Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Лабораторная работа №7

«Разработка диаграммы поведения»

Студент: Беласин Д.А.

ФИТ 3 курс 1 группа

Преподаватель: Якубенко К.Д.

Минск 2024

1. **Постановка задачи**

Задачей данной работы является детальное описание структуры взаимодействия компонентов, используемых в web-приложении, на основе данных, описанных в лабораторной работе №1.

Это предполагает построение точной и понятной модели, которая поможет глубже понять, как компоненты приложения взаимодействуют друг с другом, какие функции они выполняют, какие интерфейсы предоставляют и требуют, а также как они развертываются в целевой среде.

Диаграмма деятельности — это важнейший элемент UML, позволяющий проектировщикам, разработчикам и другим участникам процесса разработки наглядно представлять последовательность действий системы. Диаграмма последовательности играет ключевую роль в разработке приложений, так как она демонстрирует:

* Логику приложения: не считая как организованы его компоненты (библиотеки, исполняемые файлы, файлы конфигурации и т.д.).
* Взаимодействие компонентов: какие компоненты связаны между собой и каким образом (через интерфейсы, зависимости и т.д.).
* Последовательность действий: что именно клиенту будет необходимо сделать.
* Зависимости между компонентами: какие компоненты зависят от других компонентов и какие элементы используются для взаимодействия.

Диаграмма состояний — это важнейший элемент UML.

Она показывает, как объект переходит из одного состояния в другое. 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.

Задачей данной работы является построение диаграммы деятельности и диаграммы состояний для интерфейса аренды парковочных мест, что включает в себя определение узлов, состояний, событий нашей системы.

Разработка таких диаграмм помогает визуализировать логику работы и последовательность работы приложения, обеспечить правильное взаимодействие компонентов и подготовить основу для эффективного взаимодействия и масштабирования.

1. **Описание программных средств**

Для построения моделей в данной работе использовалось программное средство Draw.io, которое представляет собой мощный инструмент для создания диаграмм и различных графических схем.

Этот инструмент широко используется для разработки UML-диаграмм, блок-схем, интеллект-карт, бизнес-макетов, схем сущностей и отношений, а также для других задач, связанных с визуализацией и проектированием.

Draw.io — это онлайн-сервис и настольное приложение с открытым исходным кодом, которое позволяет создавать и редактировать диаграммы разного уровня сложности. Программа является бесплатной, что делает её доступной для широкого круга пользователей, начиная от студентов и заканчивая профессиональными разработчиками и инженерами.

Draw.io поддерживает множество функций, которые делают работу с диаграммами быстрой и удобной. Сервис распространяется на бесплатной основе с открытым исходным кодом.

Особенности Draw.io:

1. Более 500 шаблонов элементов и фигур.
2. Облегчённый интерфейс, в котором за короткий промежуток времени можно создать готовый проект.
3. Поддержка горячих клавиш, задействованных в большинстве графических редакторов.
4. Экспорт в форматы: JPG, PNG, SVG, VDSX.
5. Возможность совместной работы.
6. Наличие различных фоновых тем.
7. Мультиязычный интерфейс.

Программное средство draw.io также поддерживает импорт файлов в различных форматах, включая .vsdx (формат Microsoft Visio), Gliffy™ и Lucidchart™. Это позволяет вам работать с существующими диаграммами, созданными в других инструментах, и продолжать их редактирование и доработку в draw.io.

Cервис поддерживает множество функций, которые делают работу с диаграммами быстрой и удобной. Сервис распространяется на бесплатной основе с открытым исходным кодом.

1. **Практическое задание**

Диаграмма деятельности для нашего проекта будет выглядеть следующим образом – рисунок 4.1.

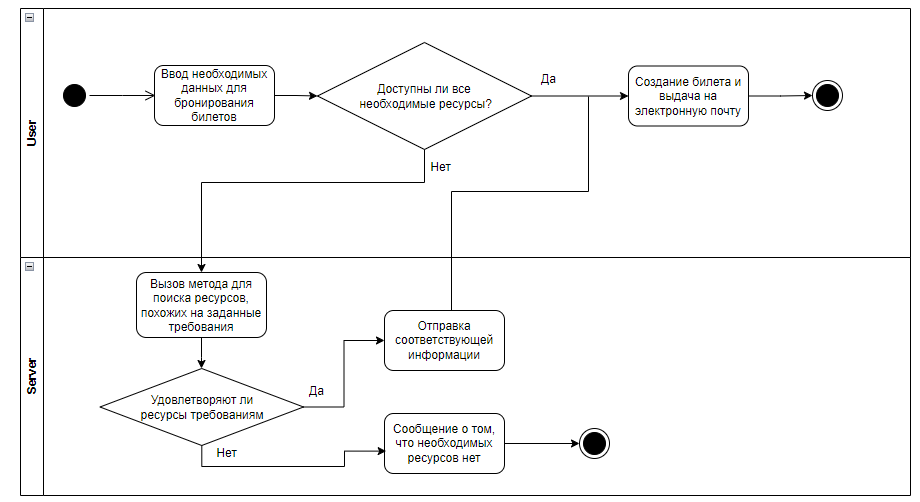


Рисунок 3.1 – Диаграмма деятельности

Разберем элементы диаграммы деятельности:

* Начало.
* Действие.
* Условие.
* Клиент.
* Сервер.
* Конец.

Диаграмма деятельности может стать эффективным инструментом для проектирования и документирования интерфейса планирования мероприятий. Она позволяет визуально представить последовательность действий, которые пользователь должен выполнить для организации события.

Одним из ключевых преимуществ использования диаграммы деятельности является ее способность наглядно показать различные сценарии взаимодействия пользователя с системой. Это помогает выявить потенциальные точки принятия решений, возможные ветвления процесса и обеспечить логичный и интуитивно понятный поток действий.

Разработчики могут использовать эту информацию для создания более удобного и эффективного интерфейса, который будет отвечать потребностям пользователей.

Далее рассмотрим диаграмму состояния для тех. поддержки на рисунке 4.2:

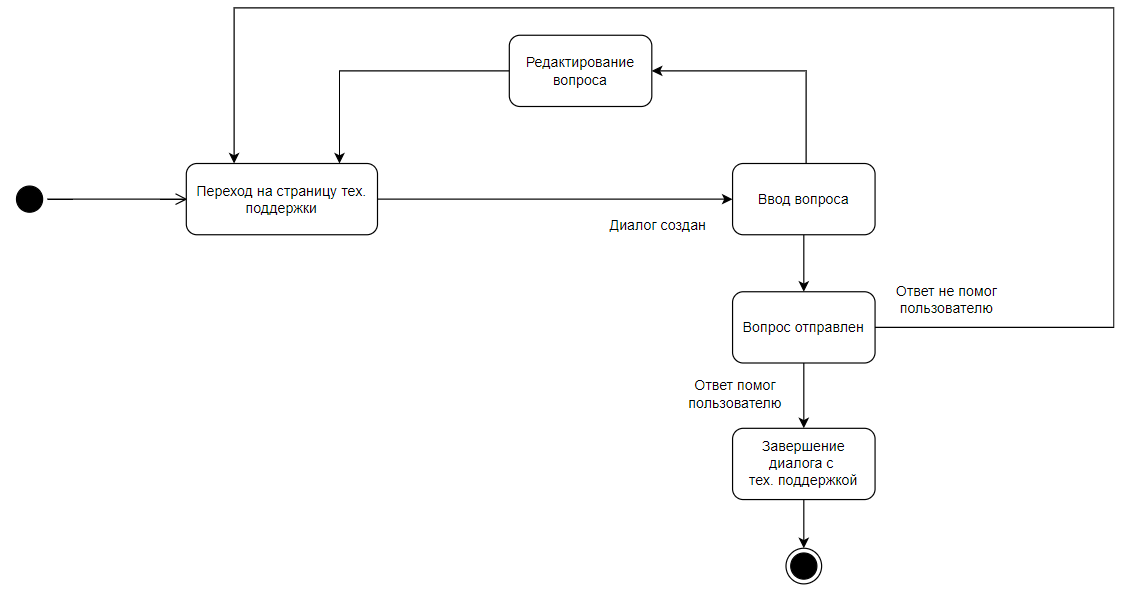


Рисунок 3.2 – Диаграмма состояния

Элементы диаграммы состояния:

* Переход на страницу тех. поддержки.
* Редактирование вопроса.
* Диалог создан.
* Ввод вопроса.
* Вопрос отправлен.
* Ответ не помог пользователю.
* Ответ помог пользователю.
* Завершение диалога с тех. поддержкой.

Таким образом, данная диаграмма состояний отражает ключевые этапы процесса обращения пользователя в техническую поддержку и взаимодействия с ней.

# Теоретические вопросы подготовки к лабораторной работе

1. Укажите виды диаграмм поведения:

Всего существует 3 вида диаграмм поведения, описанных в UML:

* диаграмма деятельности;
* диаграмма состояний;
* диаграмма вариантов использования.

1. 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.

1. Опишите основные нотации, которые используются на диаграмме состояний:

Основные нотации диаграммы состояний:

* Состояние (State): представляет конкретный момент в жизненном цикле объекта или системы, когда она находится в определённом состоянии. На диаграмме изображается в виде прямоугольника с закруглёнными углами и именем состояния внутри. Например: Logged In — состояние пользователя в приложении после успешного входа.
* Начальное состояние (Initial State): обозначает начальную точку жизненного цикла объекта или процесса. Изображается в виде закрашенного чёрного круга. Используется для того, чтобы показать, с какого состояния начинается выполнение диаграммы.
* Конечное состояние (Final State): указывает завершение жизненного цикла или процесса. Обозначается как чёрный круг, обведённый окружностью. Когда система достигает конечного состояния, дальнейшие переходы невозможны.
* Переход (Transition): показывает, как система или объект переходит из состояния. Изображается в виде стрелки, направленной от одного состояния к другому. Обычно сопровождается меткой перехода, которая описывает условие или событие, вызывающее переход.
* Событие (Event): это действие или событие, которое инициирует переход между состояниями. Например, событие submit может вызывать переход пользователя из состояния Draft в состояние Submitted. Условие (Guard Condition). Логическое условие, проверяемое перед переходом. Если условие выполняется, происходит переход. Обозначается в квадратных скобках [] рядом с переходом.
* Действие (Action): операция, выполняемая в момент перехода между состояниями. Обычно указывается после события или условия и отделяется косой чертой. Например, Login / ShowDashboard — при переходе на действие Login выполняется операция ShowDashboard.
* Составное состояние (Composite State): состояние, содержащее подстадии и показывающее более сложные процессы. Представляется как прямоугольник с закруглёнными углами, внутри которого вложены другие состояния и переходы. Используется для отображения многоуровневых процессов.
* Разветвление (Fork/Join): разделение потока на несколько параллельных переходов (Fork) или синхронизация параллельных переходов в один (Join). Intersect обозначается как пунктирная стрелка, из которой выходит несколько стрелок.

1. Укажите виды связей между объектами на диаграмме последовательностей:

Синхронное сообщение — отправитель передает управление получателю, и 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.

Ответное сообщение — данное сообщение является ответом на синхронное сообщение. Графически изображается пунктирной линией с открытой стрелкой.

Асинхронное сообщение — почти тоже самое, что и синхронное, только отличие от синхронного сообщения состоит в том, что отправитель не теряет возможности совершать другие действия. Графически изображается сплошной линией с открытой стрелкой.

1. Какая диаграмма позволяет моделировать параллельные вычисления?

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