**АВТОНОМНАЯ НЕКОММЕРЧЕСКАЯ ОРГАНИЗАЦИЯ ПРОФЕССИОНАЛЬНАЯ ОБРАЗОВАТЕЛЬНАЯ ОРГАНИЗАЦИЯ МОСКОВСКИЙ МЕЖДУНАРОДНЫЙ КОЛЛЕДЖ ЦИФРОВЫХ ТЕХНОЛОГИЙ**

**«АКАДЕМИЯ ТОП»**

**ИНДИВИДУАЛЬНЫЙ ПРОЕКТ**

Уровень профессионального образования:

Среднее профессиональное образование

Программа подготовки специалистов среднего звена

по специальности:

09.02.07 Информационные системы и программирование

Квалификация: Программист

Учебный предмет: Технология доступа к базам данных ADO.NET

Тема: Система аренды электросамокатов и велосипедов.

Преподаватель:

О. А. Рослова \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Обучающиеся:

М.С. Селиванов\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Санкт-Петербург, 2025**

**Введение**

**Актуальность проекта**

Развитие городской мобильности и популяризация экологичных видов транспорта обуславливают высокий спрос на услуги краткосрочной аренды электросамокатов и велосипедов. Эффективное управление таким бизнесом, включающее учет транспорта, клиентов, арендных сессий и финансовых операций, требует надежной и производительной системы автоматизации. Разработка специализированного программного обеспечения позволяет не только оптимизировать эти процессы, но и обеспечить масштабируемость бизнеса и высокое качество обслуживания клиентов.

**Технологический контекст и обоснование выбора технологии**

Для взаимодействия приложения с базой данных был выбран Dapper – высокопроизводительный микро-ORM (Object-Relational Mapper), разработанный командой Stack Overflow и расширяющий возможности IDbConnection.

Выбор Dapper в качестве основной технологии доступа к данным обусловлен следующими факторами:

Производительность: Dapper демонстрирует скорость работы, близкую к сырому ADO.NET, поскольку генерирует минимальный накладной код при маппинге результатов SQL-запросов в объекты. Это критически важно для системы, которая должна быстро обрабатывать множество операций аренды и возврата.

Простота и контроль: В отличие от полнофункциональных ORM, таких как Entity Framework, Dapper не скрывает SQL от разработчика. Это предоставляет полный контроль над формированием запросов, что позволяет писать максимально эффективные и оптимизированные команды для сложных выборок и отчетов.

Безопасность: Dapper полностью поддерживает параметризованные запросы, что является основной мерой защиты от SQL-инъекций, обеспечивая при этом удобный синтаксис работы с параметрами.

Легковесность: Dapper добавляет минимальное количество абстракций, что делает его простым в изучении и интеграции в проект, а также снижает потребление ресурсов приложения.

**Цель и задачи проекта**

Целью проекта является разработка высокопроизводительного и масштабируемого программного обеспечения для автоматизации деятельности сервиса аренды электросамокатов и велосипедов с использованием современного стека технологий .NET и микро-ORM Dapper.

Для достижения этой цели необходимо решить следующие задачи:

Провести анализ предметной области и сформулировать функциональные и нефункциональные требования к системе.

Спроектировать нормализованную реляционную модель базы данных, включающую таблицы для учета транспорта, пользователей, арендных операций, платежей и справочной информации.

Реализовать слой доступа к данным (Data Access Layer, DAL) с использованием Dapper для выполнения CRUD-операций (Create, Read, Update, Delete) и сложных бизнес-запросов.

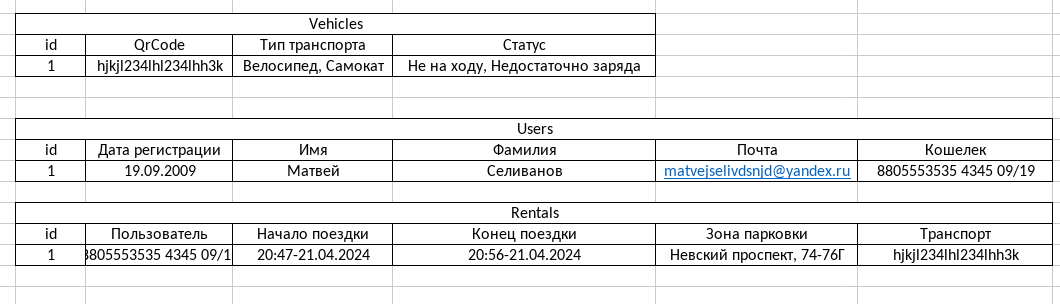
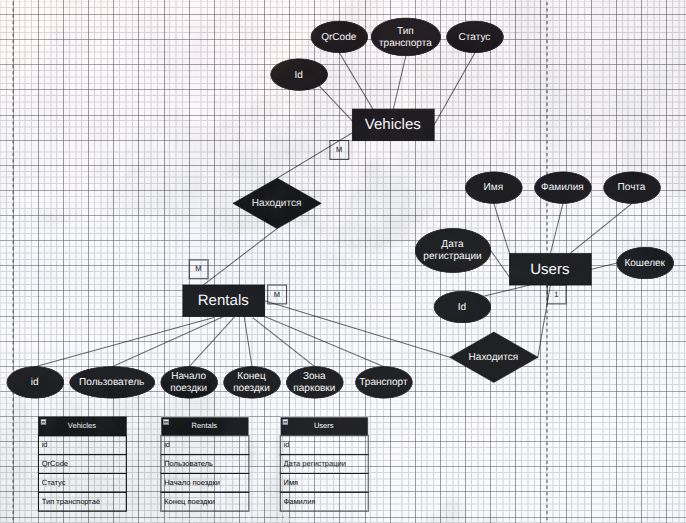
Разработать пользовательский интерфейс для операторов системы и клиентов, обеспечивающий интуитивное взаимодействие с основными функциями: регистрация, поиск транспорта, оформление и завершение аренды, формирование отчетов.

Провести интеграционное тестирование слоя доступа к данным для проверки корректности маппинга и работы всех запросов.

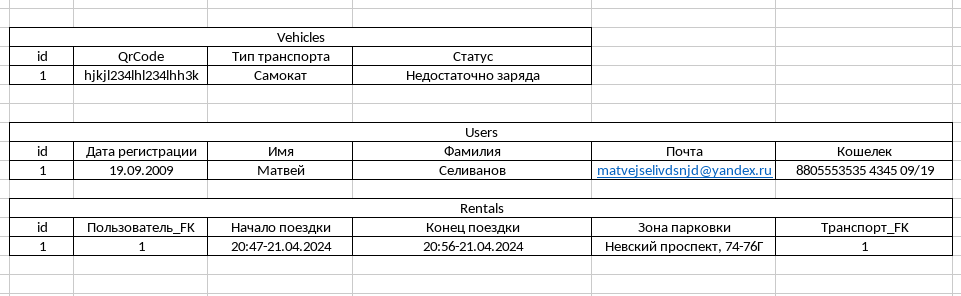
**Проектирование базы данных**

Для создание базы были выделены ключевые объекты (сущности):

1. Users (Пользователи)
   1. Атрибуты: id, Дата регистрации, Имя, Фамилия, Почта, Кошелек.
   2. Описание: содержит в себе полные данные о арендаторах, которые берут транспорт на прокат. Также содержит кошелек, в котором хранится номер карты пользователя, с которого списываются деньги за аренду.
2. Vehicles (Транспорт)
   1. Атрибуты: id, QrCode, Тип траспорта, Статус
   2. Описание: в этой таблице содержится вся информации об определенном транспорте, его статус работы, тип транспорта для учета цены в минуту и уникальный qrcode, для уникальности устройства от себя подобных.
3. Rentals (Аренды)
   1. Атрибуты: id, Пользователь, Начало поездки, Конец поездки, Зона парковки, Транспорт.
   2. Описание: Это основная таблица, в которой хранятся все данные об арендах, в ней показано дата/время – начала и конца поездки, зона парковки, чтобы узнать место, куда припаркован транспорт, чтобы выписать штраф или найти его, также указан пользователь, чтобы идентифицировать арендатора, а также сам транспорт.



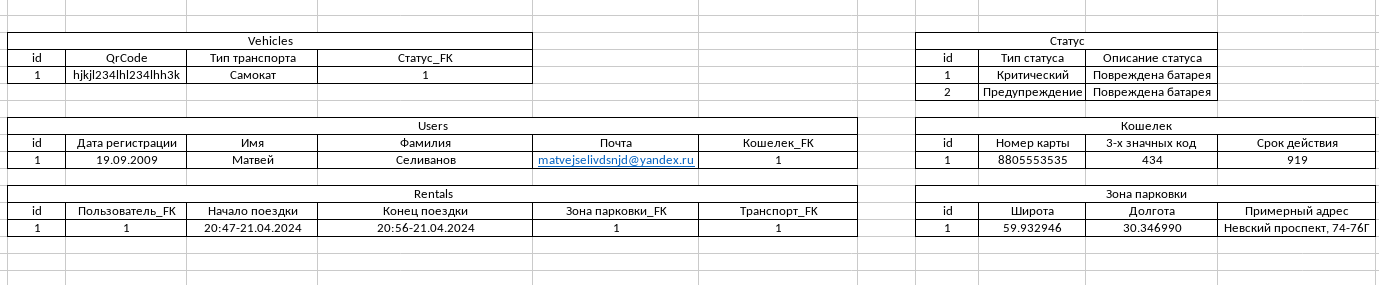
**Нормализация в 1NF**



Для нормализации в 1 форму, было выполнено:

1. Атомарные данные, нету несколько данных в одной клетке.
2. Каждая запись стала уникальная.
3. Появились ключи, по которым можно получить данные из другой таблицы без повторений.

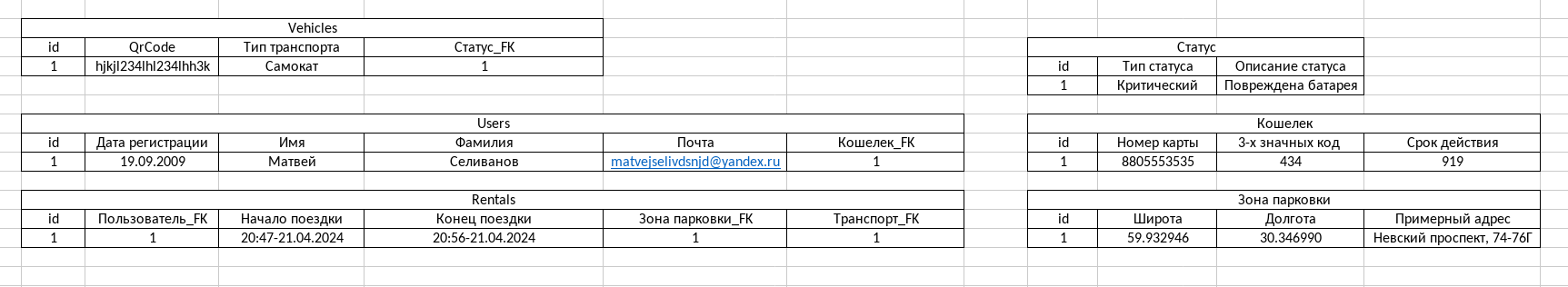
**Нормализация во 2NF**



Для нормализации во 2 форму, было выполнено:

1. Статус в таблице “Vehicles” вынесен в отдельную таблицу “Статус”, где более четко расписан.
2. Кошелек в таблице “Users” был вынесен в отдельную таблицу “Кошелек”, где полные данные о карте были разнесены по разным колонкам для лучшего их чтения
3. Зона парковки в таблице “Rentals” была вынесена в таблицу “Зона парковки”, где достаточно расписаны точные координаты транспорта и его примерный адрес.

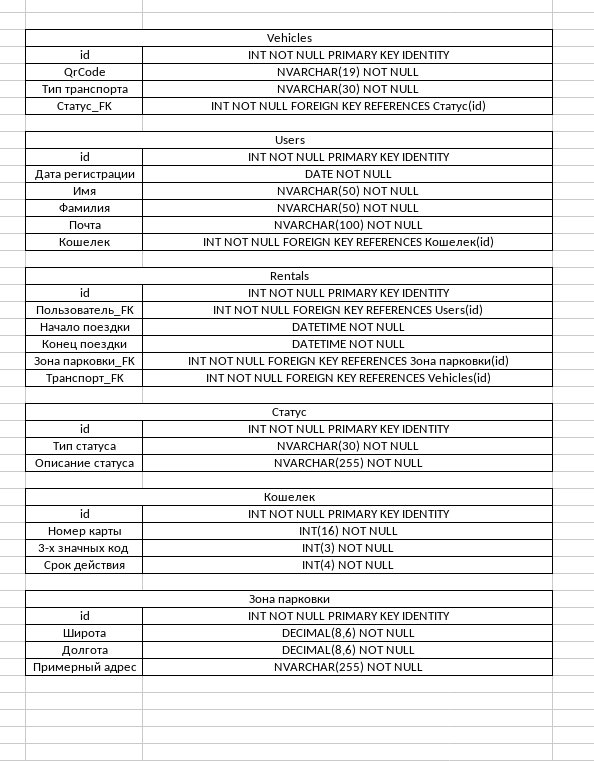
**Нормализация в 3NF**



Для нормализации в 3 форму, было выполнено:

1. Убраны транзитивные зависимости и повторяющиеся данные.

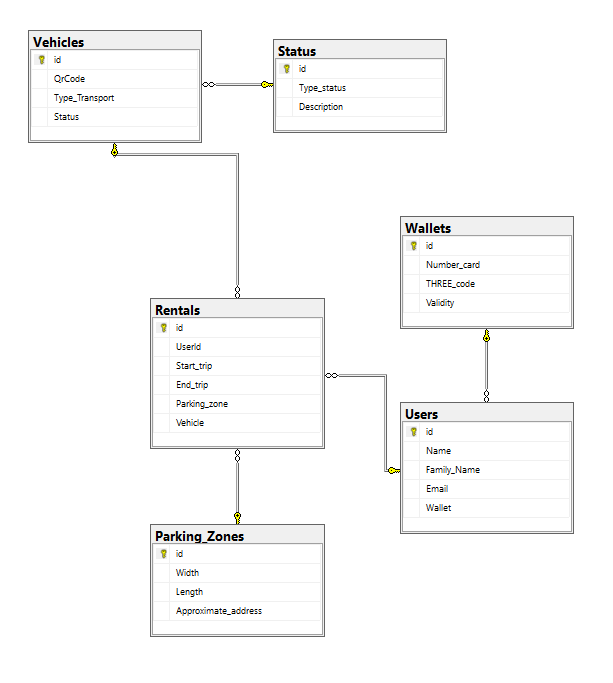
**Полная база данных в 3NF**



**Реализация таблиц в СУБД**

Файл Rental\_system.sql

**Диаграмма базы данных**



Между таблицами установлена связь через внешние ключи, они обеспечивают целостность данных.

Сама же диаграмма, отображает эти связи, где один пользователь может совершать несколько аренд, а одно транспортное средство может участвовать в различных арендах. Также у одного транспорта может быть несколько несколько статусов работы.