Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н. Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н. Э. Баумана)

ФАКУЛЬТЕТ «Информатика, искусственный интеллект и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К КУРСОВОЙ РАБОТЕ

НА ТЕМУ:

«Разработка базы данных сервиса краткосрочной аренды городских электросамокатов (кикшеринг)»

Студент ИУ7-65Б (Группа) _____ (Подпись, дата) М. Д. Котцов (И. О. Фамилия)

Руководитель курсовой работы _____ (Подпись, дата) М. В. Филиппов (И. О. Фамилия)

2023 г.

РЕФЕРАТ

Расчетно-пояснительная записка 28 с., 9 рис., 1 табл., 7 источн., 2 прил.
Ключевые слова: базы данных, SQL, REST API, PostgreSQL, TypeScript, городской транспорт, кикшеринг, электросамокаты.

Объектом разработки является база данных сервиса краткосрочной аренды городских электросамокатов (кикшеринг).

Цель работы — спроектировать и разработать базу данных для хранения данных сервиса кикшеринга.

Для достижения данной цели были решены следующие задачи:

- проанализированы варианты представления данных и выбран подходящий вариант для решения задачи;
- проанализированы системы управления базами данных и выбрана подходящая система для хранения данных;
- спроектирована база данных, описаны ее сущности и связи;
- реализован интерфейс для доступа к базе данных;
- реализовано программное обеспечение, позволяющее взаимодействовать со спроектированной базой данных.

В результате выполнения работы была спроектирована и разработана база данных для хранения данных сервиса по аренде электросамокатов. Также было разработано приложение для доступа к этим данным.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

С начала 2000-х годов отмечается появление шеринговых центров и развитие экономики совместного пользования за определенную сумму без права владения (шеринг-экономики). За последнее десятилетие шеринг стал одной из основных идей, изменивших представление о потреблении.

Бурный рост мегаполисов предъявляет всё более строгие требования к городской мобильности. Кикшеринг-сервисы стали ответом на современные вызовы. Кикшеринг можно считать наиболее инновационной системой краткосрочной аренды транспортных средств. Массовое использование электросамокатов в качестве городского транспорта началось сравнительно недавно: в России сервисы проката электросамокатов появились в 2018 году. Сегодня арендовать электросамокаты можно в более чем 30 городах страны.

В связи с растущей популярностью подобных сервисов [1] всё чаще возникает проблема хранения операционных данных о поездках, пользователях, парке электросамокатов и многом другом.

Целью данной работы является проектирование и разработка базы данных для хранения данных сервиса краткосрочной аренды электросамокатов.

Для достижения данной цели необходимо решить следующие задачи:

- проанализировать варианты представления данных и выбрать подходящий вариант для решения задачи;
- проанализировать системы управления базами данных и выбрать подходящую систему для хранения данных;
- спроектировать базу данных, описать ее сущности и связи;
- реализовать интерфейс для доступа к базе данных;
- реализовать программное обеспечение, позволяющее взаимодействовать со спроектированной базой данных.

1 Аналитический раздел

В данном разделе будут выдвинуты требования к приложению, определены пользователи системы, формализованы хранимые сервисом данные. Также будет проведен анализ существующих решений и выбрана модель базы данных.

1.1 Анализ предметной области

Предметной областью поставленной задачи является операционная деятельность сервиса краткосрочной аренды электросамокатов в крупном городе.

Пользователями кикшеринга являются лица, достигшие возраста 18 лет. Для регистрации необходим адрес электронной почты, номер телефона, а также реквизиты банковской карты [2].

Для поиска доступных для аренды электросамокатов пользователи используют специально разработанное мобильное приложение, где на карте отмечены парковки и свободные самокаты. В этом же приложении пользователи кикшеринг-сервиса выбирают нужный им самокат и или бронируют его, или начинают на нём аренду.

Бронирование позволяет скрыть электросамокат с карты для других пользователей на определенное время. В большинстве кикшеринг-сервисов бронирование бесплатно и предусматривает отмену без необходимости начинать аренду. При этом некоторые сервисы могут требовать от пользователя находиться недалеко от бронируемого самоката.

Для начала аренды пользователи сканируют QR-код на руле электросамоката или вводят его номер.

Стоимость аренды зачастую складывается из двух составляющих: платы за старт и платы за каждую минуту поездки. В зависимости от спроса на самокаты на конкретной парковке цены могут быть скорректированы в бóльшую сторону. Перед началом аренды на банковской карте пользователя блокируется определенная сумма, которая возвращается после завершения поездки. По желанию пользователя и за дополнительную плату поездка может быть застрахована.

Во время поездки пользователю доступна информация о километраже, длительности и стоимости аренды. При попадании в зону с ограничением максимальной скорости движения электросамокат автоматически сбрасывает

скорость и не позволяет превысить установленный лимит до выезда из зоны.

Пользователи также имеют возможность поставить аренду на паузу и заблокировать самокат. При этом во время простоя плата за аренду продолжает начисляться.

Завершение аренды возможно на специальных парковках, отмеченных на карте в мобильном приложении. По завершении аренды у пользователя может быть запрошена фотография припаркованного самоката.

Для решения вопросов и проблем при использовании сервиса пользователи могут обратиться к сотрудникам клиентской поддержки в чате в приложении.

Для автоматизации учета информации о пользователях, поездках, самокатах, тарифах и т.д. разрабатывается специализированная информационная система. Использование данной системы возможно пользователями с различными уровнями доступа.

1.2 Анализ существующих решений

Для проведения анализа существующих решений среди имеющихся на рынке сервисов кикшеринга были выбраны три наиболее популярных [3]: Whoosh [4], Юрент [5] и Яндекс Go [6]. Сравнение проводилось по следующим критериям.

1. Количество одновременных аренд разных самокатов, которое может начать пользователь с одного аккаунта.

2. Возможность забронировать электросамокат, находясь на любом расстоянии до него.

3. Необходимость делать фотографии припаркованного самоката после завершения поездки.

4. Возможность поставить аренду на паузу по более низкой цене в минуту, чем во время активной аренды.

5. Возможность посмотреть суммарный километраж всех поездок.

Результат сравнения представлен в таблице 1.1.

Таблица 1.1 – Анализ существующих решений

Критерий	**Whoosh**	**Юрент**	**Яндекс Go**
Количество одновременных аренд	не более 3	не более 5	не более 3
Бронирование с любого расстояния	+	−	−
Фотография после поездки	−	+	+
Пауза по сниженной цене	−	−	+
Суммарный километраж	+	−	−

Создаваемый в рамках курсовой работы сервис должен не просто быть на уровне с конкурентами, но и предоставлять пользователям дополнительный функционал. Конкурентными преимуществами станут:

- возможность взять в аренду до 10 самокатов одновременно;
- возможность бронировать самокат с любого расстояния;
- отсутствие необходимости делать фотографию после завершения аренды.

1.3 Требования к разрабатываемой базе данных и приложению

Предметная область поставленной задачи является обширной и включает в себя множество понятий и связей между ними, в связи с этим были сформулированы следующие требования к разрабатываемой программе в рамках курсовой работы.

1. Должен быть предоставлен функционал для регистрации и аутентификации пользователей в системе, также должен быть создан личный кабинет с основной информацией о пользователе.

2. Должен быть предоставлен функционал для получения списка парковок и доступных для аренды электросамокатов.

3. Должен быть предоставлен функционал для начала и завершения аренды.

4. Должен быть предоставлен функционал для ограничения скорости электросамокатов в определенных зонах города.

5. Должен быть предоставлен функционал для бронирования электросамокатов.

6. Должен быть предоставлен функционал для сохранения истории поездок с возможностью просмотра пользователем своих поездок.

7. Должен быть предоставлен функционал для взаимодействия с клиентской поддержкой.

8. Должен быть предоставлен функционал для хранения истории перемещений электросамокатов.

Также были сформулированы следующие допущения:

1. Оплата поездок производится сторонним платежным сервисом, предоставляющем программный интерфейс для взаимодействия (API). Реализация подобного функционала выходит за рамки курсовой работы по базам данных.

2. Верификация номера телефона пользователя, равно как и отправка на него СМС-сообщений реализуется сторонним сервисом, предоставляющим программный интерфейс для взаимодействия (API). Реализация подобного функционала выходит за рамки курсовой работы по базам данных.

1.4 Формализация данных

На рисунке 1.1 представлена ER-диаграмма сущностей проектируемой базы данных в нотации Чена, описывающая объекты предметной области и их взаимодействие.

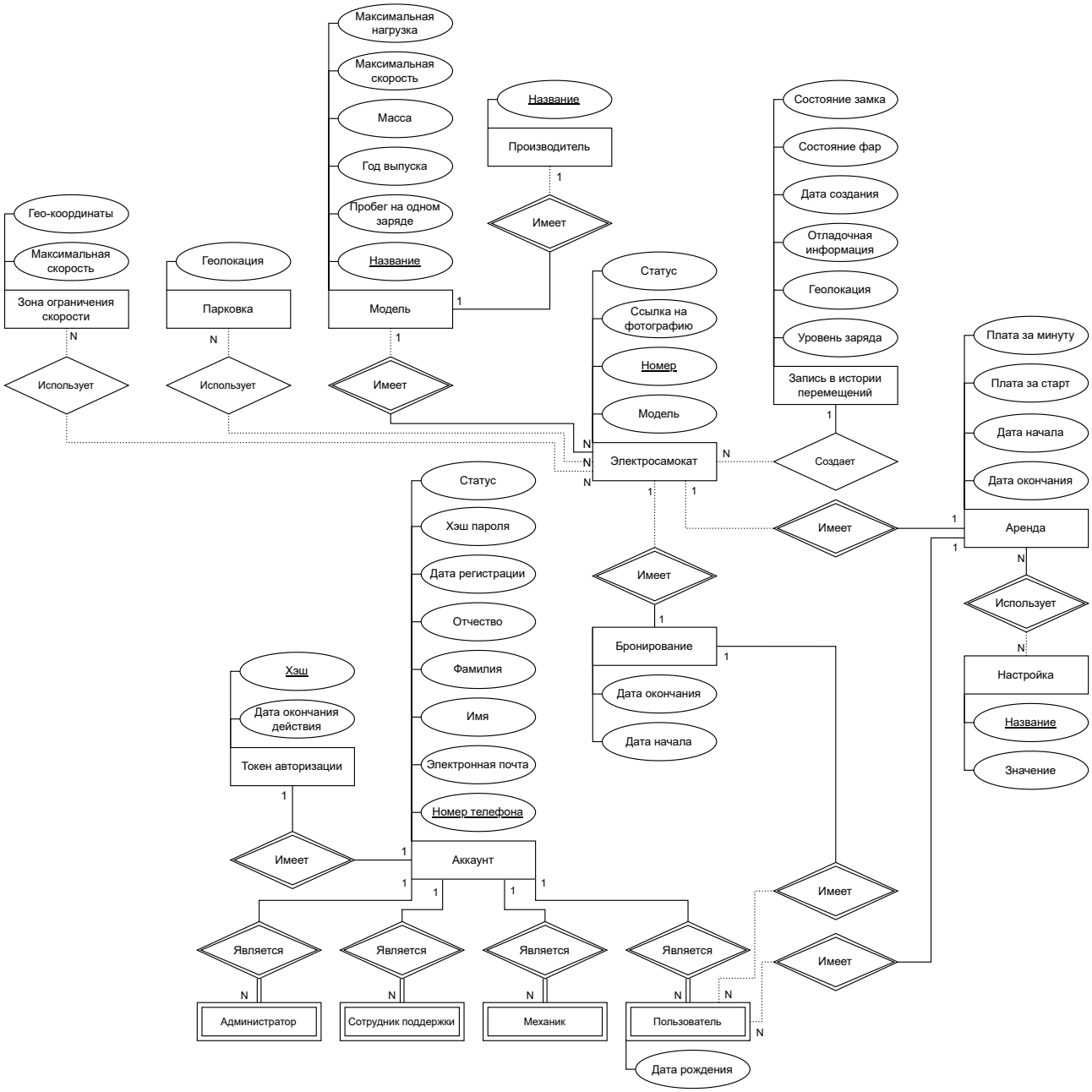

Рисунок 1.1 – ER-диаграмма сущностей проектируемой базы данных

Информационная система, проектируемая в ходе выполнения курсовой работы, включает в себя информацию о следующих объектах.

1. Аккаунт — сущность общего вида, описывающая каждый из четырех возможных подтипов сущностей-пользователей в системе: администратор, сотрудник клиентской поддержки, механик и пользователь. Сущность пользователя дополнительно хранит дату рождения для верификации возраста.

2. Парковка — сущность, описывающая городскую велопарковку или любое другое место, где можно оставить самокат.

3. Зона ограничения скорости — сущность, описывающая часть города, где максимальная скорость движения ограничена некоторым значением из соображений безопасности.

4. Электросамокат — сущность, хранящая информацию о самокате, который пользователи могут брать в аренду. Как IoT-устройство [7] самокат может отправлять информацию о себе в систему, создавая записи в истории перемещений. Во время аренды электросамокат может попадать в несколько зон ограничения скорости. Для завершения аренды самокат должен быть оставлен на одной из доступных парковок.

5. Бронирование — сущность, описывающая процесс резервирования пользователем конкретного самоката на определенный промежуток времени.

6. Аренда — сущность, описывающая процесс проката пользователем электросамоката с фиксированным на момент старта поездки тарифом.

7. Настройка — сущность, хранящая соответствие имени переменной и некоторого значения (например, стоимости страховки).

8. Токен авторизации — сущность, которая используется для авторизации запросов от пользователя к серверу для исключения передачи пароля.

9. Код — сущность, которая используется для верификации номера телефона путем отправки на него СМС-сообщения со случайным числом.

1.5 Формализация пользователей приложения

Ролевая модель используется для реализации системы безопасности сервера базы данных и позволяет разрешать или запрещать тем или иным группам пользователей работу с объектами базы данных.

В рамках поставленной задачи выделены следующие роли.

1. Администратор. Имеет максимальный уровень доступа ко всем данным сервиса.

2. Сотрудник клиентской поддержки. Имеет доступ к информации о пользователях и их поездках.

3. Механик. Имеет доступ к информации о самокатах, которые требуют обслуживания (например, замены аккумулятора).

4. Пользователь. Имеет возможность брать самокаты в аренду, просматривать историю поездок, связываться с клиентской поддержкой.

5. Электросамокат. Имеет возможность создавать записи в истории перемещений.

1.6 Диаграмма вариантов использования

На рисунках 1.2–1.6 представлены диаграммы вариантов использования системы в соответствии с выделенными типами пользователей.

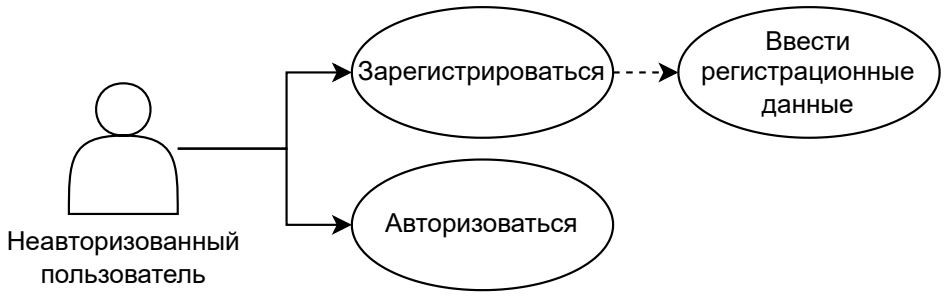

Рисунок 1.2 – Диаграмма вариантов использования системы неавторизованным пользователем

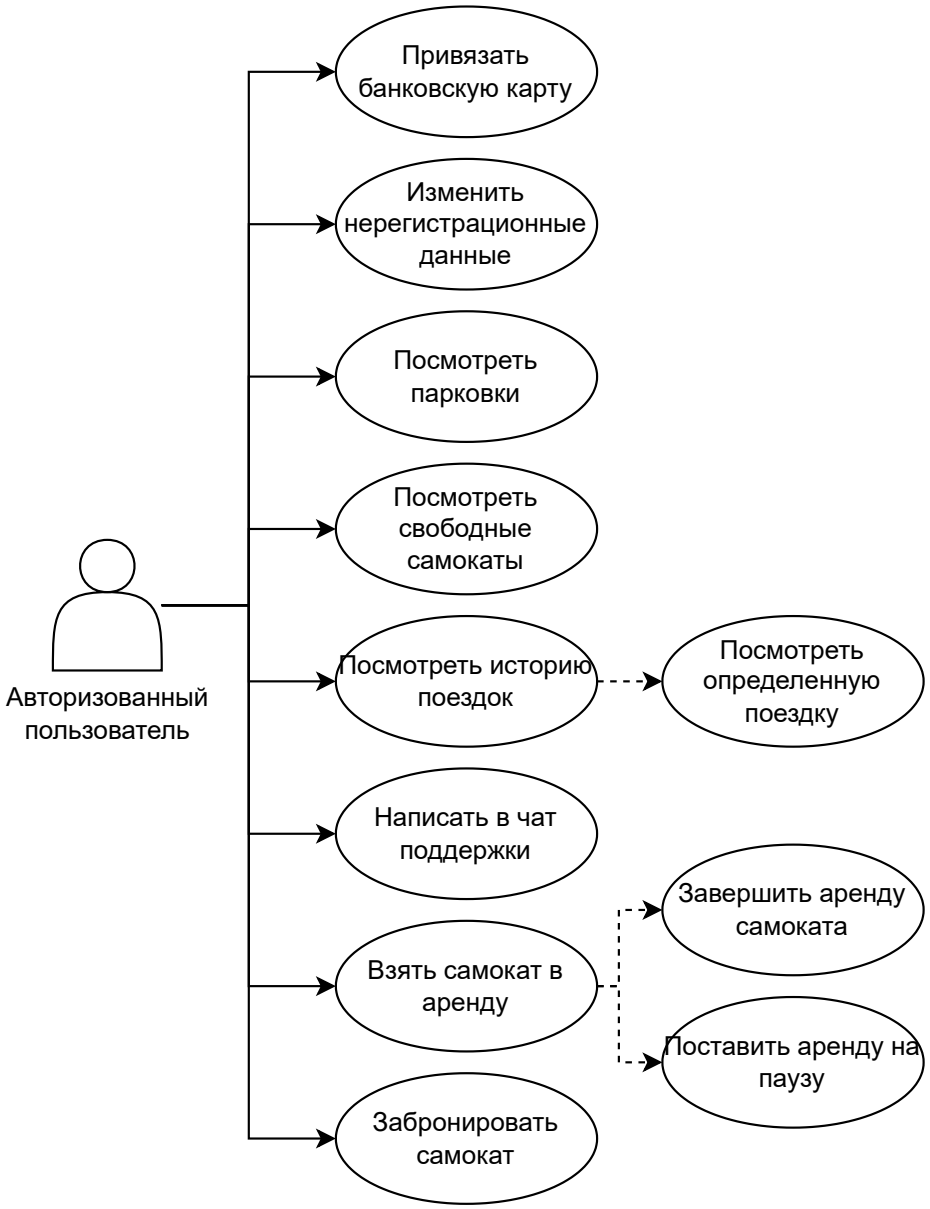

Рисунок 1.3 – Диаграмма вариантов использования системы авторизованным пользователем

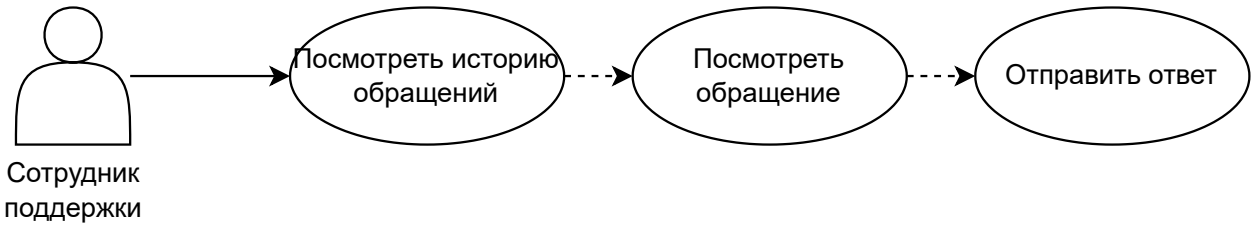

Рисунок 1.4 – Диаграмма вариантов использования системы сотрудником клиентской поддержки

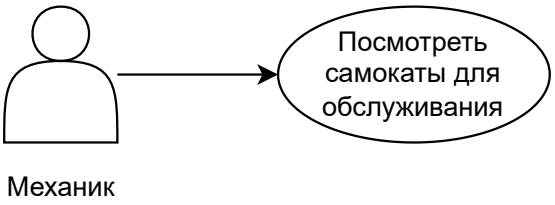

Рисунок 1.5 – Диаграмма вариантов использования системы механиком

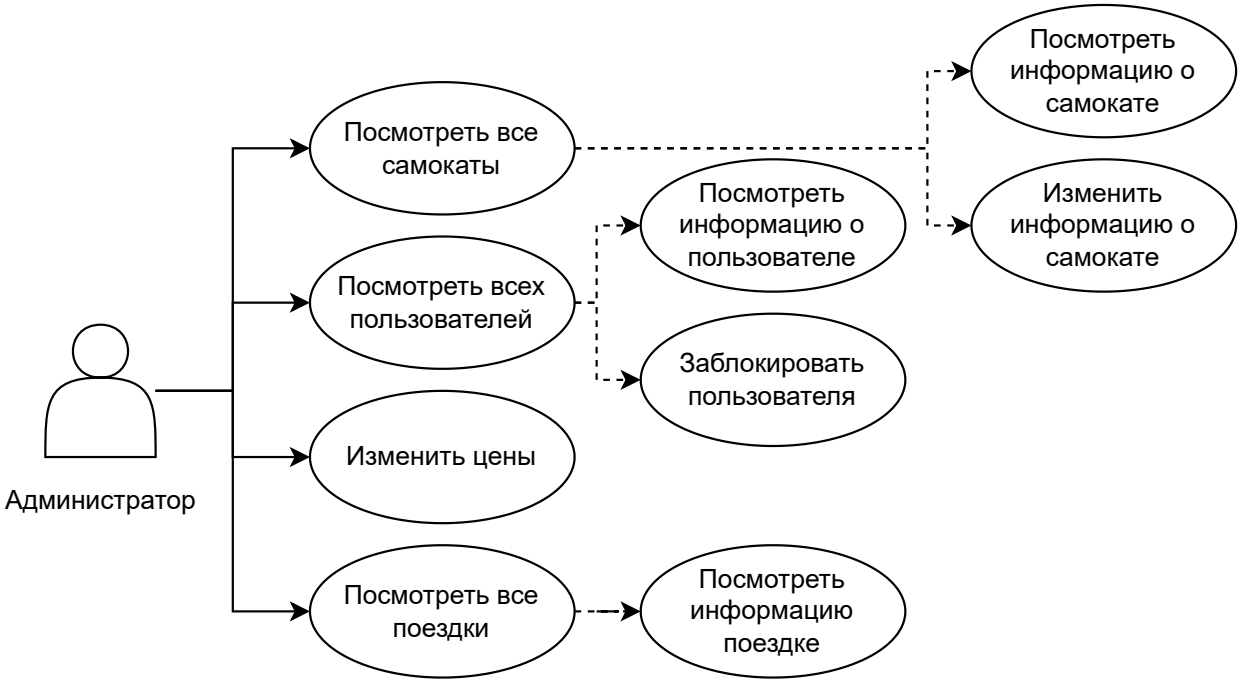

Рисунок 1.6 – Диаграмма вариантов использования системы администратором

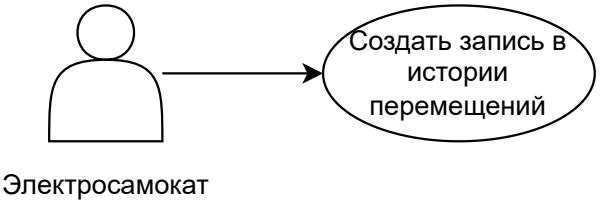

Рисунок 1.7 – Диаграмма вариантов использования системы электросамокатом

1.7 Анализ существующих баз данных

По организации и способу хранения данных все базы данных можно разделить на две группы: реляционные и нереляционные. Реляционные базы данных в свою очередь делятся на строковые и колоночные, а нереляционные — на графовые, документные и базы данных типа «ключ-значение».

1.7.1 Реляционные базы данных

Реляционные базы данных основываются на реляционной модели данных. Данные в таких базах организованы в виде набора таблиц, состоящих из столбцов и строк. В таблицах хранится информация об объектах, представленных в базе данных. Такие базы данных удобно использовать для хорошо структурированных данных.

Строковые базы данных

Строковыми базами данных называются базы данных, записи которых в памяти представлены построчно. Строковые базы данных используются в транзакционных системах. Для таких систем характерно большое количество коротких транзакций с операциями вставки, обновления и удаления данных.

Колоночные базы данных

Колоночными базами данных называются базы данных, записи которых в памяти представляются по столбцам. Колоночные базы данных используются в аналитических системах. Такие системы характеризуются низким объемом транзакций, а запросы к ним зачастую сложны и включают в себя агрегацию.

1.7.2 Нереляционные базы данных

Нереляционная база данных — это база данных, в которой в отличие от большинства традиционных систем баз данных не используется табличная схема строк и столбцов. В этих базах данных применяется модель хранения, оптимизированная под конкретные требования типа хранимых данных.

Базы данных «ключ-значение»

В базах данных «ключ-значение» данные хранятся как совокупность пар ключ-значение, где ключ служит уникальным идентификатором. Такие базы данных удобно использовать для хранения и обработки разных по типу и содержанию данных, их легко масштабировать. Однако такой тип баз данных не подходит для работы со сложными и связанными друг с другом данными.

Документные базы данных

Документные базы данных — это тип нереляционных баз данных, предназначенный для хранения и запроса данных в виде документов в формате, подобном JSON. Такие базы данных позволяют хранить и запрашивать данные в базе данных с помощью той же документной модели, которая используются в коде приложения. Документные базы данных хорошо подходят для быстрой разработки систем и сервисов, работающих с по-разному структурированными данными. Они легко масштабируются и меняют структуру при необходимости. Однако такие базы данных теряют свою эффективность при решении задач, в которых требуется работа с множеством связанных объектов.

Графовые базы данных

Графовые базы данных — это тип нереляционных баз данных, предназначенный для хранения взаимосвязей между сущностями и навигации по ним. Для хранения сущностей используются узлы, а для хранения их взаимосвязей — ребра. В таких базах отсутствуют ограничения на количество и тип взаимосвязей, которые может иметь узел. Графовые базы данных используются для решении задач, имеющих сложные взаимосвязи между данными. При незначительном количестве связей и больших объемах данных графовые базы данных демонстрируют значительно более низкую производительность.

Для решения поставленной задачи была выбрана реляционная база данных с построчным хранением данных, так как:

- задача предполагает хранение структурированных и связанных данных;
- задача предполагает постоянное добавление и изменение данных;
- задача предполагает быструю отзывчивость на запросы пользователя;
- задача не предполагает выполнения сложных аналитических запросов.

Вывод

В данном разделе были выделены ролевые модели системы, конкретизированы хранимые данные и их связь между собой, построены соответствующие

диаграммы. Также был проведен анализ существующих на рынке решений, который позволил понять, какие особенности стоит добавить в разрабатываемый проект. Был осуществлен выбор модели базы данных.

2 Конструкторский раздел

В данном разделе будет спроектирована база данных, которая требуется для реализации поставленной задачи, и описаны ее ограничения. Также будет описана используемая ролевая модель.

2.1 Диаграмма проектируемой базы данных

На рисунке 2.1 представлена диаграмма проектируемой базы данных.

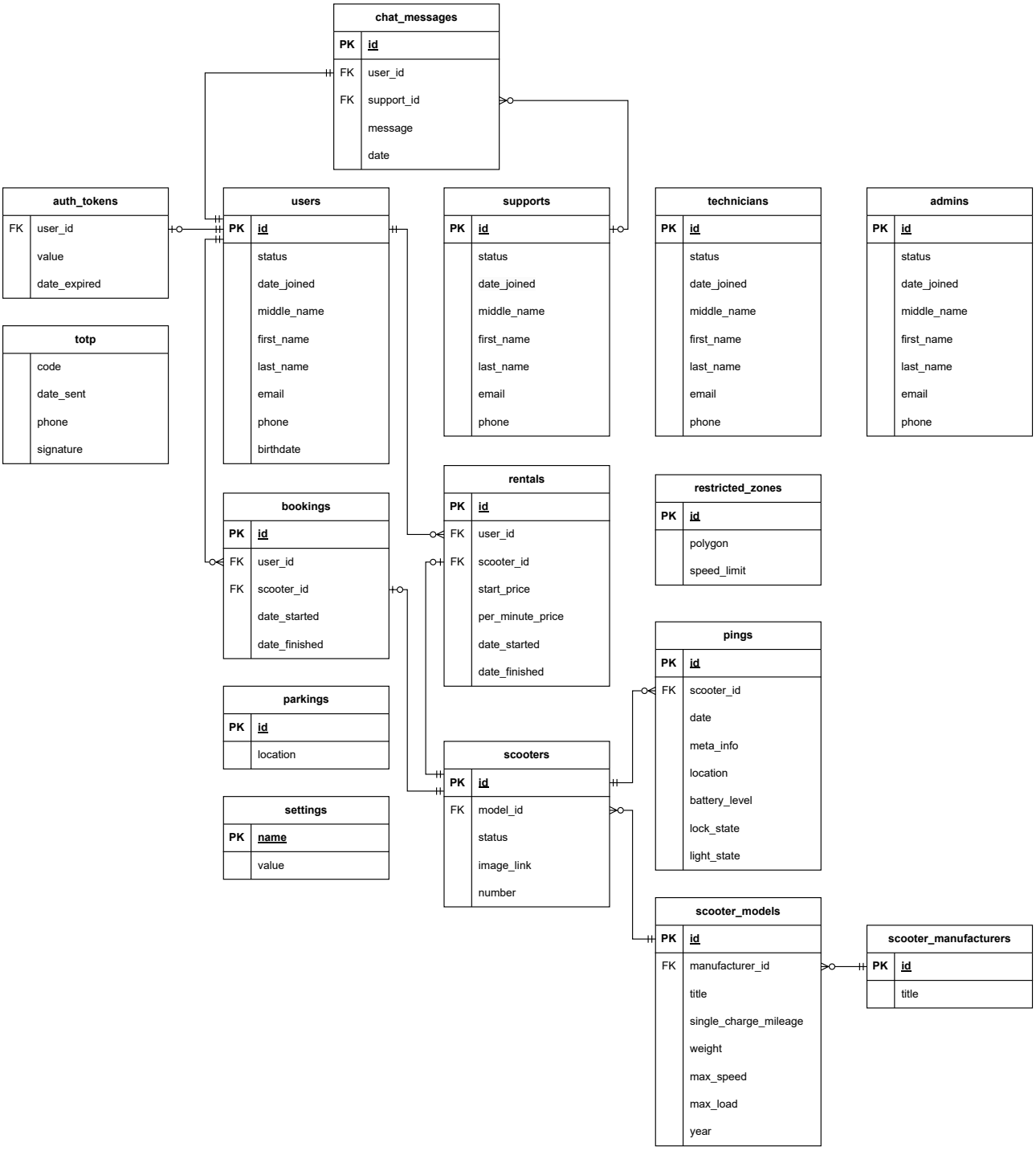

Рисунок 2.1 – Диаграмма проектируемой базы данных

2.2 Сущности проектируемой базы данных

2.2.1 Пользователь (таблица users)

Сущность пользователя содержит следующие поля.

1. id — уникальный идентификатор. Тип: число. Обязательное поле.
2. status — статус пользователя. Тип: строка. Обязательное поле. Может

принимать одно из двух возможных значений: «active» (активный) и «blocked» (заблокированный).

3. date_joined — дата регистрации пользователя в системе. Тип: дата со временем. Обязательное поле.
4. middle_name — отчество. Тип: строка.
5. first_name — имя. Тип: строка. Обязательное поле.
6. last_name — фамилия. Тип: строка.
7. email — адрес электронной почты. Тип: строка. Обязательное поле.
8. phone — номер телефона. Тип: строка. Обязательное поле.
9. birthdate — дата рождения. Тип: дата. Обязательное поле.

2.2.2 Сотрудник клиентской поддержки (таблица supports)

Сущность сотрудника клиентской поддержки содержит те же поля, что и сущность пользователя, за исключением даты рождения (birthdate). Поля отчества (middle_name) и фамилии (last_name) являются обязательными.

2.2.3 Механик (таблица technicians)

Сущность механика содержит те же поля, что и сущность пользователя, за исключением даты рождения (birthdate). Поля отчества (middle_name) и фамилии (last_name) являются обязательными.

2.2.4 Администратор (таблица admins)

Сущность администратора содержит те же поля, что и сущность пользователя, за исключением даты рождения (birthdate). Поля отчества (middle_name) и фамилии (last_name) являются обязательными.

2.2.5 Токен авторизации (таблица auth_tokens)

Сущность токена авторизации содержит следующие поля.

1. user_id — уникальный идентификатор пользователя. Тип: число. Обязательное поле.

2. value — хэш. Тип: строка. Обязательное поле.

3. date_expired — дата окончания действия. Тип: дата со временем. Обязательное поле.

2.2.6 Сообщение в чате (таблица chat_messages)

Сущность сообщения в чате содержит следующие поля.

1. id — уникальный идентификатор. Тип: число. Обязательное поле.

2. user_id — уникальный идентификатор пользователя, отправившего сообщение. Тип: число.

3. support_id — уникальный идентификатор сотрудника клиентской поддержки. Тип: число.

4. message — текст сообщения. Тип: строка. Обязательное поле.

5. date — дата отправки сообщения. Тип: дата со временем. Обязательное поле.

2.2.7 Бронирование (таблица bookings)

Сущность бронирования содержит следующие поля.

1. id — уникальный идентификатор. Тип: число. Обязательное поле.

2. user_id — уникальный идентификатор пользователя. Тип: число. Обязательное поле.

3. scooter_id — уникальный идентификатор электросамоката. Тип: число. Обязательное поле.

4. date_started — дата начала бронирования. Тип: дата со временем. Обязательное поле.

5. date_finished — дата окончания бронирования. Тип: дата со временем.

2.2.8 Аренда (таблица rentals)

Сущность аренды содержит следующие поля.

1. id — уникальный идентификатор. Тип: число. Обязательное поле.
2. user_id — уникальный идентификатор пользователя. Тип: число. Обязательное поле.
3. scooter_id — уникальный идентификатор пользователя. Тип: число. Обязательное поле.
4. start_price — стоимость начала аренды. Тип: деньги. Обязательное поле.
5. per_minute_price — стоимость аренды за минуту. Тип: деньги. Обязательное поле.
6. date_started — дата начала аренды. Тип: дата со временем. Обязательное поле.
7. date_finished — дата завершения аренды. Тип: дата со временем.

2.2.9 Зона ограничения скорости (таблица restricted_zones)

Сущность зоны ограничения скорости содержит следующие поля.

1. id — уникальный идентификатор. Тип: число. Обязательное поле.
2. polygon — координаты зоны. Тип: GeoJSON. Обязательное поле.
3. speed_limit — максимальная скорость. Тип: число. Обязательное поле.

2.2.10 Парковки (таблица parkings)

Сущность парковки содержит следующие поля.

1. id — уникальный идентификатор. Тип: число. Обязательное поле.
2. location — координаты парковки. Тип: GeoJSON. Обязательное поле.

2.2.11 Настройка (таблица settings)

Сущность настройки содержит следующие поля.

1. name — название. Тип: строка. Обязательное поле.
2. value — значение. Тип: JSON. Обязательное поле.

2.2.12 Электросамокат (таблица scooters)

Сущность электросамоката содержит следующие поля.

1. id — уникальный идентификатор. Тип: число. Обязательное поле.
2. model_id — уникальный идентификатор модели. Тип: число. Обязательное поле.
3. status — статус. Тип: строка. Обязательное поле. Может принимать одно из двух значений: «active» (активный) и «disabled» (деактивированный).
4. image_link — ссылка на фотографию самоката. Тип: строка.
5. number — номер самоката. Тип: строка. Обязательное поле.

2.2.13 Модель электросамоката (таблица scooter_models)

Сущность модели электросамоката содержит следующие поля.

1. id — уникальный идентификатор. Тип: число. Обязательное поле.
2. manufacturer_id — уникальный идентификатор производителя. Тип: число. Обязательное поле.
3. title — название модели. Тип: строка. Обязательное поле.
4. single_charge_mileage — пробег на одном заряде в километрах. Тип: число. Обязательное поле.
5. weight — масса электросамоката. Тип: число. Обязательное поле.

6. max_speed — максимальная скорость в километрах в час. Тип: число. Обязательное поле.

7. max_load — максимальная нагрузка в килограммах. Тип: число. Обязательное поле.

8. year — год выпуска. Тип: число. Обязательное поле.

2.2.14 Производитель электросамоката (таблица scooter_manufacturers)

Сущность производителя электросамоката содержит следующие поля.

1. id — уникальный идентификатор. Тип: число. Обязательное поле.

2. title — название. Тип: строка. Обязательное поле.

2.2.15 Запись в истории перемещений (таблица pings)

Сущность записи в истории перемещений содержит следующие поля.

1. id — уникальный идентификатор. Тип: число. Обязательное поле.

2. scooter_id — уникальный идентификатор самоката, отправившего информацию. Тип: число. Обязательное поле.

3. date — дата создания записи. Тип: дата со временем. Обязательное поле.

4. meta_info — информация о техническом состоянии электросамоката. Тип: JSON. Обязательное поле.

5. location — геолокация электросамоката. Тип: GeoJSON. Обязательное поле.

6. battery_level — уровень заряда батареи. Тип: число. Обязательное поле.

7. lock_state — состояние замка. Тип: строка. Обязательное поле. Может принимать одно из двух возможных значений: «locked» (закрыт) и «unlocked» (открыт).

8. light_state — состояние фар. Тип: строка. Обязательное поле. Может принимать одно из двух возможных значений: «enabled» (включены) и «disabled» (выключены).

2.3 Ограничения целостности базы данных

TODO

2.4 Описание триггера

На рисунке 2.2 представлена схема алгоритма работы триггера, который не позволяет удалять пользователей, которые имеют активные (незавершенные) аренды.

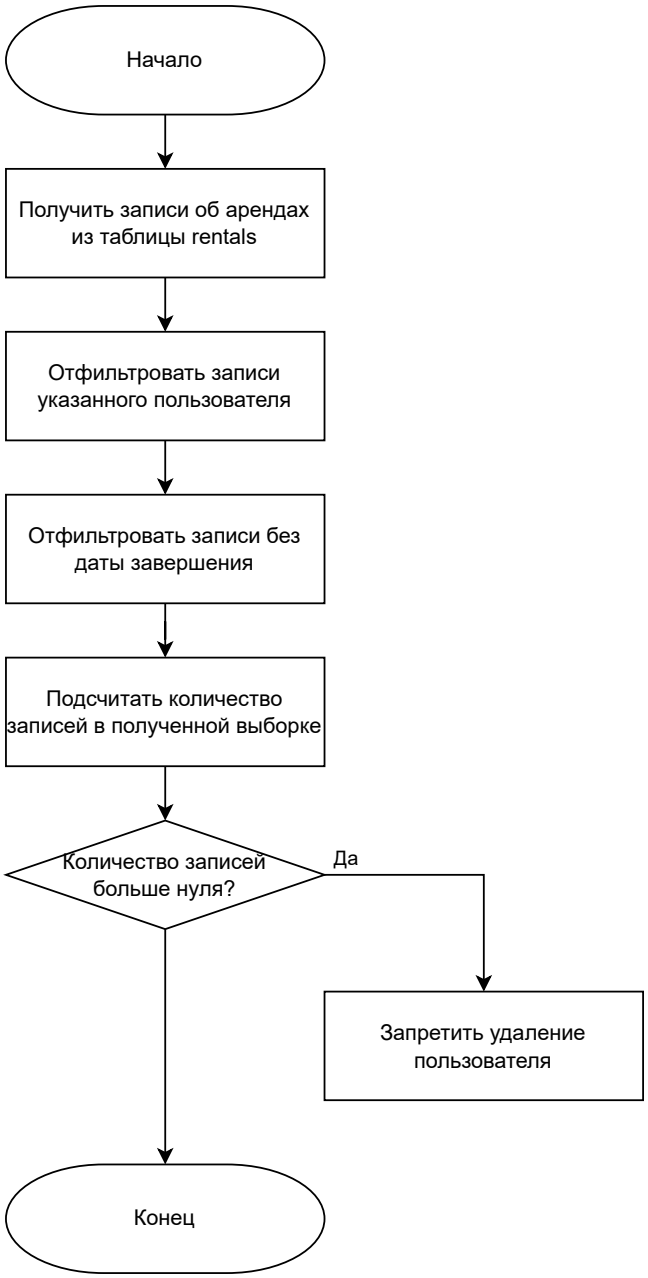

Рисунок 2.2 – Схема алгоритма работы триггера

2.5 Ролевая модель

TODO

Вывод

В данном разделе была спроектирована база данных для разрабатываемого приложения, а так же рассмотрены способы обеспечения ее целостности и ролевая модель.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. *Пологойко М. Д.* Перспективы использования сервисов проката электросамокатов в повседневных перемещениях по городу // Скиф. Вопросы студенческой науки. — 2021. — Т. 57, № 5. — С. 315—319.
2. FAQ о прокате электросамокатов Whoosh. Велопрокат. — Режим доступа: `https://whoosh-bike.ru/faq` (дата обращения: 25.03.2023).
3. Прокат самокатов в Москве — ТОП-7 сервисов по аренде самокатов в Москве. — Режим доступа: `https://arenda-samokatov.ru/prokat-samokatov-v-moskve/` (дата обращения: 19.03.2023).
4. Шеринг самокатов Whoosh, скачать приложение Whoosh здесь! — Режим доступа: `https://whoosh-bike.ru/` (дата обращения: 26.03.2023).
5. Шеринг самокатов и велосипедов Юрент | Скачать приложение Urent. — Режим доступа: `https://urent.ru/` (дата обращения: 26.03.2023).
6. Самокаты — Яндекс Go. — Режим доступа: `https://go.yandex/ru_ru/lp/rides/scooter` (дата обращения: 26.03.2023).
7. *Федотова С. Н.* Цифровизация транспортно-логистических услуг // Экономика и бизнес: теория и практика. — 2019. — Т. 57, № 11—3. — С. 124—127.