**3 Проектирование при объектном подходе**

**3.1 Диаграмма прецедентов**

Диаграмма вариантов использования в UML (Unified Modeling Language) является важным инструментом для анализа и описания функциональности системы на концептуальном уровне. Данная диаграмма представляет собой графическое изображение отношений между акторами (пользователями или внешними системами) и прецедентами (функциональными элементами системы), позволяя визуализировать различные сценарии использования системы.

Диаграмма вариантов использования помогает описать различные сценарии использования системы, иллюстрируя взаимодействие между акторами и системой. Путем визуализации прецедентов и взаимодействия между ними она помогает лучше понять поведение системы в контексте конкретных сценариев использования. Также данная диаграмма упрощает коммуникацию между заказчиком, конечными пользователями и разработчиками, позволяя обсуждать и уточнять требования к системе.

Одним из ключевых аспектов диаграммы вариантов использования является идентификация прецедентов — основных функциональных элементов системы, взаимосвязей и определение границ системы. Этот инструмент помогает определить, какие акторы взаимодействуют с системой и какие функции предоставляет данная система для этих акторов.

Диаграмма использования имеет 4 составляющие:

актер – любая сущность, выполняющая роль в системе;

случай использования – функция или действие внутри системы;

система – используется для определения сферы применения;

пакет – используются для группировки случаев использования.

Так же существует 4 вида на диаграмме использований:

ассоциация – один объект связан с другим объектом класса;

обобщения – один объект наследует другой объект класса;

расширение – функциональность одного прецедента может быть расширена функциональностью другого прецедента;

включение – один прецедент использует функциональность другого прецедента. [5]

Диаграмма прецедентов для веб-приложения «TaskMaster» представлена на рисунке 3.1.

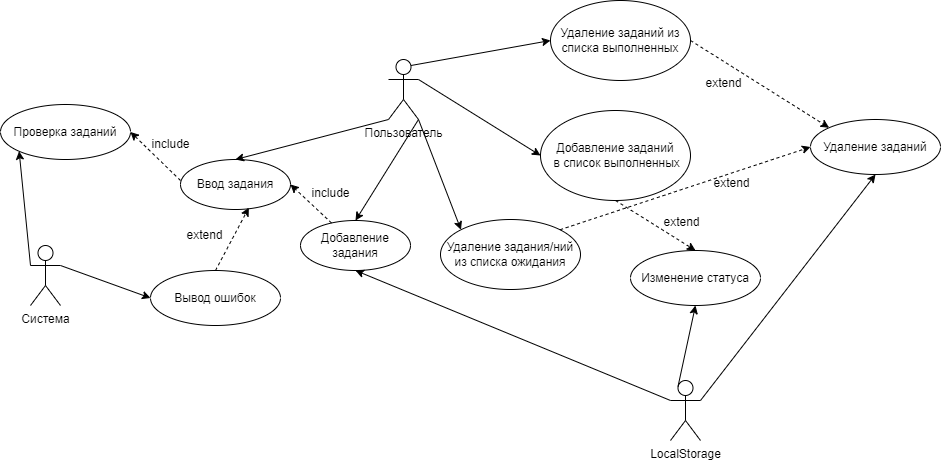


Рисунок 3.1 – Диаграмма прецедентов

Источник: собственная разработка

Описание действующих лиц представлено в таблице 3.1.

Таблица 3.1 – Описание действующих лиц

|  |  |
| --- | --- |
| Действующее лицо | Вариант использования |
| Пользователь | ввод задания;  добавление задания;  удаление задания/ний из списка ожидания;  добавление заданий в список выполненных;  удаление заданий из списка выполненных. |
| Система | проверка задания;  вывод ошибок. |
| LocalStorage | добавление задания;  изменение статуса;  удаление заданий. |

Источник: собственная разработка

Таблица 3.2 – Описание варианта использования «Ввод задания».

|  |  |
| --- | --- |
| Имя варианта использования | Ввод задания |
| Краткое описание варианта использования | Данный вариант использования описывает ввод задания в строку ввода |
| Действующие лица, взаимодействующие с вариантом использования | Пользователь |
| Предусловия для выполнения варианта использования | Отсутствуют |

Продолжение таблицы 3.2.

|  |  |
| --- | --- |
| Основной поток действий при исполнении варианта использования | Данный вариант использования начинает выполнение, когда пользователь       начинает взаимодействовать со строкой ввода. |
| Альтернативный поток действий при исполнении варианта использования | Отсутствует |
| Исключения при выполнении варианта использования | Отсутствуют |
| Постусловия после выполнения варианта использования | Если вариант использования выполнен верно, система подтверждает отсутствие ошибок, если нет, то будет показано уведомление об ошибке. |
| Связь с другими вариантами использования | Проверка заданий, вывод ошибок |

Источник: собственная разработка

Таблица 3.3 – Описание варианта использования «Добавление задания»

|  |  |
| --- | --- |
| Имя варианта использования | Добавление задания |
| Краткое описание варианта использования | Данный вариант использования описывает добавление задания в список ожидания. |
| Действующие лица, взаимодействующие с вариантом использования | Пользователь, LocalStorage |
| Предусловия для выполнения варианта использования | Выполнение варианта использования «Ввод задания» . |
| Основной поток действий при исполнении варианта использования | Вариант использования начинает выполнение, когда система проверила введенное задание. Пользователь нажимает на кнопку «ADD», после чего задание добавляется в список ожидания. |
| Альтернативный поток действий при исполнении варианта использования | Отсутствует |
| Исключения при выполнении варианта использования | Отсутствует |
| Постусловия после выполнения варианта использования | Если вариант использования выполнен верно, задание появится в списке ожидания |

Продолжение таблицы 3.3.

|  |  |
| --- | --- |
| Связь с другими вариантами использования | Ввод задания |

Источник: собственная разработка

Таблица 3.4 – Описание варианта использования «Удаление задания из списка ожидания»

|  |  |
| --- | --- |
| Имя варианта использования | Удаление задания из списка ожидания |
| Краткое описание варианта использования | Данный вариант использования описывает удаление задания из списка ожидания. |
| Действующие лица, взаимодействующие с вариантом использования | Пользователь |
| Предусловия для выполнения варианта использования | Задание должно быть в списке ожидания |
| Основной поток действий при исполнении варианта использования | Вариант использования начинает выполнение, когда пользователь нажал на кнопку удаления задания. Пользователь нажимает на кнопку «Х», после чего задание удаляется из списка ожидания. |
| Альтернативный поток действий при исполнении варианта использования | Отсутствует |
| Исключения при выполнении варианта использования | Отсутствует |
| Постусловия после выполнения варианта использования | Если вариант использования выполнен верно, задание удалится из списка ожидания. |
| Связь с другими вариантами использования | Удаление заданий |

Источник: собственная разработка

Таблица 3.5 – Описание варианта использования «Добавление задания в список выполненных»

|  |  |
| --- | --- |
| Имя варианта использования | Добавление задания в список выполненных |
| Краткое описание варианта использования | Данный вариант использования описывает добавление задания в список выполненных. |

Продолжение таблицы 3.5.

|  |  |
| --- | --- |
| Действующие лица, взаимодействующие с вариантом использования | Пользователь |
| Предусловия для выполнения варианта использования | Задание должно быть в списке ожидания |
| Основной поток действий при исполнении варианта использования | Вариант использования начинает выполнение, когда пользователь нажал на кнопку добавление задания. Пользователь нажимает на кнопку «✓», после чего задание добавляется в список выполненных. |
| Альтернативный поток действий при исполнении варианта использования | Отсутствует |
| Исключения при выполнении варианта использования | Отсутствует |
| Постусловия после выполнения варианта использования | Если вариант использования выполнен верно, задание добавится в список выполненных. |
| Связь с другими вариантами использования | Изменение статуса |

Источник: собственная разработка

Таблица 3.6 – Описание варианта использования «Удаление задания из списка выполненных»

|  |  |
| --- | --- |
| Имя варианта использования | Удаление задания из списка выполненных |
| Краткое описание варианта использования | Данный вариант использования описывает удаление задания из списка выполненных. |
| Действующие лица, взаимодействующие с вариантом использования | Пользователь |
| Предусловия для выполнения варианта использования | Задание должно быть в списке выполненных. |
| Основной поток действий при исполнении варианта использования | Вариант использования начинает выполнение, когда пользователь нажал на кнопку удаления задания на вкладке выполненных заданий. Пользователь нажимает на кнопку «Х», после чего задание удаляется. |
| Альтернативный поток действий при исполнении варианта использования | Отсутствует |

Продолжение таблицы 3.6.

|  |  |
| --- | --- |
| Исключения при выполнении варианта использования | Отсутствует |
| Постусловия после выполнения варианта использования | Если вариант использования выполнен верно, задание удалится из списка выполненных. |
| Связь с другими вариантами использования | Удаление заданий |

Источник: собственная разработка

Таблица 3.7 – Описание варианта использования «Проверка заданий»

|  |  |
| --- | --- |
| Имя варианта использования | Проверка заданий |
| Краткое описание варианта использования | Данный вариант использования описывает проверку задания перед его добавлением. |
| Действующие лица, взаимодействующие с вариантом использования | Система |
| Предусловия для выполнения варианта использования | Ввод данных |
| Основной поток действий при исполнении варианта использования | Вариант использования начинает выполнение, когда пользователь начинает взаимодействовать со строкой ввода. |
| Альтернативный поток действий при исполнении варианта использования | Отсутствует |
| Исключения при выполнении варианта использования | Отсутствует |
| Постусловия после выполнения варианта использования | Если вариант использования выполнен верно, задание будет обработано системой. |
| Связь с другими вариантами использования | Ввод данных |

Источник: собственная разработка

Таблица 3.8 – Описание варианта использования «Вывод ошибок»

|  |  |
| --- | --- |
| Имя варианта использования | Вывод ошибок |
| Краткое описание варианта использования | Данный вариант использования описывает вывод ошибок о некорректно введенном значении. |

Продолжение таблицы 3.8.

|  |  |
| --- | --- |
| Действующие лица, взаимодействующие с вариантом использования | Система |
| Предусловия для выполнения варианта использования | Ввод данных |
| Основной поток действий при исполнении варианта использования | Вариант использования начинает выполнение, когда пользователь начинает взаимодействовать со строкой ввода. |
| Альтернативный поток действий при исполнении варианта использования | Отсутствует |
| Исключения при выполнении варианта использования | Отсутствует |
| Постусловия после выполнения варианта использования | Если вариант использования выполнен верно, система выведет ошибку. |
| Связь с другими вариантами использования | Ввод данных |

Источник: собственная разработка

Таблица 3.9 – Описание варианта использования «Изменение статуса»

|  |  |
| --- | --- |
| Имя варианта использования | Изменение статуса |
| Краткое описание варианта использования | Данный вариант использования описывает изменение статуса задания. |
| Действующие лица, взаимодействующие с вариантом использования | LocalStorage |
| Предусловия для выполнения варианта использования | Добавление задания в список ожидания/выполненных |
| Основной поток действий при исполнении варианта использования | Вариант использования начинает выполнение, когда пользователь добавляет новое задание. Заданию присваивается статус «ожидание». При отметке задания выполненным, статус задания становится «выполнено». |
| Альтернативный поток действий при исполнении варианта использования | Отсутствует |
| Исключения при выполнении варианта использования | Отсутствует |

Продолжение таблицы 3.9.

|  |  |
| --- | --- |
| Постусловия после выполнения варианта использования | Если вариант использования выполнен верно, задание будет находится в LocalStorage с предлагаемыми статусами. |
| Связь с другими вариантами использования | Добавление задания в список ожидания, добавление задания в список выполненных. |

Источник: собственная разработка

Таблица 3.10 – Описание варианта использования «Удаление заданий»

|  |  |
| --- | --- |
| Имя варианта использования | Удаление заданий |
| Краткое описание варианта использования | Данный вариант использования описывает удаление заданий из LocalStorage. |
| Действующие лица, взаимодействующие с вариантом использования | LocalStorage |
| Предусловия для выполнения варианта использования | Отсутствуют |
| Основной поток действий при исполнении варианта использования | Вариант использования начинает выполнение, когда пользователь взаимодействует с кнопками удаления на странице. По нажатию кнопок задание удаляется из LocalStorage. |
| Альтернативный поток действий при исполнении варианта использования | Отсутствует |
| Исключения при выполнении варианта использования | Отсутствует |
| Постусловия после выполнения варианта использования | Если вариант использования выполнен верно, задание будет удалено из LocalStorage. |
| Связь с другими вариантами использования | Удаление задания из списка ожидания, удаление задания из списка выполненных. |

Источник: собственная разработка

Как итог, диаграмма прецедентов является эффективным средством для общения между участниками проекта (заказчиками, разработчиками, дизайнерами) и позволяет лучше понять функциональные требования к системе. Диаграмма прецедентов помогает определить границы системы, идентифицировать основные функциональные возможности и обеспечивает основу для дальнейшего анализа и проектирования системы.

**3.2 Диаграмма классов**

Диаграмма классов в языке моделирования UML является одной из ключевых структурных диаграмм, которая представляет общую структуру иерархии классов в системе, атрибуты, методы, интерфейсы и взаимосвязи между классами. Классы представляют абстракцию для объектов, которые имеют общие свойства, поведение и отношения, обычно представлены в виде прямоугольников с именем класса.

Основными элементами являются классы и связи между ними. Классы характеризуются при помощи атрибутов и операций. Атрибуты описывают свойства объектов класса. Методы (операции) представляют функциональность объектов класса, описывая действия, которые могут быть выполнены над объектами класса, с возможностью иметь параметры и возвращаемые значения.

Существует 6 типов связей:

ассоциация - это структурная связь между элементами модели, которая описывает набор связей, существующих между объектами;

агрегация – особая разновидность ассоциации, представляющая структурную связь целого с его частями;

композиция — более строгий вариант агрегации. Композиция имеет жёсткую зависимость времени существования экземпляров класса контейнера и экземпляров содержащихся классов;

обобщение – выражает специализацию или наследование, в котором специализированный элемент (потомок) строится по спецификациям обобщенного элемента (родителя);

зависимость – семантически представляет собой связь между двумя элементами модели, в которой изменение одного элемента (независимого) может привести к изменению семантики другого элемента (зависимого);

реализация – это семантическая связь между классами, когда один из них (поставщик) определяет соглашение, которого второй (клиент) обязан придерживаться. [1]

Диаграмма классов для веб-приложения «TaskMaster» представлена на рисунке 3.2.

На диаграмме классов представлено и описано 3 класса: «Задание», «Список заданий» и «TaskMaster».

Класс «Задание» и «Список заданий» имеют связь агрегацию. «Задание» является частью «Список заданий», но при этом задание может существовать отдельно.

Класс «LocalStorage» имеет связь зависимость с классами «Задание» и «Список заданий». Изменение классов «Задание» и «Список заданий» влияет на изменение класса «LocalStorage».

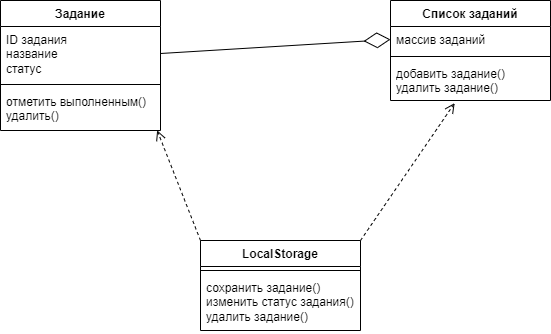


Рисунок 3.2 – Диаграмма классов

Источник: собственная разработка

Описание атрибутов классов представлено в таблице 3.11.

Таблица 3.11 – Описание атрибутов

|  |  |  |  |
| --- | --- | --- | --- |
| Название класса | Имя атрибута | Тип данных | Описание атрибута |
| 1 | 2 | 3 | 4 |
| Задание | ID задания | Integer | Номер задания. |
| Название | String | Текст задания. |
| Статус | String | Статус задания. |
| Список заданий | Массив заданий | Array | Содержит в себе задания и статусы. |
| LocalStorage | - | - | - |

Источник: собственная разработка

Описание операций классов представлено в таблице 3.12.

Таблица 3.12 – Описание операций классов

|  |  |  |
| --- | --- | --- |
| Название класса | Название операции | Описание операции |
| 1 | 2 | 3 |
| Задание | Отметить выполненным | Добавление задания в список выполненных заданий. |
| Удалить | Удаление задания из списка выполненных. |

Продолжение таблицы 3.12.

|  |  |  |
| --- | --- | --- |
| Список заданий | Добавить задание | Добавление задания в список ожидания. |
| Удалить задание | Удаление задания из списка ожидания. |
| LocalStorage | Сохранить задание | Добавление задания в хранилище LocalStorage. |
|  | Изменить статус задания | Изменение статуса задания в LocalStorage. |
|  | Удалить задание | Удаление задания из localStorage. |

Источник: собственная разработка

В целом, диаграмма классов играет важную роль в проектировании программного обеспечения, предоставляя ясное представление о структуре системы и облегчая процесс разработки и сопровождения.

**3.3 Описание программного средства**

**3.3.1 Диаграмма последовательности и кооперации**

Диаграмма последовательности в UML - важный инструмент для визуализации взаимодействия между объектами в системе, позволяя понять порядок передачи сообщений между ними во времени и взаимодействие в рамках конкретного сценария использования.

Фрагменты и блоки могут быть использованы для показа различных путей выполнения в системе, такие как опциональные фрагменты, циклы и параллельные конструкции. Ветвления и объединения на диаграмме показывают разветвления в логике взаимодействия и точки слияния различных потоков выполнения. Операции и вызовы методов могут быть включены, чтобы показать, как объекты взаимодействуют для выполнения определенных действий. Ограничения и стереотипы могут использоваться для дополнительной аннотации диаграммы и уточнения смысла. Синхронные сообщения показывают ожидание ответа, а асинхронные - продолжение выполнения без ожидания ответа. Фокусы используются для выделения аспектов взаимодействия, а петли и повторения могут быть показаны с помощью фрагментов и циклов для отображения повторяющихся операций.

Диаграммы последовательности являются мощным инструментом для анализа и проектирования систем, помогая лучше понять взаимодействие между объектами и последовательность действий в рамках сценария использования. [1]

Диаграмма последовательности для веб-приложения «TaskMaster» представлена на рисунке 3.3.

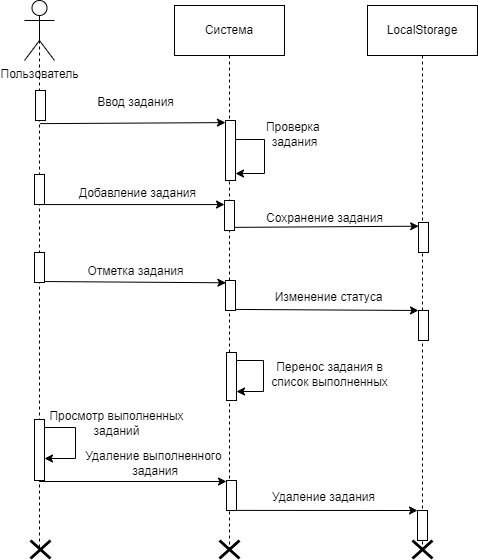


Рисунок 3.3 – Диаграмма последовательности

Источник: собственная разработка

Основными объектами на данной диаграмме последовательности выступают: Пользователь, Система, LocalStorage. Все объекты являются активными, так как взаимодействуют друг с другом с помощью сообщений. Каждый объект имеет линию жизни. Конец линии жизни обозначен «Х» в конце.

Диаграмма кооперации предоставляет описание поведения системы на уровне индивидуальных объектов, которые взаимодействуют друг с другом путем обмена сообщениями с целью достижения определенной цели или реализации конкретного сценария использования. Этот тип диаграммы представляет структуру модели как совокупность взаимодействующих объектов и обеспечивает наглядное представление взаимодействия объектов.

На диаграмме кооперации объекты располагаются как экземпляры классов, они связаны друг с другом через ассоциации и обмениваются

сообщениями. Связи дополняются стрелками сообщений, отображая только те объекты, которые активно участвуют в моделируемой кооперации. Как и на диаграмме классов, также показываются структурные отношения между объектами с использованием различных типов линий. Связи могут быть дополнены ролями, которые объекты играют в данном контексте взаимодействия.

Далее на диаграмме изображаются динамические взаимосвязи в виде потоков сообщений с указанием направления стрелок рядом с соединительными линиями между объектами. Каждое сообщение имеет имя и порядковый номер в общей последовательности сообщений, что помогает понять порядок взаимодействия объектов в рамках данного сценария. Диаграммы кооперации обеспечивают понимание взаимодействия объектов и динамики сообщений в системе, что помогает при анализе и проектировании сложных систем. [5]

Диаграмма кооперации для веб-приложения «TaskMaster» представлена на рисунке 3.4.

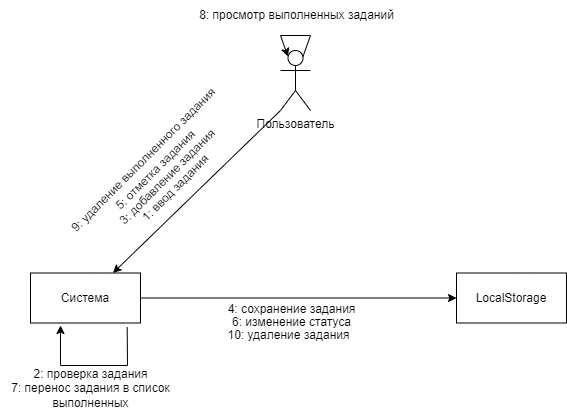


Рисунок 3.4 – Диаграмма кооперации

Источник: собственная разработка

На данной диаграмме кооперации есть три активных объекта: Пользователь, Система, LocalStorage. Отношения между объектами описаны с помощью связей.

Как итог, диаграммы последовательности и кооперации дополняют друг друга, обеспечивая полное представление как статических, так и динамических аспектов взаимодействия объектов в системе.

**3.3.2 Диаграмма деятельности**

Диаграмма деятельности в объектно-ориентированном проектировании является мощным инструментом, который позволяет подробно разрабатывать алгоритмическую и логическую структуру системы. Модель помогает дизайнерам и разработчикам увидеть последовательность шагов, необходимых для достижения определенной цели или выполнения конкретной операции в системе.

Каждое действие на диаграмме деятельности разбивается на фундаментальные процессы, что способствует более детальному анализу и пониманию функционирования системы. Этот тип диаграммы широко применяется для моделирования процессов работы системы после того, как был запущен определенный сценарий или прецедент.

При создании диаграммы деятельности используются различные элементы, такие как начальный узел, конечный узел, конечный узел потока, узлы управления, состояния действий, сигналы, объекты, центральный буфер и дорожки.

Использование диаграмм деятельности способствует более точному пониманию логики работы системы, улучшает коммуникацию между участниками проекта и помогает выявить потенциальные проблемы или улучшения в процессах выполнения системы. [5]

Диаграмма деятельности для веб-приложения «TaskMaster» представлена на рисунке 3.5.

На данной диаграмме деятельности присутствует три дорожки: Пользователь, Система, LocalStorage. Присутствует начальный узел, конечный узел и узлы управления. Установлена связь между состояниями действий с помощью сигналов.

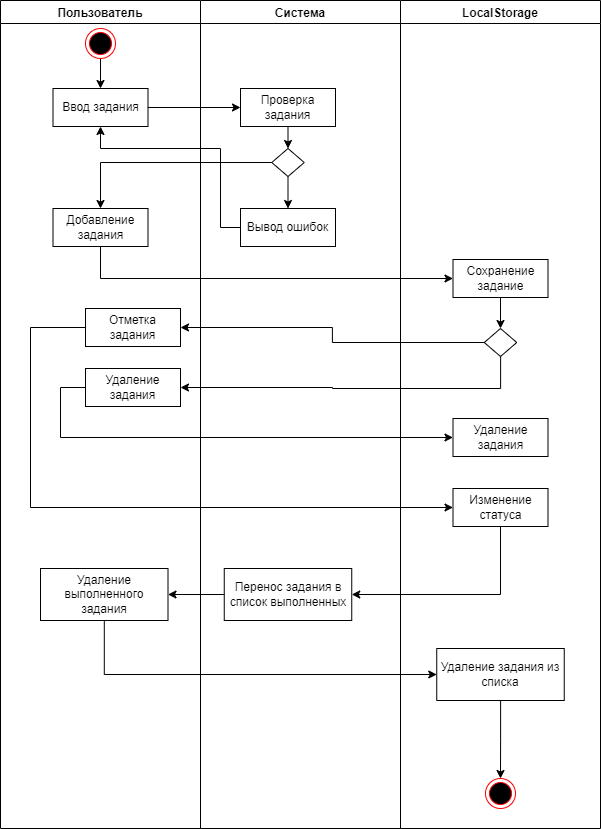


Рисунок 3.5 – Диаграмма деятельности

Источник: собственная разработка

Как результат, диаграмма деятельности в объектно-ориентированном проектировании представляет мощный инструмент для детального анализа алгоритмической и логической организации системы.

**3.3.3 Диаграмма состояний**

Диаграмма состояний является важным инструментом моделирования в объектно-ориентированном проектировании, который позволяет описать все возможные состояния, в которых может находиться объект, а также процессы перехода между этими состояниями под воздействием внешних событий или действий.

Основными элементами диаграммы состояний являются «Состояние» и «Переход». «Состояние» представляет конкретное состояние объекта в определенный момент времени, а «Переход» обозначает изменение состояния объекта в ответ на определенное событие или условие.

Диаграмма состояний имеет схожую семантику с диаграммой деятельности, но вместо действий здесь используются состояния и переходы. По сравнению с диаграммой деятельности, где различие между «Деятельностью» и «Действием» заключается в возможности дальнейшей декомпозиции, на диаграмме состояний «Состояние» символизирует стабильное состояние, в котором объект находится продолжительное время, в то время как «Действие» представляет моментальные изменения.

Использование диаграмм состояний позволяет разработчикам и аналитикам визуализировать и анализировать поведение объектов в системе, переходы между различными состояниями, что способствует более глубокому пониманию функционирования системы и помогает выявить потенциальные проблемы или улучшения в управлении состояниями объектов. [5]

Диаграмма состояний для объекта «Пользователь» для веб-приложения «TaskMaster» представлена на рисунке 3.6.



Рисунок 3.6 – Диаграмма состояний для объекта «Пользователь»

Источник: собственная разработка

На данной диаграмме присутствует начальный и конечный узел, состояния и переходы между ними.

Как результат, диаграмма состояний является мощным инструментом для визуализации всех возможных состояний, в которых может находиться объект.

**3.4 Проект интерфейса ПО**

**3.4.1 Граф переходов состояний интерфейса**

Граф переходов состояний интерфейса в разработке программного обеспечения представляет собой визуальное представление различных состояний, в которых может находиться интерфейс приложения, а также переходов между этими состояниями в ответ на определенные события или действия пользователя.

Этот тип диаграммы позволяет разработчикам и дизайнерам программного обеспечения лучше понять логику поведения интерфейса и взаимодействия пользователя с приложением. Граф переходов состояний интерфейса помогает в описании различных сценариев использования приложения, определении возможных путей навигации пользователя и выявлении потенциальных проблемных ситуаций. [8]

**3.4.2 Проектирование экранных форм ввода вывода информации**

Для разрабатываемого веб-приложения «TaskMaster» был реализован прототип в Figma. [8]

На рисунке 3.7 представлен прототип главной страницы веб-приложения «TaskMaster». Главная страница включает в себя блок для добавления и отображения заданий, кнопку очистки и боковую панель для просмотра выполненных заданий.

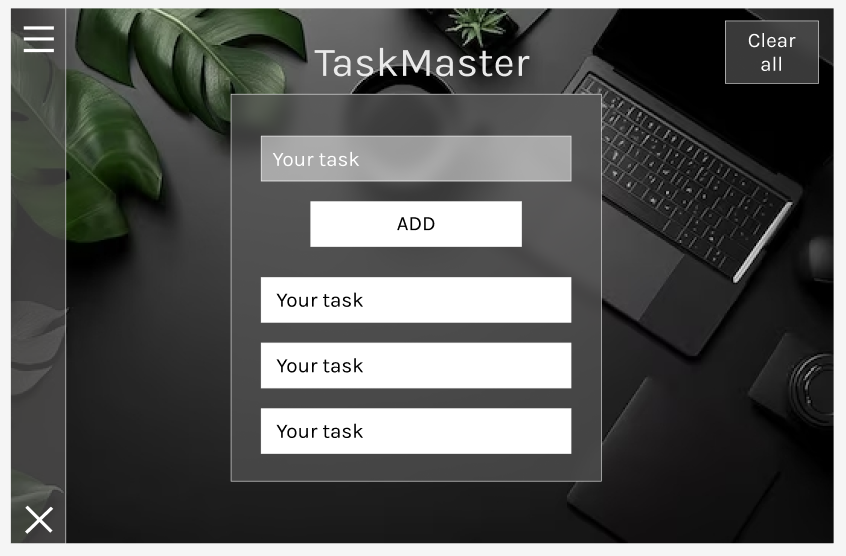


Рисунок 3.7 – Прототип главного меню

Источник: собственная разработка

На рисунке 3.8 представлен прототип блоков с заданиями, которые будут анимированы. При наведении мыши на блок будут появляться кнопки удаления и отметки задания.

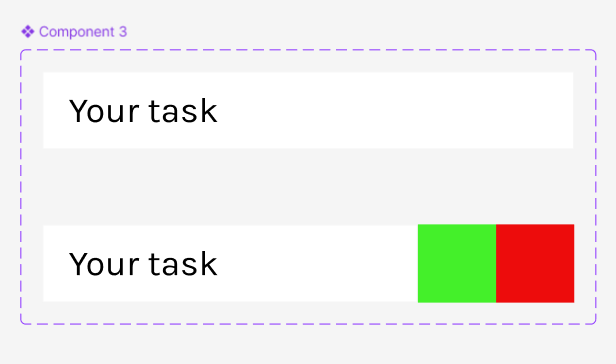


Рисунок 3.8 – Прототип блоков с заданиями

Источник: собственная разработка

На рисунке 3.9 представлен прототип блока выполненных заданий. При наведении мыши на боковую панель будут отображены выполненные задания. По кнопкам «Х» задания будут удаляться.

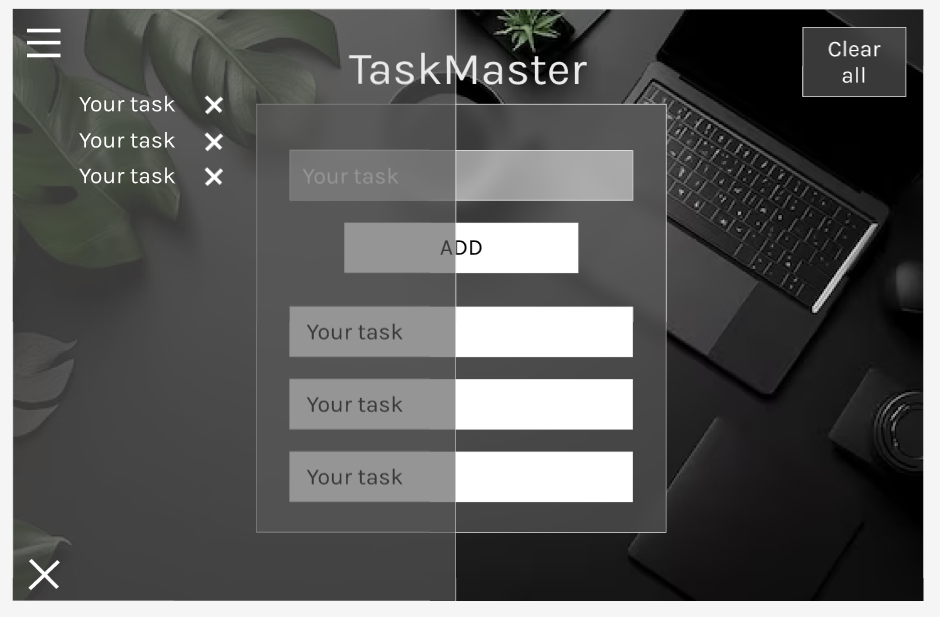


Рисунок 3.9 – Прототип блока выполненных заданий

Источник: собственная разработка

Как итог, были реализованы модели при объектном подходе и прототип разрабатываемого веб-приложения.