов (доходы, популярное оборудование, истории оборота оборудованием, история аренд клиентов).

**Менеджеры по аренде**:

* Управление конкретными единицами оборудования (добавление / удаление, изменение статуса (доступно, забронировано, в аренде, на обслуживании, списано)).
* Управление арендой (снятие / продление брони, подтверждение аренды, приём оборудования обратно)
* Просмотр информации о клиенте, его истории аренд.
* Получение уведомлений (вопрос от клиента, новая бронь, истечение брони до выдачи оборудования, истечение времени арендования до возврата оборудования).

**Клиенты**:

* Поиск оборудования.
* Получение информации о его стоимости и доступности.
* Бронирование оборудования.

## Информационные потоки для каждого типа пользователей

**Администраторы**:

* Входящие данные: описание оборудования, тарифы, штрафы.
* Исходящие данные: финансовые отчеты, информация о популярном оборудовании, истории оборота оборудованием, история аренд.

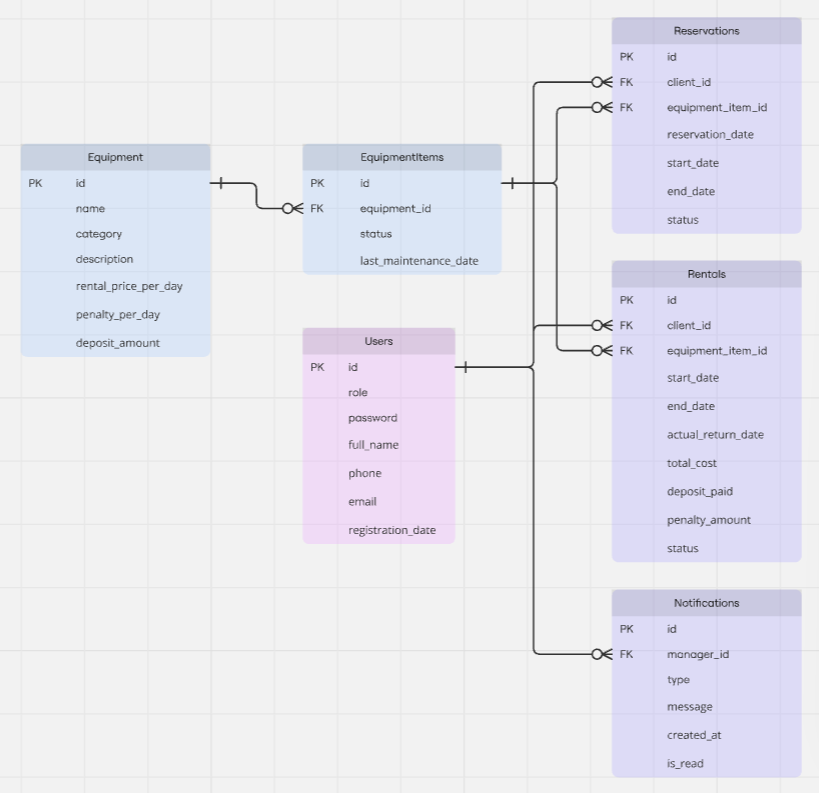
**Менеджеры по аренде**:

* Входящие данные: данные о конкретных единицах оборудования и об арендах.
* Исходящие данные: информация о клиентах, уведомления.
* Отчеты: список активных аренд, просроченные аренды.

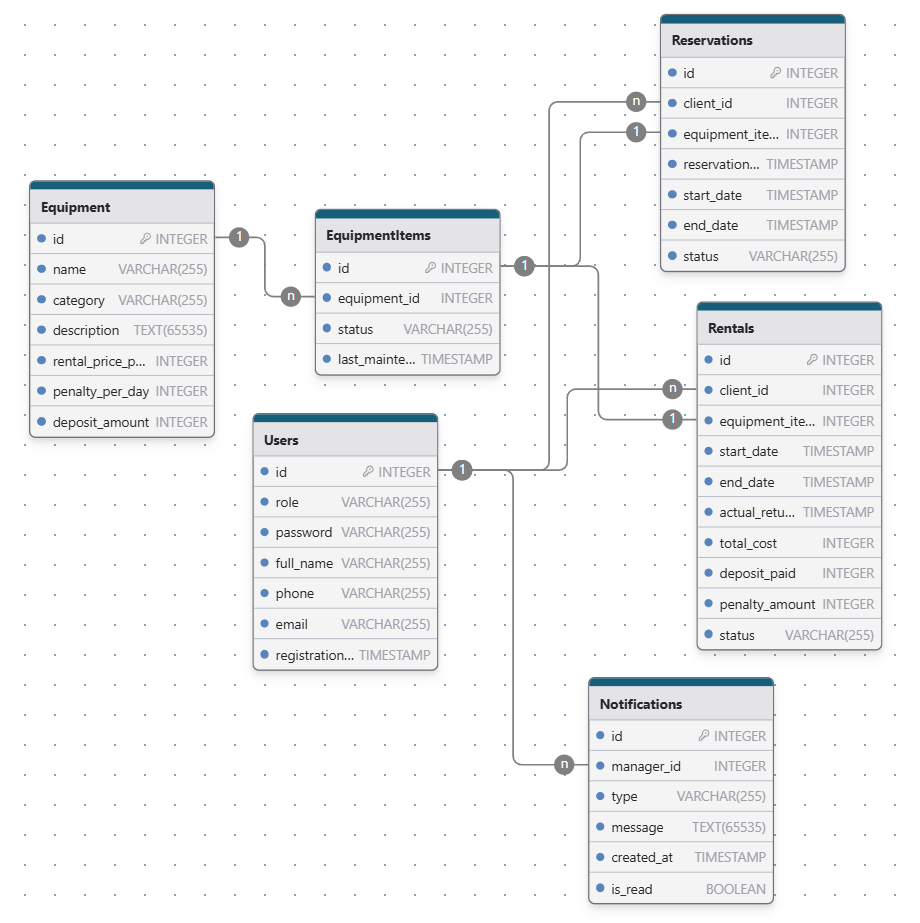
**Клиенты:**

* Входящие данные: личные данные, запросы на аренду.
* Исходящие данные: поиск оборудования, расчет стоимости, уведомления о сроках возврата.

## Основные информационные объекты и связи



## Схема базы данных



1. **Оборудование (Equipment)**:

* id (уникальный идентификатор)
* name (название оборудования)
* category (категория оборудования)
* description (описание)
* rental\_price\_per\_day (стоимость аренды за день)
* penalty\_per\_day (штраф за просрочку возврата за день)
* deposit\_amount (сумма залога)

1. **Единицы оборудования (EquipmentItems)**:

(конкретные физические объекты оборудования)

* + id (уникальный идентификатор среди всех оборудований)
  + equipment\_id (внешний ключ на таблицу Equipment)
  + status (статус: доступно, забронировано, в аренде, на обслуживании, списано)
  + last\_maintenance\_date (дата последнего обслуживания)

1. **Пользователи (Users)**:
   * id (уникальный идентификатор)
   * role (роль: администратор, менеджер, клиент)
   * password (пароль)
   * name (ФИО или просто имя)
   * phone (контактный телефон)
   * email (электронная почта)
   * registration\_date (дата регистрации в системе)
2. **Бронирования (Reservations)**:
   * id (уникальный идентификатор)
   * client\_id (внешний ключ на таблицу Users)
   * equipment\_id (внешний ключ на таблицу Equipment)
   * reservation\_date (дата бронирования)
   * start\_date (планируемая дата начала аренды)
   * end\_date (планируемая дата окончания аренды)
   * status (статус: активна, отменена, завершена)
3. **Аренда (Rentals)**:
   * id (уникальный идентификатор)
   * client\_id (внешний ключ на таблицу Users)
   * equipment\_item\_id (внешний ключ на таблицу EquipmentItems)
   * start\_date (дата начала аренды)
   * end\_date (дата окончания аренды)
   * actual\_return\_date (фактическая дата возврата)
   * total\_cost (сумма, заплаченная за аренду)
   * deposit\_paid (сумма залога, внесенная клиентом)
   * penalty\_amount (штраф за повреждения или просрочку)
   * status (статус: активна, завершена)
4. **Уведомления (Notifications)**:
   * id (уникальный идентификатор)
   * manager\_id (внешний ключ на таблицу Users)
   * type (тип уведомления: вопрос от клиента, новая бронь, истечение брони, истечение аренды)
   * message (текст уведомления)
   * created\_at (дата и время создания уведомления)
   * is\_read (флаг прочтения уведомления)

## Основные ограничения

1. **Целостность данных**:
   1. Каждая аренда должна быть связана с существующим клиентом и оборудованием.
   2. Оборудование может быть арендовано только если его статус "доступно".
2. **Уникальность данных**:
   1. Каждая единица оборудования должна иметь уникальный идентификатор (например, серийный номер).
   2. Клиенты должны быть уникально идентифицированы (например, по email или телефону).
3. **Ограничения на ввод данных**:
   1. Дата возврата аренды не может быть раньше даты начала.
   2. Стоимость аренды не может быть отрицательной.
4. **Бизнес-правила**:
   1. Штрафы за просрочку возврата рассчитываются автоматически на основе тарифов.

## Стек технологий

## 

## Фронтенд (Клиентская часть)

* **Язык программирования:** JavaScript.
* **Библиотека:** React — используется для построения интерфейса на основе компонентов.
* **Сборщик и дев-сервер:** Vite — обеспечивает быструю сборку и горячую перезагрузку во время разработки.

### Бэкенд (Серверная часть)

* **Язык программирования:** Python.
* **Фреймворк:** Flask — лёгкий веб-фреймворк, предназначенный для создания REST API и быстрой обработки HTTP-запросов.

### База данных

* **СУБД:** MySQL
* **Связь с бэкендом:** Через Python-библиотеку mysql-connector.

### Взаимодействие компонентов

* **Фронтенд ↔ Бэкенд:**
  + Используются HTTP-запросы к REST API, реализованному во Flask.
  + Данные передаются в формате JSON.
* **Бэкенд ↔ База данных:**
  + Взаимодействие через SQL-запросы или ORM.
  + Обработка, сохранение и извлечение данных из MySQL.

### Дополнительно

* Возможность контейнеризации с использованием Docker.

## Реализация

[Репозиторий с исходниками](https://github.com/SolomennikovKolya/nsu-course-3/tree/main/db/project_rental)

На данный момент сделано:

* Основа бэкенда (архитектура, основные процедуры).
* Основа фронтенда (каркас сайта).
* Система авторизации: на сайте можно зайти как менеджер или администратор, после чего откроются специальные возможности. Данные аккаунтов, в которые можно зайти фиксированы и добавляются другими менеджерами. То есть, чтобы новый сотрудник мог зайти как менеджер или администратор, другой администратор должен добавить его данные в систему. Авторизация реализована через JWT и Refresh токены. JWT хранятся у пользователя, а Refresh токены - в базе данных на сервере. На сайте роль пользователя используется, чтобы скрывать недоступные элементы, а на сервере роль проверяется, чтобы выполнить или отклонить запрос. Также роль используется, чтобы ограничивать права изменения базы данных (реализовано через подключение к базе данных под именем специальных пользователей, с заданными правами).
* Поддержка .env файлов для хранения ключей и других приватных данных.

## SQL запросы

[Основные SQL запросы](https://github.com/SolomennikovKolya/nsu-course-3/tree/main/db/project_rental/backend/src/db/queries)

Остальные SQL запросы представлены в коде и обёрнуты в функции, причём у этих функций указан собственный декоратор @with\_db. Это означает, что при выполнениях каждой из этих функций будет производиться подключение к базе данных с заданными параметрами входа, а при завершении функции - автоматическое освобождение всех ресурсов и отключение от базы данных. Решение не поддерживать соединение, а переподключаться к базе данных каждый раз заново, было принято специально, так как запросы от клиентов приходят не часто, а пользователей может быть много. Пример кода:

