Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Лабораторная работа по предмету «Проектирование информационных систем» на тему «Объектно-ориентированное модулирование. Диаграммы поведения UML»

Студент: Коршун Н.И.

ФИТ 4 курс 5 группа

Преподаватель: Якубенко К. Д.

Минск 2024

# 1 Постановка задач

Темой данной лабораторной работы является построение UML-диаграмм поведения (на выбор: деятельности, последовательности, состояний) веб-приложения каталог автомобилей «AUTOSTOP», описание которого было представлено в лабораторной работе №1.

Web-приложение должно быть выполнено с использованием асинхронного программирования, взаимодействовать с базой данных, реализовано под разными платформами. Web-приложение должно представлять собой web-приложение с асинхронным UI с использованием фреймворка NextJS. Отображение, бизнес-логика и хранилище данных должны быть максимально независимы друг от друга для возможности расширения. Диаграмму вариантов использования разработать на основе UML, также необходимо разработать логическую схему базы данных и структурную схему приложения. Язык разработки проекта JavaScript, платформа «NodeJS». Развертывание конечного приложения для последующего использования должно осуществляться с использованием Docker.

Функционально web-приложение должно:

* обеспечивать возможность просмотра информации о различных моделях автомобилей (характеристики, фотографии);
* предоставлять функционал просмотра видеообзоров на автомобили;
* позволять пользователю сравнивать характеристики нескольких автомобилей;
* обеспечивать регистрацию и авторизацию пользователей с возможностью восстановления пароля через электронную почту
* поддерживать роли гостя, пользователя, администратора;
* для гостей доступен просмотр автомобилей и видеообзоров;
* для зарегистрированных пользователей доступен функционал сохранения избранных автомобилей и добавление автомобилей в список для сравнения;
* поддерживать функционал торговой площадки для продажи и покупки автомобилей с возможностью размещения объявлений, загрузки фотографий и контактов;
* администратор может добавлять и редактировать информацию о автомобилях, видеообзорах и управлять пользователями;
* поддерживать фильтрацию и поиск по различным параметрам автомобилей (марка, год выпуска, тип двигателя и т.д.).

Основные задачи приложения:

* разработать удобный интерфейс для поиска информации о автомобилях;
* предоставить функционал для просмотра видеообзоров на автомобили;
* создать систему сравнения характеристик автомобилей для более детального анализа.

# 2 Описание программных средств

Draw.io является мощным инструментом для создания диаграмм и визуального моделирования, который использовался для разработки и документирования архитектуры системы в процессе выполнения лабораторной работы. Приложение предоставило интуитивно понятный графический интерфейс для создания структурных схем, которые помогли визуализировать ключевые компоненты системы, а также их взаимосвязи и процессы взаимодействия. Использование draw.io стало важным шагом для проектирования системы, так как оно упростило процесс создания диаграмм, описывающих функциональные блоки приложения и их взаимодействие. Приложение также поддерживает интеграцию с различными облачными сервисами, что облегчило совместную работу над схемами и предоставило гибкость в управлении проектами.

Инструмент draw.io был выбран за его следующие особенности:

* Доступность: кроссплатформенная, поддержка работы в браузере, а также на Windows, macOS и Linux.
* Поддержка стандартов: возможность создания UML–диаграмм, блок–схем, диаграмм ER, IDEF0 и IDEF3, что позволило эффективно визуализировать бизнес–процессы и архитектуру системы.
* Легкость в использовании: интуитивно понятный интерфейс, который позволяет создавать диаграммы без необходимости владения сложными графическими навыками.
* Интеграция с облачными хранилищами: поддержка Google Drive, OneDrive, GitHub, что упрощает доступ к схемам и их совместное редактирование.

Основное назначение draw.io в этом проекте заключалось в создании визуальных моделей бизнес–процессов и архитектурных решений, которые легли в основу разработки системы. Эти диаграммы помогли в структурировании процессов разработки и визуализации потоков данных, что обеспечило более четкое понимание системы в целом.

Данные о технологии draw.io:

* Разработчик: JGraph Ltd.
* Адрес загрузки: https://app.diagrams.net.
* Использовался для: создания диаграмм и визуальных моделей бизнес–процессов и архитектурных решений системы.
* Доступность: кроссплатформенная, поддержка веб–версии и настольных приложений.

# 3 Описание практического задания

Описание практического задания начнется с диаграммы деятельности, которая представлена на рисунке 3.1.

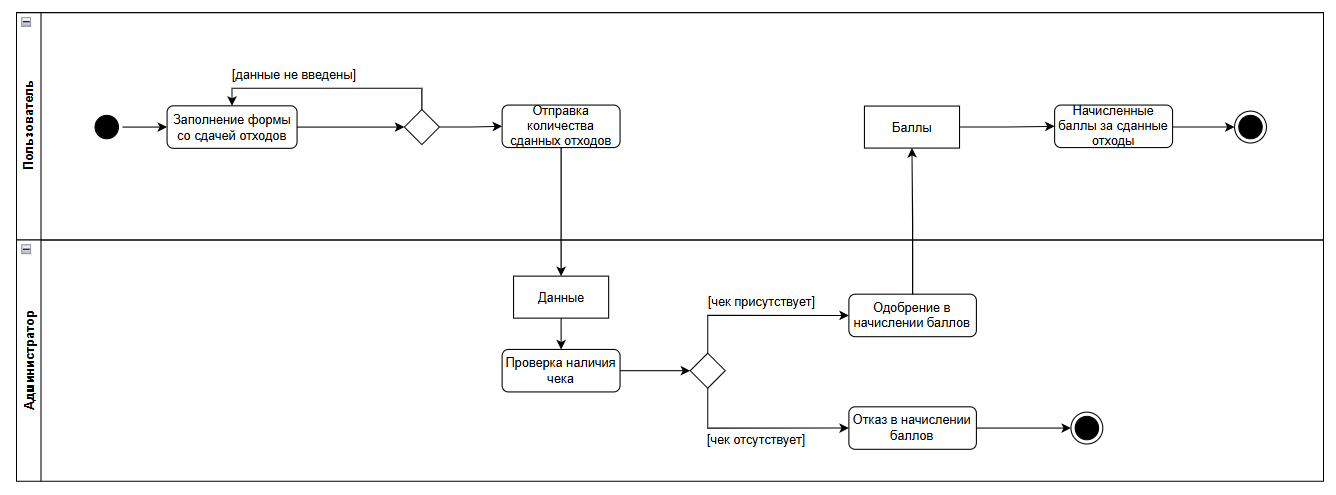


Рисунок 3.1 – Диаграмма деятельности

Диаграммы деятельности – представляют собой графическое представление рабочих процессов поэтапных действий и действий с поддержкой выбора, итерации и параллелизма.

На данной диаграмме деятельности можем наблюдать процесс работы с получением баллов за сдачу отходов, он состоит из 5 шагов, следовательно, первым из них будет заполнение формы с количеством сданных отходов, далее отправка этих данных администратору для подтверждения. Администратор выполняет проверку подлинности предоставленного чека о сдаче отходов, если чек достоверен администратор подтверждает начисление баллов и эти баллы начисляются пользователь, в противном случаем пользователю будет отказано в начислении баллов.

В диаграмме деятельности применялись такие элементы как:

* принятие решения;
* активное состояние;
* начальное состояния;
* конечное состояние;
* объект в состоянии;
* переход.

Таким образом, диаграмма деятельности наглядно иллюстрирует последовательность действий в процессе получения баллов за сдачу отходов, подчеркивая важность каждого этапа. Она демонстрирует, как пользователи и администраторы взаимодействуют в рамках рабочего процесса, а также как принимаются решения на основе проверки данных. Каждый шаг логически вытекает из предыдущего, что обеспечивает ясность и предсказуемость процесса.

Следующая диаграмма – это диаграмма состояний, которая представлена на рисунке 3.2.

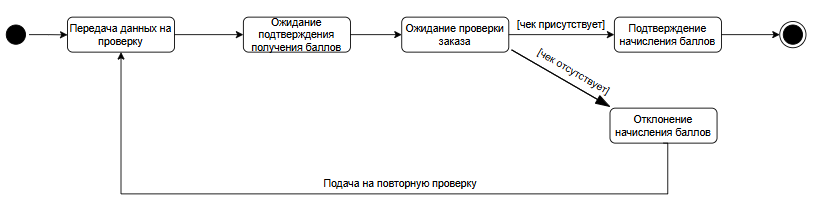


Рисунок 3.2 – Диаграмма состояний

Диаграммы состояний служат для моделирования динамических аспектов системы. Данная диаграмма полезна при моделировании жизненного цикла объекта. От других диаграмм диаграмма состояний отличается тем, что описывает процесс изменения состояний только одного экземпляра определенного класса - одного объекта, причем объекта реактивного, то есть объекта, поведение которого характеризуется его реакцией на внешние события.

Для диаграммы состояний был выбран непосредственно сам процесс начисления баллов. Первое состояние передача данных на проверку. После передачи данных на проверку объект переходит в состояние ожидания начисления, где выполняется обработка на соответствие критериям. В зависимости от результата обработки, возможны два исхода: подтверждение начисления, которое переводит объект в состояние «Баллы начислены», или отклонение с указанием причины, завершающее процесс.

В диаграмме состояний применялись такие элементы как:

* класс;
* состояние;
* начальное состояние;
* конечное состояние;
* переход.

Таким образом, диаграмма наглядно демонстрирует возможные пути переходов между состояниями в зависимости от внешних событий, моделируя реакцию объекта на различные ситуации и изменения условий. Диаграмма также подчеркивает важность передачи данных между компонентами системы для обеспечения безопасности и точности. Каждый шаг в процессе тщательно синхронизирован, чтобы минимизировать задержки и ошибки. В случае некорректных данных, пользователь получает уведомление об ошибке и может повторить попытку снова.

# 4 Ответы на вопросы

1. Укажите виды диаграмм поведения. Какая между ними связь?

Диаграммы поведения в UML условно можно разделить на пять типов в соответствии с основными способами моделирования динамики системы.

Диаграммы прецедентов – описывают организацию поведения системы.

Диаграммы последовательностей – акцентируют внимание на времени упорядоченности сообщений.

Диаграммы кооперации сфокусированы на структурной организации объектов, посылающих и получающих сообщения.

Диаграммы состояний – описывают изменение состояния системы в ответ на события.

Диаграммы деятельности – демонстрируют передачу управления от одной деятельности к другой.

2. Опишите назначение диаграммы деятельности.

Диаграмма деятельности — UML-диаграмма, на которой показаны действия, состояния которых описаны на диаграмме состояний. Диаграммы деятельности используются при моделировании бизнес-процессов, технологических процессов, последовательных и параллельных вычислений.

3. Опишите основные нотации, которые используются на диаграмме состояний.

Таблица 4.1 – Таблица нотаций

|  |  |
| --- | --- |
| Элемент/Нотация | Предназначение |
| Пример | Класс (Class) |
| Пример | Состояние (State) |
| Пример | Состояние (StateEx) |
| Пример | Составное состояние (Composite state) |
| Пример | Разделитель (Concurrent state) |
| Пример | История (History) |
| Пример | Глубокая история (Deep history) |
| Пример | Начальное состояние (Start state) |
| Пример | Конечное состояние (Final state) |
| ПримерПример | Синхронизатор/разветвитель (Complex transition) |
| Пример | Переход (Transition) |
| Пример | Сообщение (Event message) |
| Пример | Точка изгиба связей (Point) |

|  |  |
| --- | --- |
| Элемент/Нотация | Предназначение |
| Пример | Комментарий (Note) |
| Пример | Коннектор комментария (Note connector) |

4. Укажите виды связей между объектами на диаграмме последовательностей.

Синхронное сообщение — отправитель передаёт ход управления актёру-получателю, которому необходимо провести в прецеденте некоторое действие. Пока проводимое получателем действие не будет завершено (не будет получено ответное сообщение), отправитель теряет возможность производить какие-либо действия. Графически изображается как сплошная линия со стрелкой в виде закрашенного треугольника, после которой идёт прямоугольник, отражающий деятельность объекта, в конце которого находится ответное сообщение.

Ответное сообщение — данное сообщение является ответом на синхронное сообщение. Обычно, содержит какое-либо возвращаемое изначальному отправителю значение, также возвращающее ему управление (возможность действовать). Графически изображается пунктирной линией с открытой стрелкой.

Асинхронное сообщение — отправитель передаёт ход управления получателю, которому необходимо провести в прецеденте некоторое действие. Основное отличие от синхронного сообщения состоит в том, что отправитель не теряет возможности совершать другие действия. Графически изображается сплошной линией с открытой стрелкой.

Потерянное сообщение — сообщение без адресата.

Найденное сообщение — сообщение без отправителя.

Последние два вида стрелок (взаимодействий) используются крайне редко. В основном они используются для демонстрации взаимодействия имеющихся объектов в данном прецеденте с внешними системами.

5. Какая диаграмма позволяет моделировать параллельные вычисления?

Диаграммы деятельности используются при моделировании бизнес-процессов, технологических процессов, последовательных и параллельных вычислений.