Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ

УНИВЕРСИТЕТ»

**Отчёт**

по предмету «Проектирование программного обеспечения»

лабораторная работа №4

«Объектно ориентированное моделирование. UML диаграммы поведения»

тема «Интерфейс сети автопарковок»

Студент: Беласин Д.А.

ФИТ 3 курс 1 группа

Преподаватель: Якубенко К. Д.

Минск 2024

# 1. Постановка задачи

Функциональное назначение интерфейса сети автопарковок «ParkSmart» заключается в предоставлении удобной и многофункциональной платформы для аренды парковочных мест на короткий срок. Интерфейс должен обеспечивать пользователям возможность легко находить доступные парковочные места, арендовать их и управлять своим процессом аренды, а также отслеживать информацию по завершенным арендам.

Программа систематизирует данные о доступных парковочных местах, их местоположении, стоимости аренды и статусе, что позволяет пользователям делать осознанный выбор. С помощью интерактивной карты интерфейс предоставляет информацию о ближайших доступных местах в реальном времени.

Интерфейс предлагает пользователям возможность создавать личные кабинеты, где они могут управлять своими арендами, отслеживать историю использования, пополнять счет и получать уведомления о доступных бонусах и акциях. Функционал корзины дает возможность добавлять места в аренду, резервировать их на определенное время и оплачивать несколько арендуемых мест одновременно.

Для операторов парковок интерфейс предоставляет широкий функционал, включая управление парком парковочных мест, мониторинг их статуса, отслеживание активности пользователей и управление статистикой аренды через специальную панель администратора.

В интерфейсе предусмотрена гибкая система тарифов, основанная на длительности аренды и типе парковочного места, а также возможность возврата автомобиля на любые станции сети, что делает процесс аренды удобным и гибким для пользователей.

# 1.1 Описание объектов системы

Описание объекта «парковочное место»:

* Это пространство, которое можно арендовать через приложение для временной парковки автомобиля.
* Основные атрибуты парковочного места: идентификационный номер, текущее местоположение, доступность для аренды, тип места (стандартное, электрическое), стоимость аренды.
* Связи: Парковочное место принадлежит сети аренды и может быть арендовано пользователями.

Описание объекта «оператор»:

* Это лицо или организация, управляющее сетью парковочных мест, следящее за их состоянием, доступностью и обработкой платежей.
* Основные атрибуты оператора: имя компании, контактная информация, список доступных парковочных мест и станции аренды.
* Связи: оператор создаёт и управляет парком парковочных мест, отслеживает их состояние и взаимодействует с пользователями через приложение.

Описание объекта «пользователь»:

* Это физическое лицо, которое может арендовать парковочное место, следить за своими поездками и управлять арендой через личный кабинет.
* Основные атрибуты пользователя: имя, контактные данные, история аренды, текущие бронирования.
* Связи: пользователь арендует парковочные места, управляет своими арендами и взаимодействует с платёжной системой.

Описание объекта «аренда»:

* Это процесс, который позволяет пользователю взять парковочное место на определённый срок и оплатить его использование.
* Основные атрибуты аренды: уникальный идентификатор, продолжительность, дата начала и завершения, стоимость аренды, статус (активная, завершена).
* Связи: аренда привязана к пользователю и парковочному месту.

Описание объекта «платёж»:

* Это финансовая транзакция, выполняемая пользователем для оплаты аренды парковочного места.
* Основные атрибуты платежа: сумма, способ оплаты (банковская карта, электронные кошельки), статус (успешный, неудачный), дата проведения.
* Связи: платёж связан с арендой и пользователем.

Программа должна поддерживать следующие функции:

* Регистрация и авторизация пользователей.
* Личный кабинет пользователя должен включать историю аренд и доступ к текущим арендам.
* Отображение доступных парковочных мест на карте с информацией о местоположении, типе и статусе.
* Процесс аренды парковочного места должен включать выбор пункта аренды и типа места с возможностью оплаты через приложение.
* Операторы должны иметь личный кабинет с доступом к управлению парком парковочных мест и мониторингу активности.
* Поиск и бронирование парковочных мест: пользователи смогут искать доступные парковочные места через карту.

Процесс аренды и возврата: процесс аренды будет простым и интуитивным: пользователь выбирает парковочное место, завершает бронирование и оплачивает аренду через приложение. Возврат автомобиля будет возможен на любое свободное место в сети.

# 2. Описание программных средств

Описание программного средства Draw.io:

* Название: Draw.io.
* Версия: Актуальная версия (по состоянию на 2024 год). Программа обновляется регулярно.
* Разработчик: JGraph Ltd.
* Адрес загрузки: https://app.diagrams.net/. Draw.io доступен как онлайн-приложение, так и в виде настольного клиента для различных операционных систем.
* Режим использования: Draw.io предоставляет бесплатный доступ ко всем своим функциям без ограничений. Это полностью бесплатный инструмент с открытым исходным кодом, доступный как для личного, так и для корпоративного использования.
* Платформы: Draw.io доступен как веб-приложение, которое работает в любом современном браузере. Также доступны настольные версии для Windows, macOS и Linux.
* Типы моделей, с которыми работает Draw.io: Draw.io используется для создания диаграмм, схем, моделей процессов и прочих визуальных структур. Программа поддерживает работу с блок-схемами, диаграммами UML, IDEF0, сетевыми диаграммами, архитектурными схемами, организационными диаграммами и многими другими визуальными представлениями данных.
* Основные функции: Создание и редактирование диаграмм различных типов (например, UML, блок-схемы, BPMN, ERD). Поддержка импорта и экспорта файлов в различных форматах, включая XML, PNG, SVG и PDF. Возможность совместной работы с другими пользователями через облачные сервисы (Google Drive, OneDrive, Dropbox и др.). Функции автосохранения и интеграции с популярными системами управления версиями. Поддержка версии оффлайн, которая позволяет работать без подключения к интернету. Гибкая система настройки шаблонов и библиотек элементов для ускорения работы.

Draw.io — это удобный, бесплатный инструмент для создания диаграмм и схем, который поддерживает совместную работу и интеграцию с облачными сервисами, обеспечивая гибкость и доступность на различных платформах.

# Описание практического задания

UML диаграмма прецедентов (вариантов использования) иллюстрирует возможные сценарии внешнего взаимодействия пользователей (Actors) с прецедентами (Use cases) системы, т.е. описывает функциональное назначение системы. Кроме того, могут быть показаны отношения между вариантами использования, группировка прецедентов через роли и приведены комментарии в случае необходимости.

Actor — это типовые пользователи (менеджер, операторы и т. д.). Актер представляет собой некоторую роль, которую играет пользователь (или другая система) по отношению к системе.

Возможные отношения между вариантами использования:

* Отношения Include (Включения) — используется, когда имеется какой-либо фрагмент поведения системы, который повторяется более чем в одном варианте использования и необходимо, чтобы его описание копировалось в каждом из этих вариантов использования.
* Отношения Extend (Расширения) — расширяющий вариант использования может дополнять (делать более точным) поведение базового варианта использования, определив в базовом варианте использования точки расширения.

Система включает четыре основные категории пользователей (актеров) с различными правами доступа и обязанностями:

* Гость — пользователь, который не имеет учетной записи и ограничен в доступе к функциям.
* Пользователь — зарегистрированный пользователь, имеющий доступ к демонстрационным данным, например, о ближайших парковочных местах и их доступности.
* Оператор — зарегистрированный пользователь, который занимается мониторингом и управлением парковкой, а также поддержкой пользователей.
* Администратор — пользователь с расширенными правами, отвечающий за управление учётными записями и безопасность системы.

Актёры в системе «Интерфейс сети автопарковок» имеют иерархическую структуру:

* Гость — базовый уровень доступа, с минимальным набором действий.
* Пользователь — наследует действия Гостя, добавляя функции бронирования и оплаты парковочных мест.
* Оператор — наследует действия Пользователя и добавляет функции, связанные с контролем состояния мест, их расположения и доступности в режиме реального времени, а также помощь клиентам в решении вопросов, связанных с бронированием.
* Администратор — наследует действия Оператора и добавляет функции, связанные с администрированием пользователей.

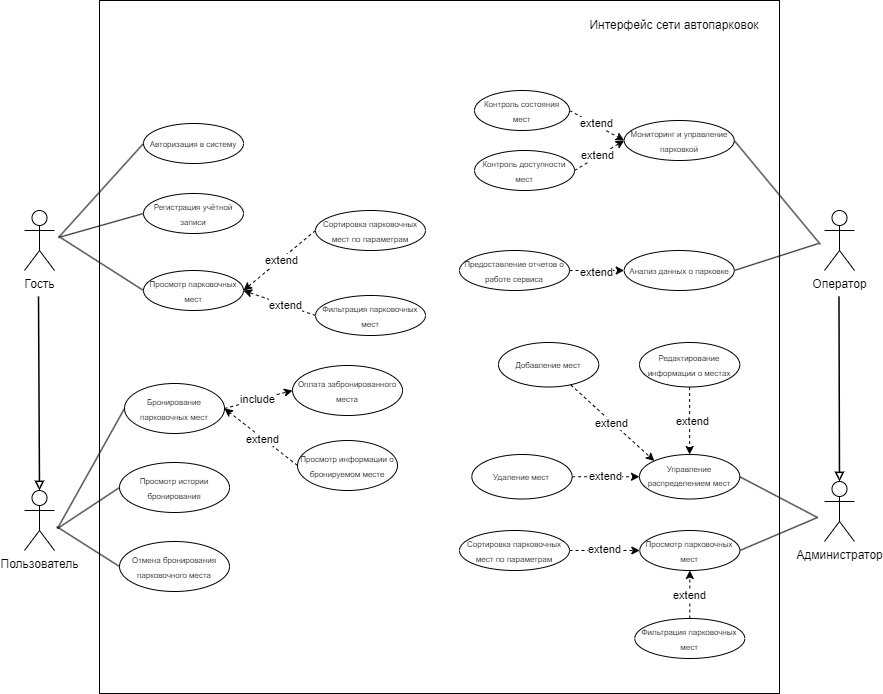


Рисунок 3.1 – Контекстная диаграмма

Диаграмма демонстрирует основные действия доступные актёрам системы.

В режиме «Гость» пользователю доступны следующие функции:

* Регистрация учетной записи — возможность создать учетную запись для полноценного доступа.
* Авторизация в систему — возможность осуществлять вход в систему с использованием своей учётной записи.
* Поиск парковочных мест — осуществление поиска доступных парковочных мест для аренды.

В режиме «Пользователь» пользователю доступны следующие функции:

* Аренда парковочных мест — возможность осуществлять аренду доступных парковочных мест
* Создание и просмотр отчётов — создание и просмотр отчётов о своих парковках и затратах.
* Отмена бронирования парковочного места .
* Просмотр истории бронирования мест — предоставляет возможность пользователю просмотреть ранее забронированные места.

В режиме «Оператор» пользователю доступны следующие функции:

* Мониторинг и управление парковкой — контроль состояния мест, их расположения и доступности в режиме реального времени.
* Поддержка пользователей — помощь клиентам в решении вопросов, связанных с бронированием, включая технические проблемы и ситуации с задержками.
* Анализ данных — предоставление отчетности о работе сервиса и использования мест.

В режиме «Администратор» пользователю доступны следующие функции:

* Настройка резервного копирования данных для обеспечения сохранности данных.
* Восстановление после сбоев для минимизации простоев и потерь данных.
* Управление распределением мест и их перемещением между парковками.
* Полный контроль над всеми операциями по бронированию и оплате парковочных мест.

Эти действия составляют основу для Use Case диаграммы и позволяют выделить основные сценарии использования системы, обеспечивая четкое понимание взаимодействий между различными ролями и функциональностью сервиса. Используя диаграмму, можно лучше визуализировать, как каждый пользователь взаимодействует с системой, а также выявить ключевые требования и ожидания пользователей. Кроме того, Use Case диаграммы способствуют идентификации заинтересованных сторон, уточнению требований, улучшению коммуникации и обеспечению гибкости разработки. Имея четкое представление о сценариях использования, команда может быстрее адаптироваться к изменениям в требованиях или приоритетах. Таким образом, Use Case диаграммы не только упрощают процесс проектирования, но и служат важным инструментом для управления проектом на всех его этапах.

# Теоретические вопросы подготовки к лабораторной работе

1. **Unified Process (UP)**:

* Unified Process — это методология разработки программного обеспечения, основанная на объектно-ориентированном подходе.
* UP описывает процесс разработки в виде итераций и инкрементов, что позволяет адаптироваться к изменениям требований на протяжении всего жизненного цикла проекта.
* Методология включает четыре основных фазы: **инициация**, **элеборация**, **конструкция** и **передача**. Каждая фаза состоит из итераций, которые могут включать в себя анализ, проектирование, разработку и тестирование.

**UML (Unified Modeling Language)**:

* UML — это стандартный язык моделирования, используемый для визуализации, спецификации, конструирования и документирования систем.
* UML поддерживает объектно-ориентированное проектирование и предоставляет набор графических нотаций для описания различных аспектов системы, включая ее структуру, поведение и взаимодействия.

### 2. Основные диаграммы UML 2.0

**Структурные диаграммы**:

* Диаграмма классов (Class Diagram).
* Диаграмма объектов (Object Diagram).
* Диаграмма компонентов (Component Diagram).
* Диаграмма развёртывания (Deployment Diagram).
* Диаграмма пакетов (Package Diagram).
* Диаграмма структуры (Composite Structure Diagram).

**Диаграммы поведения**:

* Диаграмма вариантов использования (Use Case Diagram).
* Диаграмма последовательностей (Sequence Diagram).
* Диаграмма взаимодействия (Communication Diagram).
* Диаграмма состояний (State Machine Diagram).
* Диаграмма активности (Activity Diagram).

### 3. CASE-средства, поддерживающие создание UML диаграмм . Среди широко используемых CASE-средств, поддерживающих создание UML-диаграмм, можно выделить:

* **IBM Rational Rose.**
* **Visual Paradigm.**
* **Enterprise Architect.**
* **StarUML.**
* **ArgoUML.**
* **Lucidchart.**
* **Draw.io.**

### 4. Назначение диаграммы вариантов использования

Диаграмма вариантов использования (Use Case Diagram) предназначена для:

* **Определения функциональности системы**: Она показывает, какие функции доступны пользователям системы и как они могут взаимодействовать с системой.
* **Идентификации участников**: Диаграмма помогает выявить, кто будет взаимодействовать с системой (актеры), и какие сценарии использования они будут выполнять.
* **Общего представления о системе**: Предоставляет высокоуровневое представление о системе, что позволяет заинтересованным сторонам быстро понять основные функции и взаимодействия.

### 5. Нотации для построения Use-Case диаграммы

Для построения диаграммы вариантов использования используются следующие нотации:

* **Актер**: Представляется в виде человечка (стикмана), обозначает пользователя или другую систему, взаимодействующую с рассматриваемой системой.
* **Вариант использования**: Изображается в виде овала, который содержит название функции или сценария использования.
* **Связь (Ассоциация)**: Обозначается сплошной линией, соединяющей актера с вариантом использования, указывая на взаимодействие.
* **Обобщение**: Используется для обозначения отношений между актерами или вариантами использования и представляется с помощью стрелки с пустым треугольником, указывающей на родительский элемент.
* **Включение**: Показывает, что один вариант использования всегда включает выполнение другого, обозначается стрелкой с пунктирной линией и «include».
* **Расширение**: Показывает, что один вариант использования может быть расширен другим в определенных условиях, также обозначается стрелкой с пунктирной линией и «extend».