Федеральное государственное образовательное бюджетное
учреждение высшего образования
«ФИНАНСОВЫЙ УНИВЕРСИТЕТ ПРИ ПРАВИТЕЛЬСТВЕ РОССИЙСКОЙ ФЕДЕРАЦИИ»
(ФИНАНСОВЫЙ УНИВЕРСИТЕТ)
Колледж информатики и программирования

ОТЧЁТ

по практическим работам

Дисциплина МДК 05.02 «Разработка кода информационных систем»

Выполнил: Бондарев Н.Н.
Группа: 3ИСИП-722
Преподаватель: Абзалимов Р.Р.

Москва
2025

Практическая работа №1 на тему

«Построение диаграммы Вариантов использования и диаграммы Последовательности и генерация кода»

Цель работы:

Изучение объектно-ориентированного анализа и моделирования бизнес-процессов в исследуемой предметной области с помощью языка UML.

Задание:

В ходе выполнения задания для выбранной (или заданной) предметной области изучается процесс построения диаграммы вариантов использования (Use Case Diagram) и диаграммы деятельности (Activity Diagram) с использованием CASE-средства, поддерживающего язык UML. Также будет рассмотрено создание диаграммы последовательности (Sequence Diagram) для более детального отображения взаимодействия объектов.

Диаграммы UML:

В ходе работы были разработаны различные диаграммы в программе Draw.io. Одной из них является диаграмма вариантов использования интернет-магазина с товарами для украшения интерьера, которая представлена на рисунке 1.

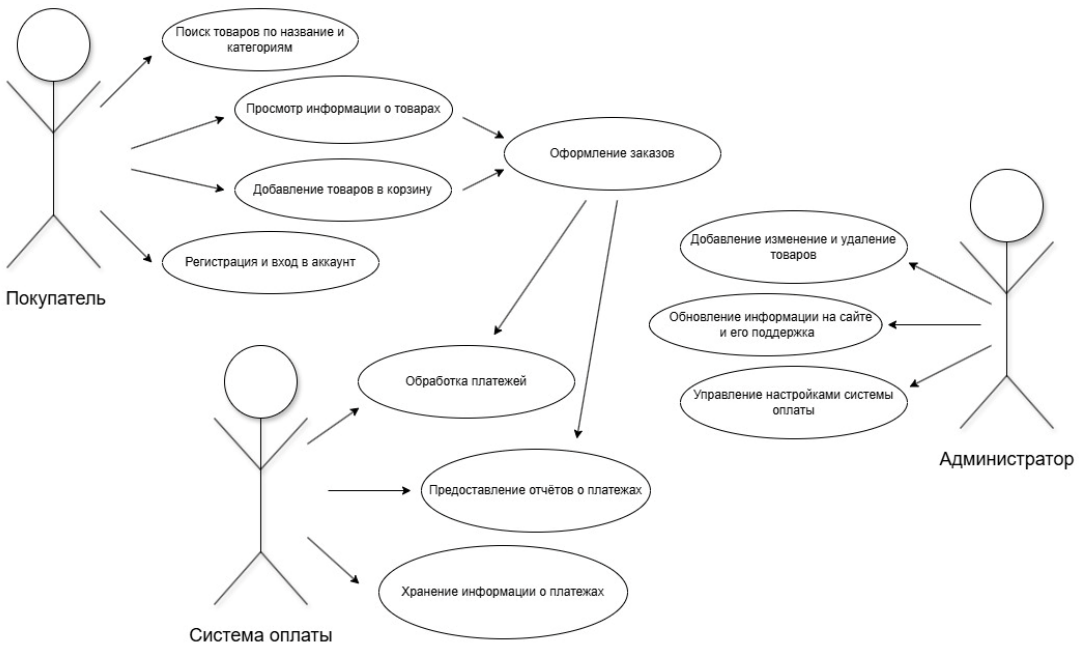

Рисунок 1. Диаграмма вариантов использования интернет-магазина «Украшение Интерьера»

Также были разработаны диаграммы деятельности оформления заказа AliExpress, Wildberries, Ozon и диаграмма будущего сайта, которые представлены на рисунке 2-3.

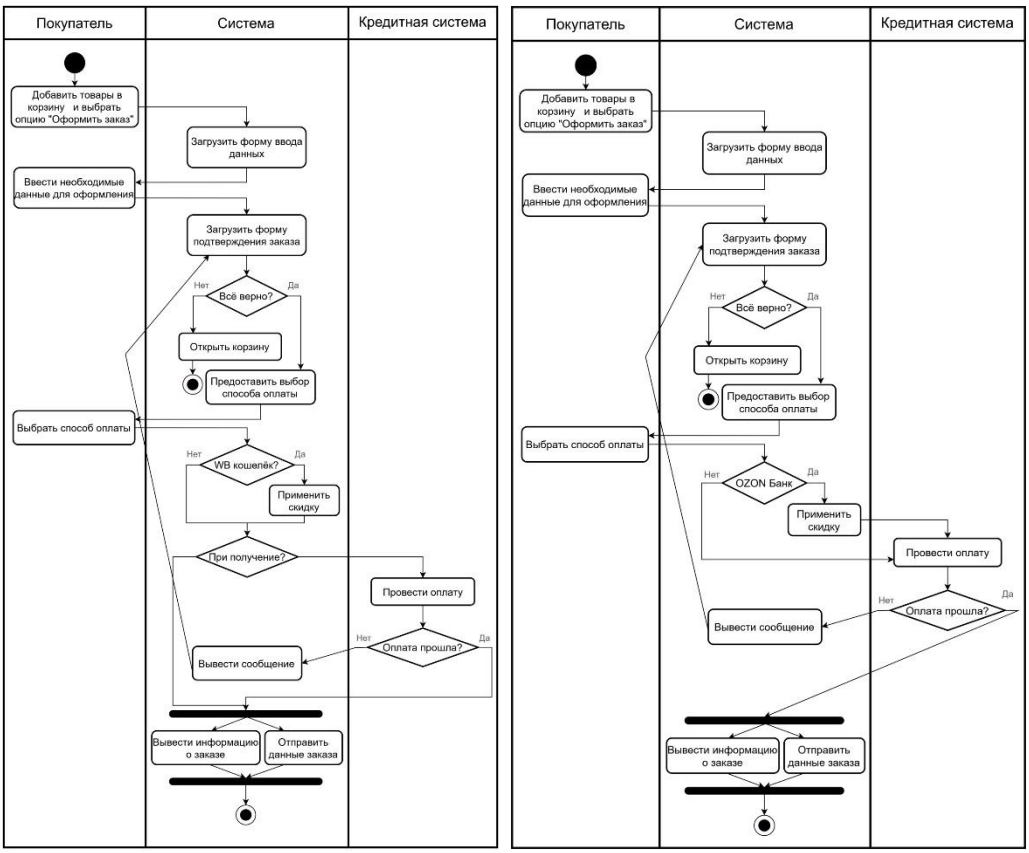

Рисунок 2. Диаграммы деятельности оформления заказа Wildberries и Ozon

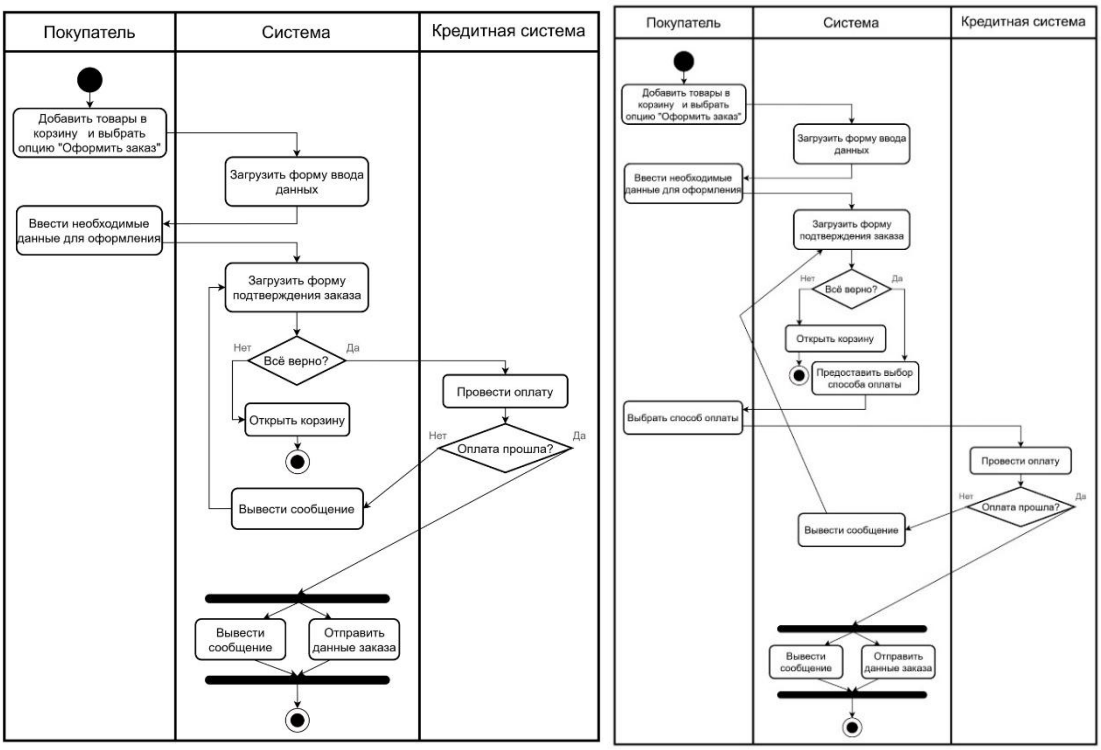

Рисунок 3. Диаграммы деятельности оформления заказа будущего сайта и AliExpress

UI прототип корзины сайта по продаже товаров украшений интерьера:

В ходе работы был разработаны прототип корзины интернет-магазина на тему «Украшение интерьера» в графическом редакторе Figma. Итоговый результат представлен на рисунке 4.

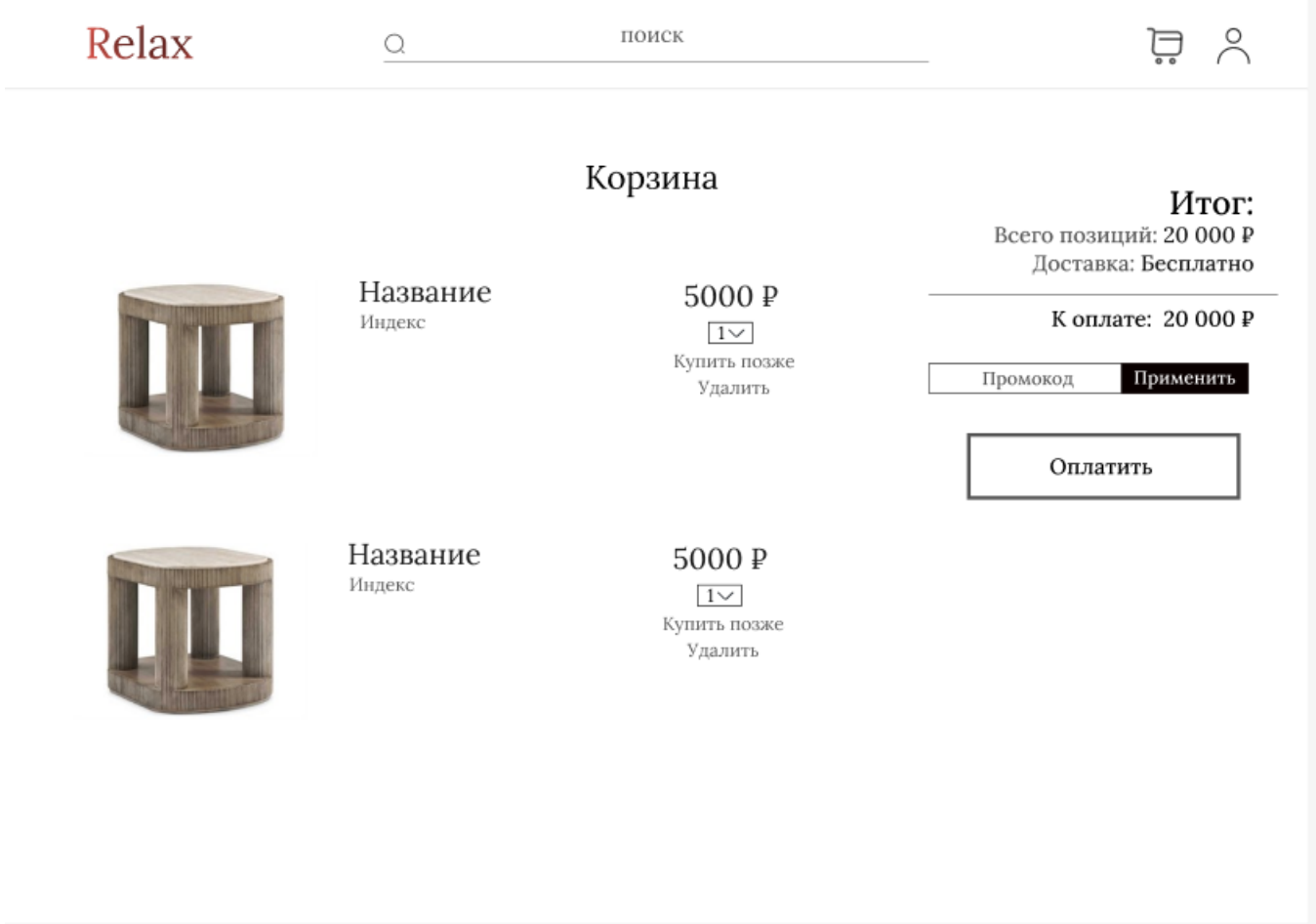

Рисунок 4. UI прототип корзины интернет-магазина

Ответы на вопросы:

1) Что такое UML и каковы его основные компоненты?

UML (Unified Modeling Language) — это универсальный язык моделирования, предназначенный для визуального представления, спецификации, разработки и документирования программных систем. Он помогает упрощать процесс проектирования за счет использования стандартного подхода в различных аспектах

анализа и разработки. UML применим как в объектно-ориентированном программировании, так и в бизнес-моделировании.

Ключевые элементы UML:

- Диаграммы — основной инструмент для визуализации структуры и поведения системы.
- Структурные диаграммы описывают статические аспекты, например, классы и их взаимосвязи.
- Поведенческие диаграммы показывают динамические аспекты, такие как взаимодействие объектов и изменения состояний.
- Типы диаграмм:
- Диаграмма классов — отображает классы, их свойства, методы и связи.
- Диаграмма последовательностей — демонстрирует обмен сообщениями между объектами.
- Диаграмма вариантов использования — иллюстрирует взаимодействие пользователей с системой.
- Диаграмма состояний — описывает изменения состояний объекта.
- Диаграмма компонентов — представляет физическую архитектуру системы.
- Диаграмма развертывания — показывает оборудование, на котором работает система.

2) Основные этапы объектно-ориентированного анализа и моделирования бизнес-процессов

Объектно-ориентированный подход к анализу и моделированию бизнес-процессов включает несколько ключевых шагов:

- Сбор требований: Определение нужд пользователей и ограничений проекта через общение с заинтересованными сторонами.
- Анализ требований: Выявление функций системы, пользовательских сценариев и вариантов использования. Создание диаграмм вариантов использования помогает визуализировать запросы.

- Объектный анализ: Идентификация объектов, их атрибутов, методов и связей, используя диаграммы классов.
- Моделирование бизнес-процессов: Разработка моделей текущих и целевых процессов, включая диаграммы активностей и последовательностей.
- Проектирование системы: Создание архитектуры, детализированных диаграмм классов, компонентов и развертывания.
- Реализация: Программирование системы в соответствии с разработанными моделями.
- Тестирование и валидация: Проверка функциональности системы с помощью различных видов тестов.
- Поддержка и развитие: Внедрение изменений на основе обратной связи от пользователей.

3) Диаграмма вариантов использования

Диаграмма вариантов использования отображает, как пользователи (актёры) взаимодействуют с системой для выполнения определённых функций.

Элементы диаграммы:

- Акторы — пользователи или внешние системы, взаимодействующие с системой.
- Варианты использования — действия, выполняемые системой (например, авторизация, создание отчётов).
- Связи:
  - Ассоциации между актёрами и вариантами использования.
  - Обобщение, включение и расширение для описания дополнительных функций.
- Роль диаграммы:
- Помогает уточнить функциональные требования.
- Облегчает понимание системы и взаимодействий.
- Служит основой для тестирования и разработки.
- Упрощает коммуникацию между участниками проекта.

4) Диаграмма деятельности

Диаграмма деятельности описывает последовательность действий или шагов процесса.

Основные элементы:

- Начальное состояние: начало процесса (закрашенный круг).
- Действия: конкретные шаги процесса (прямоугольники).
- Переходы: стрелки, указывающие поток управления.
- Условия: точки ветвления (ромбы).
- Конечное состояние: завершение процесса (круг с окружностью).
- Различия с диаграммой вариантов использования:
- Диаграмма деятельности фокусируется на внутренних процессах, а диаграмма вариантов использования — на взаимодействиях актёров с системой.
- Диаграмма деятельности описывает алгоритмы, а диаграмма вариантов использования — функциональность системы.

5) Диаграмма последовательности

Диаграмма последовательности показывает взаимодействия объектов в определённой временной последовательности.

Элементы:

- Объекты: участники процесса (прямоугольники).
- Линии жизни: время существования объектов (пунктирные вертикальные линии).
- Сообщения: взаимодействия между объектами:
    - Синхронные (сплошные стрелки).
    - Асинхронные (пунктирные стрелки).
    - Возвратные (пунктирные стрелки обратно).

6) Преимущества использования CASE-средств

CASE-средства обеспечивают:

- Продуктивность: автоматизация рутинных задач, снижение ошибок.
- Качество: стандарты разработки, поддержка тестирования.
- Планирование: визуализация архитектуры, управление требованиями.
- Документирование: автоматическая генерация документации.
- Совместную работу: упрощение коммуникации в команде.

7) Синхронизация в диаграммах деятельности

Синхронизация управляет параллельными потоками:

- Разделение: начало параллельных процессов (линия "fork").
- Объединение: завершение потоков в одну линию ("join").
- Применение: визуализация параллельных операций, например, обработки платежа и подготовки товара.

8) Оценка качества сгенерированного кода

- Качество кода: читаемость, соблюдение стандартов.
- Производительность: анализ времени выполнения и ресурсов.
- Покрытие тестами: модульные тесты, автоматическое тестирование.
- Соответствие требованиям: проверка кода на соответствие диаграммам.

9) Основные выводы и уроки, извлеченные из выполнения практической работы

В процессе выполнения данной практической работы были выявлены ключевые моменты, которые можно считать значимыми для понимания и улучшения подхода к проектированию и разработке:

Роль проектирования в успехе разработки. Эффективное использование UML-диаграмм и их правильное построение закладывают основу для качественной разработки. Тщательное проектирование диаграмм позволяет избежать логических ошибок и упрощает процесс реализации.

Тестирование как неотъемлемая часть работы. Тестирование необходимо на всех этапах — от проверки правильности диаграмм до анализа соответствия сгенерированного кода исходным требованиям.

Важность обратной связи. Совместная работа аналитиков, программистов и тестировщиков позволяет минимизировать количество ошибок, улучшить качество документации и повысить эффективность конечного продукта.

Практическое применение знаний. Выполнение работы способствовало укреплению навыков в проектировании UML и генерации кода. Это подчеркнуло значимость систематического подхода и автоматизации рутинных задач.

Автоматизация как инструмент повышения эффективности. Использование CASE-средств для генерации кода и создания документации сокращает временные затраты и снижает вероятность ошибок, связанных с человеческим фактором.

Применение в реальных проектах. Практические навыки, полученные в ходе работы, являются актуальными и полезными для реализации крупных проектов, требующих структурированного подхода к проектированию и разработке.

Вывод: в рамках работы был проведен анализ функциональности корзин интернет-магазинов, таких как AliExpress, Wildberries и Ozon. На основе проведенного анализа была спроектирована корзина для веб-сайта, предлагающего товары для умного дома. В основу разработки были включены лучшие практики, включая отображение скидок для наглядного представления экономии, отображение общего количества товаров в корзине, а также функции добавления и удаления позиций. Также реализована возможность использования купонов и промокодов для получения скидок. Модель системы оплаты, представленная в UML-диаграмме, была адаптирована с учетом подхода AliExpress, поскольку создание собственного банка, как это сделано у Wildberries и Ozon, нецелесообразно для интернет-магазина с узкой специализацией.

Практическая работа №2

на тему «Построение диаграммы Кооперации и диаграммы Развертывания и генерация кода»

Цель работы: ознакомиться с методологией моделирования информационных систем на основе языка UML.

Задание: в ходе выполнения задания для выбранной (или заданной) предметной области изучается процесс построения диаграммы Кооперации и диаграммы Развертывания с помощью CASE-средства, поддерживающего язык UML.

1. Основная цель моделирования в выбранной предметной области

Целью моделирования в заданной сфере является предоставление более четкого понимания структуры и функционирования системы через визуализацию и упрощение её сложных процессов. Моделирование используется для:

- анализа взаимодействий между компонентами,
- визуализации потоков данных,
- исследования ключевых связей, влияющих на проектирование. Это позволяет выявить проблемные зоны и оптимизировать систему.

2. Ключевые объекты и функции для исследования

Основные объекты системы:

- **Пользователь:** содержит идентификатор, имя, роль;
- **Система:** представляет интерфейс и бизнес-логику;
- **Товар:** включает название, цену, описание;
- **Заказ:** структура для управления покупками;
- **Оплата:** отвечает за процесс оплаты;
- **Отзывы:** связывает клиентов с продуктами.

Ключевые функции: регистрация, авторизация, поиск товаров, оформление заказа, оплата, написание отзывов.

3. Характеристики CASE-средства

Выбранное средство предоставляет:

- поддержку UML,

- генерацию кода и обратное проектирование,
- инструменты проверки синтаксиса моделей,
- возможности командной работы и документирования.

4. Интерфейс CASE-средства

Интерфейс интуитивен, содержит панели инструментов для построения UML-диаграмм, настройки атрибутов объектов и функций управления проектом.

5. Этапы создания проекта

Процесс начинается с инициализации проекта, настройки параметров, создания моделей и их сохранения. Включает:

1. выбор шаблона;
2. настройку модулей;
3. построение диаграмм;
4. сохранение и резервное копирование.

6. Диаграмма кооперации

Диаграмма показывает взаимодействия объектов через обмен сообщениями. Основные элементы:

- объекты,
- связи,
- сообщения,
- последовательность взаимодействий.

7. Построение диаграммы кооперации

Определяются:

- участники,
- сценарии использования,
- типы и порядок сообщений.

Диаграмма создается с помощью UML-инструментов, проверяется и уточняется.

8. Основные элементы диаграммы развертывания

Включают:

- узлы (серверы, устройства),

- компоненты (ПО, сервисы),
- артефакты (исполняемые файлы),
- связи (коммуникационные каналы).

Они визуализируют распределение системы по инфраструктуре.

9. **Как вы построили диаграмму развертывания для выбранной информационной системы?**

Этапы создания диаграммы развертывания:

1. **Определение целей и контекста:**
   На начальном этапе я уточнил, для чего будет использоваться диаграмма развертывания. Например, нужно было отразить физическую инфраструктуру, показать размещение компонентов системы или описать сетевые взаимодействия.
2. **Идентификация узлов:**
   Узлы представляют собой физические или виртуальные устройства, на которых размещаются компоненты системы. Для выбранной информационной системы я определил такие узлы, как:
   - **Сервер приложений**, на котором размещена бизнес-логика.
   - **Сервер базы данных**, хранящий данные системы.
   - **Клиентские устройства**, взаимодействующие с сервером (например, компьютеры пользователей или мобильные телефоны).
3. **Определение компонентов:**
   Я выделил программные модули, развернутые на каждом узле. Например:
   - На сервере приложений размещены компоненты API и бизнес-логики.
   - На сервере базы данных развернута СУБД с таблицами и хранимыми процедурами.
   - На клиентских устройствах используется веб-браузер или мобильное приложение.

4. **Связи между узлами и компонентами:**
   На диаграмме были обозначены соединения между узлами, которые отражают взаимодействие. Например:
   - Клиентское устройство соединяется с сервером приложений через HTTPS.
   - Сервер приложений обменивается данными с сервером базы данных по протоколу SQL через защищённое соединение.
5. **Использование инструментов CASE:**
   Для создания диаграммы я применил инструменты UML-моделирования, которые позволили добавить узлы, связи, компоненты и подписать их, чтобы отображение было понятным.
6. **Проверка и доработка:**
   После завершения диаграммы я проверил её на полноту и соответствие архитектуре системы. Обсудил диаграмму с командой, чтобы убедиться, что все аспекты взаимодействия корректно отражены.

10. **Каковы основные элементы диаграммы последовательностей и их значение?**
    Диаграмма последовательностей (sequence diagram) показывает последовательность сообщений, которыми обмениваются объекты в системе для выполнения определённого сценария. Основные элементы и их значение:

1. **Объекты (Actors и Objects):**
   - Представляют участников взаимодействия (пользователей, системы, компоненты).
   - Обозначаются прямоугольниками с названием объекта.
   - Пример: Пользователь, Система, База данных.
2. **Линии жизни (Lifelines):**
   - Вертикальные пунктирные линии, которые начинаются от объектов и отображают их существование во времени.

- На линии жизни указываются сообщения, которые объект отправляет или получает.

3. **Сообщения (Messages):**
   - Горизонтальные стрелки, которые соединяют линии жизни объектов.
   - Представляют вызовы методов, запросы или ответы.
   - Пример: Пользователь → Система: «Авторизация».
4. **Активности (Activation Bars):**
   - Прямоугольники на линии жизни, которые показывают период активности объекта при выполнении метода.
5. **Возврат сообщений (Return Messages):**
   - Обозначают результат выполнения операции или передачу данных.
   - Обычно изображаются пунктирной стрелкой от получателя к отправителю.
6. **Фреймы (Frames):**
   - Используются для группировки сообщений по определённому условию или циклу.
   - Пример: фрейм «Loop» показывает повторяющиеся действия.

Значение элементов:

- **Объекты** показывают участников сценария.
- **Сообщения** иллюстрируют взаимодействие между участниками.
- **Линии жизни** и **активности** помогают понять последовательность и длительность операций.
- **Возврат сообщений** указывает результат, что важно для анализа логики системы.

11. **Как вы построили диаграмму последовательностей для выбранной информационной системы?**

Для построения диаграммы последовательностей я выполнил следующие шаги:

1. **Определение сценария:**
   Сначала я выбрал конкретный сценарий использования системы. Например, сценарий «Авторизация пользователя» или «Оформление заказа».
2. **Определение участников:**
   Я идентифицировал всех участников, включая пользователей и компоненты системы. Например:
   - **Пользователь** — инициатор взаимодействия.
   - **Система** — выполняет бизнес-логику.
   - **База данных** — хранит и обрабатывает данные.
3. **Определение последовательности взаимодействий:**
   - Установил порядок действий между участниками. Например:
     - Пользователь вводит данные авторизации.
     - Система проверяет данные и передаёт запрос к базе данных.
     - База данных возвращает результат проверки.
4. **Построение диаграммы:**
   - Использовал CASE-средство для добавления объектов и линий жизни.
   - Нарисовал сообщения, соединяющие линии жизни участников.
   - Добавил активности на линии жизни для отображения обработки запросов.
   - Указал возврат сообщений для отображения результатов.
5. **Проверка диаграммы:**
   После завершения диаграммы я проверил последовательность действий, чтобы убедиться, что она отражает выбранный сценарий.