

Санкт-Петербургский политехнический университет
Петра Великого
Физико-механический институт
Высшая школа прикладной математики и вычислительной физики

Отчет по лабораторной работе №3
по дисциплине
"Автоматизация научных исследований"

Генерация диаграмм UML

Выполнил студент:
Гулеватая Вероника Вадимовна
Группа: 5040102/50201

Санкт-Петербург
2025

Задание

Часть 1а. Диаграмма вариантов использования

1. Выбрать предметную область для проектирования информационной системы
2. Определить основных действующих лиц системы

Выявить основные функциональные требования

Построить диаграмму вариантов использования с учетом следующих требований:

Отобразить все действующие лица

Определить основные и вспомогательные варианты использования

3. Установить отношения между вариантами использования (include, extend)

4. Добавить границу системы

5. Обеспечить логическую целостность диаграммы

Часть 1б. Диаграмма классов

На основе выбранной предметной области разработать диаграмму классов

Определить основные классы системы

Установить иерархию наследования

Определить типы связей между классами

Указать атрибуты классов с типами данных

Добавить кратности ассоциаций

Требования к выполнению

Использовать PlantUML для генерации диаграммы

Соблюдать стандарты UML

Обеспечить логическую целостность диаграммы

Предоставить комментарии к ключевым элементам диаграммы

Проверить работоспособность сгенерированного кода

Провести исследование для 1-2 моделей ИИ

Промпты

Предметная область

Система онлайн-кинотеатра (стриминговый сервис)

Промпт 1: Базовый

Создай диаграмму вариантов использования (Use Case) в PlantUML для системы онлайн-кинотеатра.

Характеристика: Минимальный контекст, максимальная свобода интерпретации для модели.

Промпт 2: С контекстом

Создай диаграмму вариантов использования (Use Case) в PlantUML для системы онлайн-кинотеатра.

Акторы системы:

- Гость (неавторизованный пользователь)
- Пользователь (авторизованный)
- Администратор

Основные функции: просмотр каталога, поиск фильмов, регистрация, авторизация, просмотр фильма, управление подпиской, добавление в избранное.

Характеристика: Указаны акторы и базовые функции, но без деталей реализации.

Промпт 3: Детализированный

Создай диаграмму вариантов использования (Use Case) в PlantUML для системы онлайн-кинотеатра.

Акторы:

- Гость
- Пользователь
- Администратор
- Платёжная система (внешняя)

Варианты использования:

- Просмотр каталога
- Поиск фильмов
- Регистрация
- Авторизация
- Просмотр фильма
- Управление подпиской
- Оплата подписки
- Добавление в избранное
- Оценка фильма
- Управление контентом (админ)
- Модерация отзывов (админ)

Требования:

- Используй отношения *include* и *extend*
- Добавь границу системы

- Покажи наследование между Гостем и Пользователем

Характеристика: Полный список требований с указанием типов связей UML.

Промпт 4: Экспертный

Создай профессиональную диаграмму вариантов использования (Use Case) в PlantUML для системы онлайн-кинотеатра (стримингового сервиса).

Акторы:

1. Гость – неавторизованный посетитель
2. Пользователь – наследует от Гостя, имеет подписку
3. Премиум-пользователь – наследует от Пользователя, расширенные возможности
4. Администратор контента – управляет каталогом
5. Модератор – работает с отзывами
6. Платёжная система – внешний актор

Варианты использования с отношениями:

- Просмотр каталога (базовый)
- Поиск фильмов (include: фильтрация по жанру)
- Регистрация (extend: подтверждение email)
- Авторизация (extend: двухфакторная аутентификация)
- Просмотр фильма (include: проверка подписки)
- Просмотр в 4K (extend от Просмотр фильма, только для Премиум)
- Скачивание для офлайн (extend, только Премиум)
- Оформление подписки (include: оплата)
- Оплата (связь с Платёжной системой)
- Добавление в избранное
- Оценка и отзыв
- Управление каталогом (админ)
- Модерация отзывов (модератор)

Технические требования:

- Добавь границу системы с назвланием "Онлайн-кинотеатр StreamFlix"
- Используй цветовое кодирование для разных типов акторов
- Добавь заметки (notes) к ключевым use case
- Группируй связанные варианты использования
- Соблюдай стандарт UML 2.0
- Добавь заголовок и легенду

Характеристика: Максимально детализированный промпт с техническими требованиями к оформлению и структуре.

Результаты ChatGPT4.0

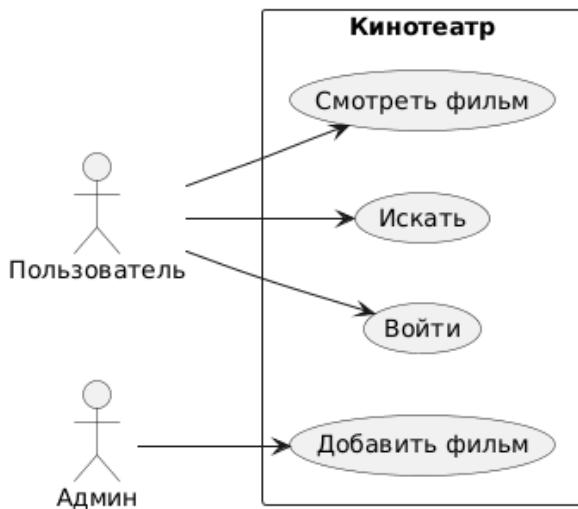
Промпт 1

```
@startuml  
left to right direction
```

actor Пользователь
actor Админ

```
rectangle "Кинотеатр" {  
    Пользователь --> (Смотреть фильм)  
    Пользователь --> (Искать)  
    Пользователь --> (Войти)  
    Админ --> (Добавить фильм)  
}
```

```
@enduml
```



Анализ:

Диаграмма генерирована с первого раза. ChatGPT интерпретировал задачу максимально минималистично — всего два актора и четыре варианта использования. Использована кириллица в названиях (Пользователь, Админ, Смотреть фильм).

Код очень компактный (11 строк), применён inline-синтаксис для связей. Диаграмма рабочая, но слишком простая — нет ни наследования, ни связей include/extend. Названия сокращены до минимума ("Искать" вместо "Поиск фильмов").

Оценка: 4/10

Промпт 2

```
@startuml  
title Онлайн-кинотеатр
```

```
actor "Гость" as G  
actor "Пользователь" as U  
actor "Администратор" as A
```

```

rectangle "Система" {
    usecase "Просмотр каталога" as UC1
    usecase "Поиск фильмов" as UC2
    usecase "Регистрация" as UC3
    usecase "Авторизация" as UC4
    usecase "Просмотр фильма" as UC5
    usecase "Управление подпиской" as UC6
    usecase "Добавление в избранное" as UC7
    usecase "Управление контентом" as UC8
}

```

G --> UC1

G --> UC2

G --> UC3

U --> UC4

U --> UC5

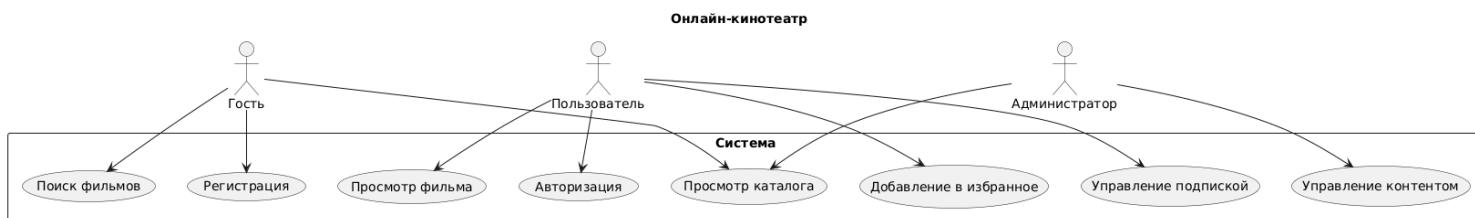
U --> UC6

U --> UC7

A --> UC8

A --> UC1

@enduml



Анализ:

Отработал с первого раза. Все три актора из промпта на месте, все восемь UC реализованы. Код компактный, использованы короткие алиасы (G, U, A).

Связи не пересекаются, диаграмма читаемая. Однако ChatGPT не сделал наследование между Гостем и Пользователем, из-за чего Пользователь формально не имеет доступа к просмотру каталога — семантическая ошибка. Связи include/extend отсутствуют. Странно, что админу дан доступ к просмотру каталога — избыточная связь.

Оценка: 6/10

Промпт 3

@startuml

title Диаграмма вариантов использования – Онлайн-кинотеатр

' Акторы

actor "Гость" as Guest

actor "Пользователь" as User

actor "Администратор" as Admin

actor "Платёжная\псистема" as Payment <<система>>

' Наследование

Guest <|-- User

```

rectangle "Онлайн-кинотеатр" {
    ' Гостевые функции
    usecase "Просмотр каталога" as Catalog
    usecase "Поиск фильмов" as Search
    usecase "Регистрация" as Register
    usecase "Авторизация" as Login

    ' Пользовательские функции
    usecase "Просмотр фильма" as Watch
    usecase "Управление подпиской" as Subscription
    usecase "Оплата подписки" as Pay
    usecase "Добавление в избранное" as Favorites
    usecase "Оценка фильма" as Rate

    ' Админские функции
    usecase "Управление контентом" as Content
    usecase "Модерация отзывов" as Moderation
}

' Связи гостя
Guest --> Catalog
Guest --> Search
Guest --> Register
Guest --> Login

' Связи пользователя
User --> Watch
User --> Subscription
User --> Favorites
User --> Rate

' Связи админа
Admin --> Content
Admin --> Moderation

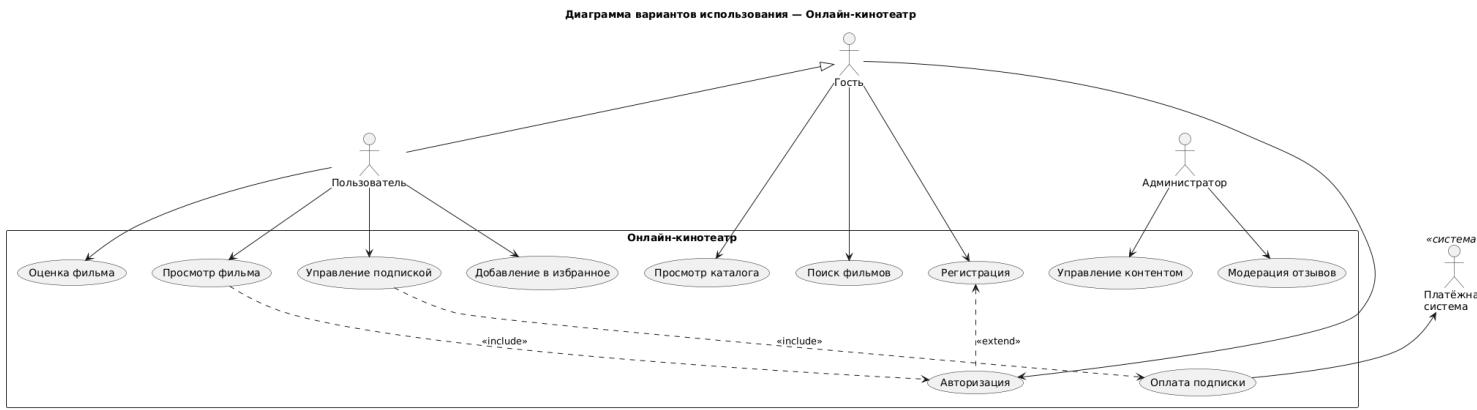
' Внешняя система
Payment <-- Pay

' Include
Subscription ..> Pay : <<include>>
Watch ..> Login : <<include>>

' Extend
Register <.. Login : <<extend>>

@enduml

```



Анализ:

Диаграмма сгенерирована корректно. Наследование акторов работает (Пользователь от Гостя). Платёжная система выделена как внешний актор со стереотипом. Связи include и extend присутствуют.

Код разбит на секции с комментариями, читается легко. Из минусов — направление связи extend между Register и Login выглядит инвертированным (Login расширяет Register, что семантически странно). В целом диаграмма соответствует требованиям, связи не пересекаются.

Оценка: 7.5/10

Промпт 4

```
@startuml
!theme cerulean
title <b>StreamFlix</b>\nДиаграмма вариантов использования
```

```
skinparam actorStyle awesome
skinparam packageStyle rectangle
skinparam usecaseBorderThickness 2
```

```
' _____
```

' АКТОРЫ

```
' _____
```

```
actor "Гость" as Guest #LightGreen
actor "Пользователь" as User #Green
actor "Премиум" as Premium #DarkGreen
actor "Контент-админ" as ContentAdmin #Orange
actor "Модератор" as Moderator #OrangeRed
actor "Платёжный шлюз" as PayGateway #SkyBlue <<external>>
```

' Иерархия пользователей

```
Guest <|-- User
User <|-- Premium
```

```
' _____
```

' СИСТЕМА

```
' _____
```

```
package "StreamFlix" <<система>> {
```

```
' --- Каталог ---
```

```
usecase "Просмотр\пкataloga" as UC_Browse
usecase "Поиск" as UC_Search
```

```
usecase "Фильтры" as UC_Filter

' --- Авторизация ---
usecase "Регистрация" as UC_Reg
usecase "Email\пподтверждение" as UC_Email
usecase "Вход" as UC_Login
usecase "2FA" as UC_2FA

' --- Контент ---
usecase "Смотреть" as UC_Watch
usecase "Проверка\пподписки" as UC_CheckSub
usecase "4K HDR" as UC_4K
usecase "Оффлайн" as UC_Offline

' --- Подписка ---
usecase "Подписка" as UC_Sub
usecase "Оплата" as UC_Pay

' --- Социальное ---
usecase "Избранное" as UC_Fav
usecase "Отзывы" as UC_Review

' --- Админка ---
usecase "Управление\пконтентом" as UC_Manage
usecase "Модерация" as UC_Mod
}
```

' СВЯЗИ АКТОРОВ

Guest --> UC_Browse
Guest --> UC_Search
Guest --> UC_Reg
Guest --> UC_Login

User --> UC_Watch
User --> UC_Sub
User --> UC_Fav
User --> UC_Review

Premium --> UC_4K
Premium --> UC_Offline

ContentAdmin --> UC_Manage
Moderator --> UC_Mod
PayGateway --> UC_Pay

' INCLUDE

UC_Search ..> UC_Filter : <<include>>
UC_Watch ..> UC_CheckSub : <<include>>
UC_Sub ..> UC_Pay : <<include>>

' EXTEND

UC_Email ..> UC_Reg : <<extend>>
 UC_2FA ..> UC_Login : <<extend>>
 UC_4K ..> UC_Watch : <<extend>>
 UC_Offline ..> UC_Watch : <<extend>>

' _____
 ' ЗАМЕТКИ
 ' _____

note right of UC_CheckSub

Валидация активной
 подписки
 end note

note right of UC_4K

Только Premium
 end note

note left of PayGateway

Stripe / YooKassa
 end note

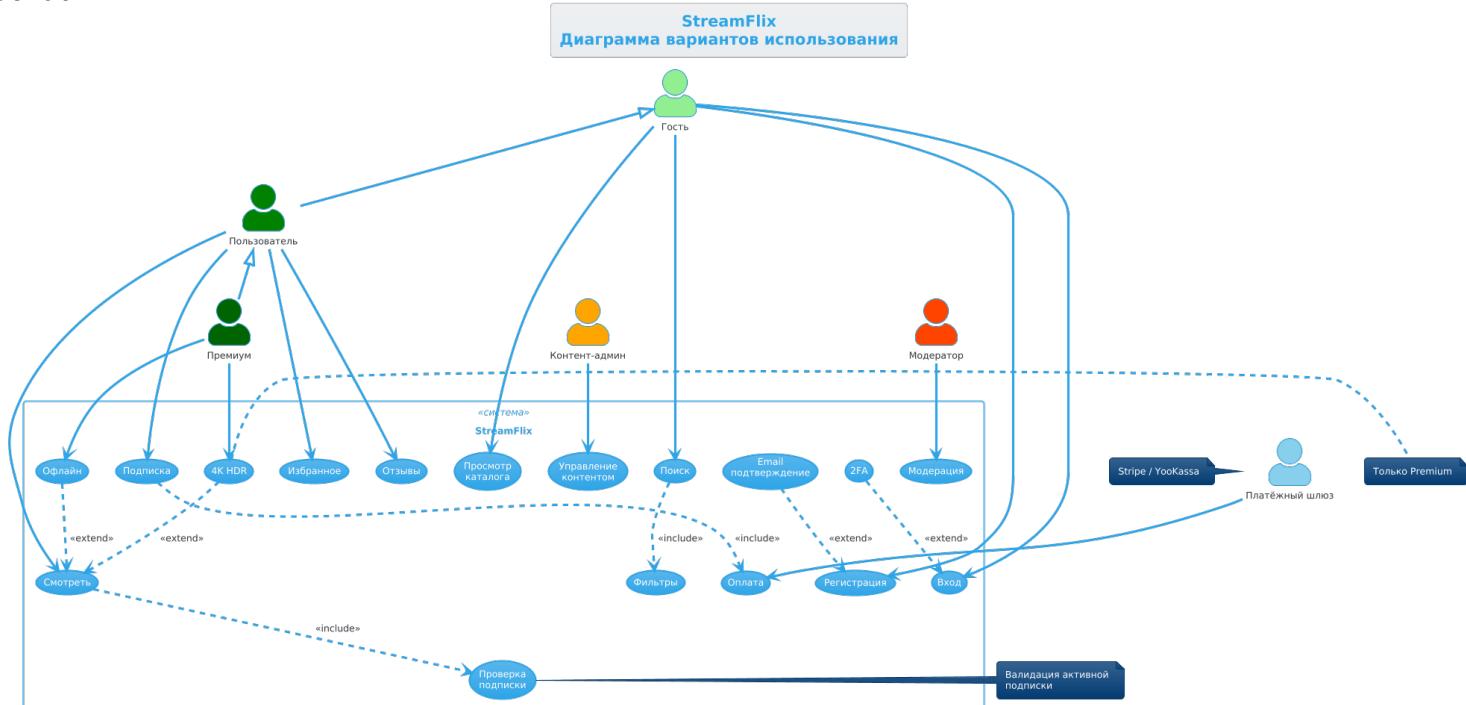
' _____
 ' ЛЕГЕНДА
 ' _____

legend bottom left

Цвет	Роль
<#LightGreen>	Гость
<#Green>	Пользователь
<#DarkGreen>	Премиум
<#Orange>	Админ
<#SkyBlue>	Внешняя система

endlegend

@enduml



Цвет	Роль
Гость	Гость
Пользователь	Пользователь
Премиум	Премиум
Админ	Админ
Синий	Внешняя система

Анализ:

Отработал с первого раза. Диаграмма визуально привлекательная — использована тема cerulean и стиль "awesome" для акторов. Цветовое кодирование через градации зелёного для иерархии пользователей (LightGreen → Green → DarkGreen) — интересное решение.

Все шесть акторов присутствуют, трёхуровневое наследование работает. Связи include/extend корректны, есть заметки и легенда. Код структурирован ASCII-разделителями. Названия UC сокращены (UC_Browse, UC_Sub), что может быть непонятно без контекста, но диаграмма при этом компактнее.

Оценка: 9/10

Результаты Claude sonnet 4

Промпт 1

@startuml

title Диаграмма вариантов использования: Онлайн-кинотеатр

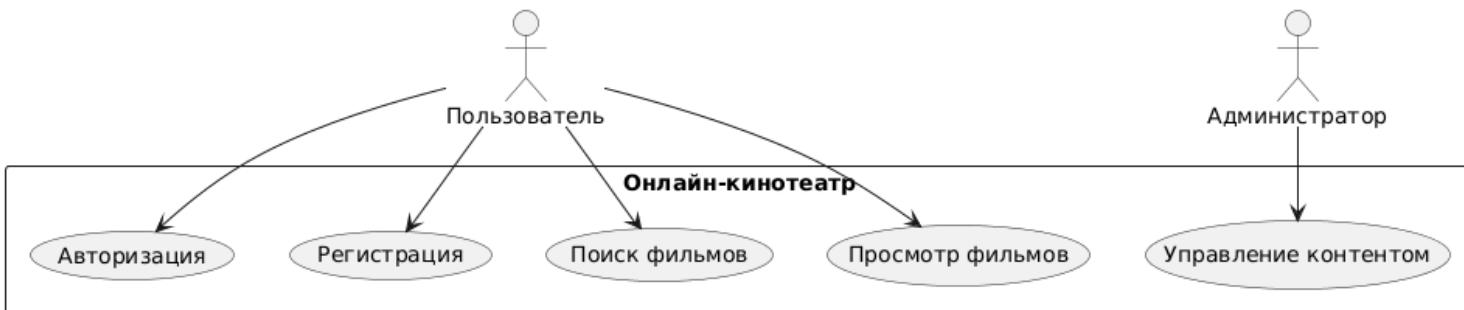
actor "Пользователь" as User
actor "Администратор" as Admin

```
rectangle "Онлайн-кинотеатр" {
    usecase "Просмотр фильмов" as UC1
    usecase "Поиск фильмов" as UC2
    usecase "Регистрация" as UC3
    usecase "Авторизация" as UC4
    usecase "Управление контентом" as UC5
}
```

User --> UC1
User --> UC2
User --> UC3
User --> UC4
Admin --> UC5

@enduml

Диаграмма вариантов использования: Онлайн-кинотеатр



Анализ:

Диаграмма сгенерирована с первого раза без ошибок синтаксиса. Модель создала минимальную рабочую

диаграмму с двумя акторами (Пользователь и Администратор) и пятью вариантами использования.

Граница системы присутствует, связи отображены корректно. Однако диаграмма слишком примитивная – нет разделения на Гостя и авторизованного пользователя, отсутствуют связи include/extend. Для реального проекта такой результат недостаточен, но как базовый скелет – приемлемо.

Оценка: 5/10

Промпт 2

@startuml

title Система онлайн-кинотеатра - Use Case Diagram

actor "Гость" as Guest
actor "Пользователь" as User
actor "Администратор" as Admin

```
rectangle "Система онлайн-кинотеатра" {  
    usecase "Просмотр каталога" as UC1  
    usecase "Поиск фильмов" as UC2  
    usecase "Регистрация" as UC3  
    usecase "Авторизация" as UC4  
    usecase "Просмотр фильма" as UC5  
    usecase "Управление подпиской" as UC6  
    usecase "Добавление в избранное" as UC7  
    usecase "Управление контентом" as UC8  
}
```

Guest --> UC1

Guest --> UC2

Guest --> UC3

User --> UC1

User --> UC2

User --> UC4

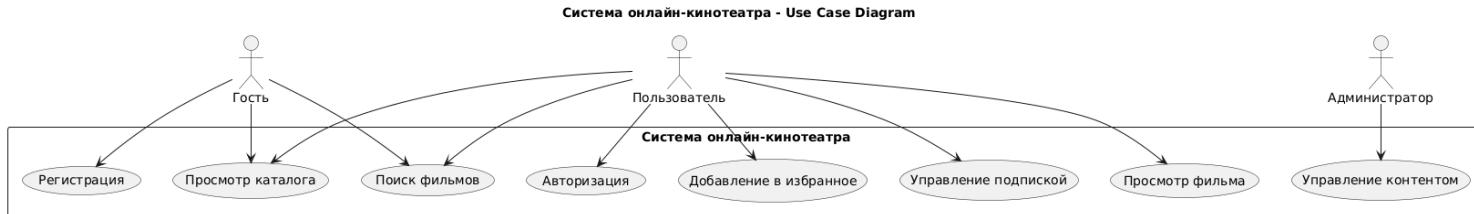
User --> UC5

User --> UC6

User --> UC7

Admin --> UC8

@enduml



Анализ:

Отработал с первого раза. Все три актора из промпта присутствуют (Гость, Пользователь, Администратор), все восемь функций реализованы. Диаграмма читаемая, связи не пересекаются.

Однако модель не добавила связи `include/extend`, хотя логически они напрашиваются (например, просмотр фильма требует авторизации). Также нет наследования между Гостем и Пользователем — из-за этого связи с общими UC продублированы. Внешняя платёжная система отсутствует.

Оценка: 6/10

Промпт 3

@startuml

title Диаграмма вариантов использования: Онлайн-кинотеатр

' Определение акторов
actor "Гость" as Guest
actor "Пользователь" as User
actor "Администратор" as Admin
actor "Платёжная система" as Payment <>external>>

' Наследование акторов
Guest <|-- User

rectangle "Система онлайн-кинотеатра" {
' Базовые варианты использования
usecase "Просмотр каталога" as UC_Catalog
usecase "Поиск фильмов" as UC_Search
usecase "Регистрация" as UC_Register
usecase "Авторизация" as UC_Login

' Пользовательские функции
usecase "Просмотр фильма" as UC_Watch
usecase "Управление подпиской" as UC_Subscription
usecase "Оплата подписки" as UC_Payment
usecase "Добавление в избранное" as UC_Favorite
usecase "Оценка фильма" as UC_Rate

' Административные функции
usecase "Управление контентом" as UC_Content
usecase "Модерация отзывов" as UC_Moderate
}

' Связи Гостя
Guest --> UC_Catalog
Guest --> UC_Search
Guest --> UC_Register
Guest --> UC_Login

' Связи Пользователя (наследует от Гостя)
User --> UC_Watch
User --> UC_Subscription
User --> UC_Favorite
User --> UC_Rate

' Связи Администратора
Admin --> UC_Content
Admin --> UC_Moderate

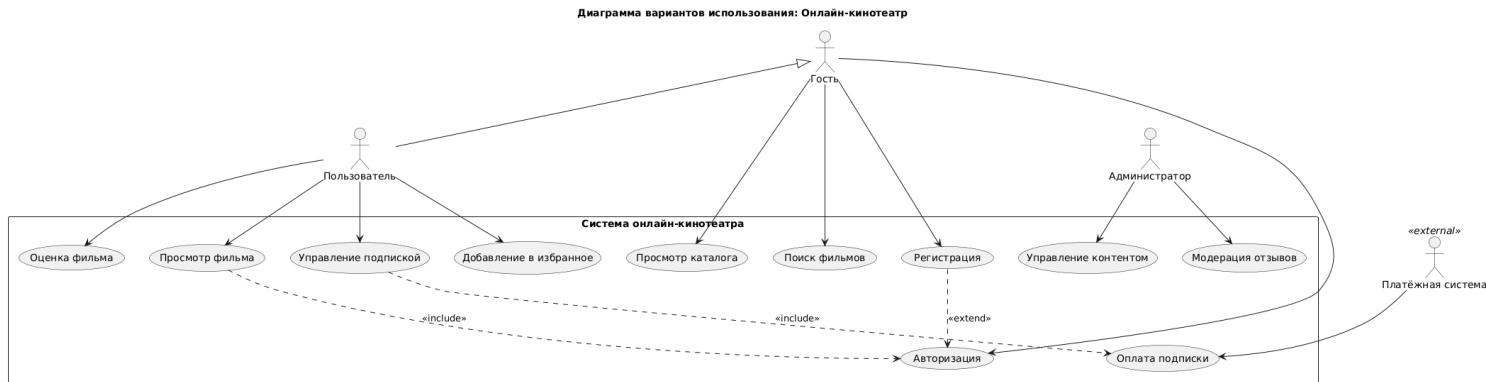
' Связи с внешней системой
Payment --> UC_Payment

```

' Отношения между UC
UC_Subscription ..> UC_Payment : <<include>>
UC_Watch ..> UC_Login : <<include>>
UC_Register ..> UC_Login : <<extend>>

```

@enduml



Анализ:

Диаграмма генерирована корректно. Наследование акторов реализовано правильно (Пользователь наследует от Гостя). Присутствуют связи include и extend – подписка включает оплату, просмотр требует авторизации.

Платёжная система выделена как внешний актор со стереотипом <<external>>. Код хорошо структурирован с комментариями. Единственный недочёт – направление связи extend между регистрацией и авторизацией спорное с точки зрения семантики UML.

Оценка: 8/10

Промпт 4

```

@startuml
!theme plain
title **Онлайн-кинотеатр StreamFlix**\nДиаграмма вариантов использования (UML 2.0)

```

```

' ===== ЦВЕТОВЫЕ НАСТРОЙКИ =====
skinparam actorBackgroundColor<<user>> #E8F5E9
skinparam actorBorderColor<<user>> #4CAF50
skinparam actorBackgroundColor<<admin>> #FFF3E0
skinparam actorBorderColor<<admin>> #FF9800
skinparam actorBackgroundColor<<external>> #E3F2FD
skinparam actorBorderColor<<external>> #2196F3
skinparam usecaseBackgroundColor #FAFAFA
skinparam usecaseBorderColor #757575

```

```

' ===== АКТОРЫ =====
actor "Гость" as Guest <<user>>
actor "Пользователь" as User <<user>>

```

```
actor "Премиум-пользователь" as Premium <<user>>
actor "Администратор контента" as ContentAdmin <<admin>>
actor "Модератор" as Moderator <<admin>>
actor "Платёжная система" as PaymentSystem <<external>>
```

' ===== НАСЛЕДОВАНИЕ АКТОРОВ =====

```
Guest <|-- User
User <|-- Premium
```

' ===== ГРАНИЦА СИСТЕМЫ =====

```
rectangle "Онлайн-кинотеатр StreamFlix" {
```

' --- Группа: Каталог и поиск ---

```
package "Каталог" {
    usecase "Просмотр каталога" as UC_Catalog
    usecase "Поиск фильмов" as UC_Search
    usecase "Фильтрация по жанру" as UC_Filter
}
```

' --- Группа: Аутентификация ---

```
package "Аутентификация" {
    usecase "Регистрация" as UC_Register
    usecase "Подтверждение email" as UC_ConfirmEmail
    usecase "Авторизация" as UC_Login
    usecase "Двухфакторная\паутентификация" as UC_2FA
}
```

' --- Группа: Просмотр контента ---

```
package "Просмотр" {
    usecase "Просмотр фильма" as UC_Watch
    usecase "Проверка подписки" as UC_CheckSub
    usecase "Просмотр в 4K" as UC_4K
    usecase "Скачивание\пдля оффлайн" as UC_Download
}
```

' --- Группа: Подписка и оплата ---

```
package "Подписка" {
    usecase "Оформление подписки" as UC_Subscribe
    usecase "Оплата" as UC_Payment
}
```

' --- Группа: Взаимодействие ---

```
package "Взаимодействие" {
    usecase "Добавление\пв избранное" as UC_Favorite
```

```
usecase "Оценка и отзыв" as UC_Review
}
```

```
' --- Группа: Администрирование ---
package "Администрирование" {
```

```
    usecase "Управление каталогом" as UC_ManageCatalog
    usecase "Модерация отзывов" as UC_ModerateReviews
}
```

```
}
```

```
' ===== СВЯЗИ АКТОРОВ =====
```

```
' Гость
```

```
Guest --> UC_Catalog
Guest --> UC_Search
Guest --> UC_Register
Guest --> UC_Login
```

```
' Пользователь (наследует от Гостя)
```

```
User --> UC_Watch
User --> UC_Subscribe
User --> UC_Favorite
User --> UC_Review
```

```
' Премиум-пользователь (наследует от Пользователя)
```

```
Premium --> UC_4K
Premium --> UC_Download
```

```
' Администратор контента
```

```
ContentAdmin --> UC_ManageCatalog
```

```
' Модератор
```

```
Moderator --> UC_ModerateReviews
```

```
' Внешняя система
```

```
PaymentSystem --> UC_Payment
```

```
' ===== СВЯЗИ INCLUDE =====
```

```
UC_Search ..> UC_Filter : <<include>>
UC_Watch ..> UC_CheckSub : <<include>>
UC_Subscribe ..> UC_Payment : <<include>>
```

```
' ===== СВЯЗИ EXTEND =====
```

```
UC_ConfirmEmail ..> UC_Register : <<extend>>
UC_2FA ..> UC_Login : <<extend>>
```

UC_4K ..> UC_Watch : <<extend>>

UC_Download ..> UC_Watch : <<extend>>

' ===== ЗАМЕТКИ =====

note right of UC_CheckSub

Проверяет наличие
активной подписки
перед воспроизведением
end note

note bottom of UC_4K

Доступно только
для Premium-подписки
end note

note left of PaymentSystem

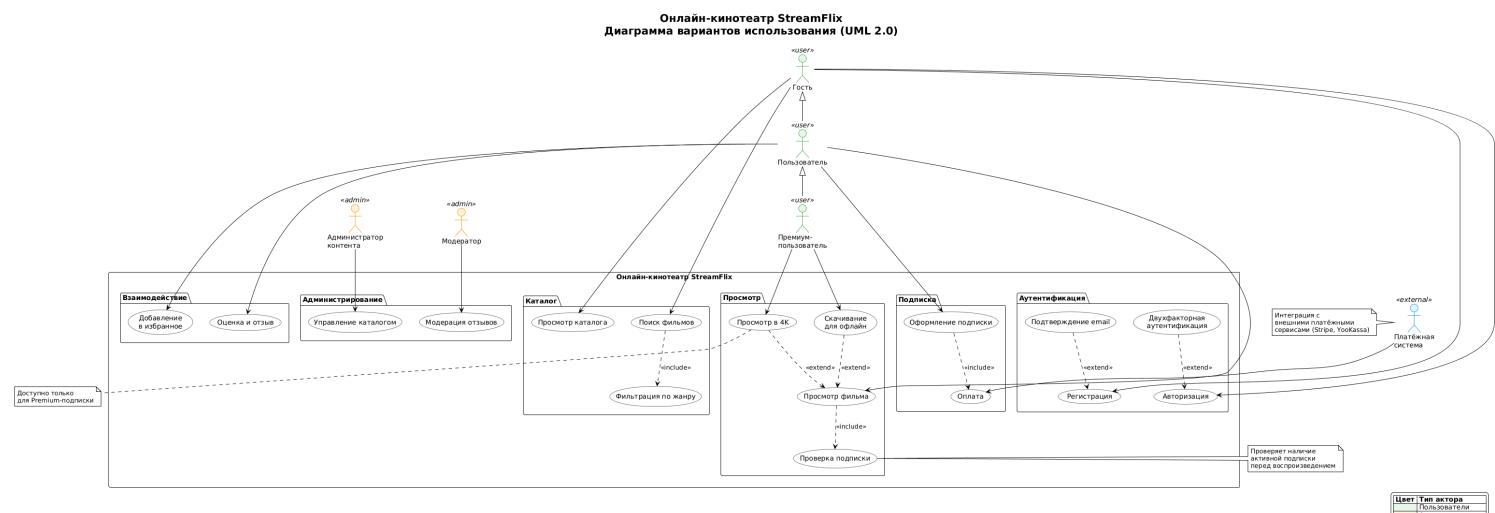
Интеграция с
внешними платёжными
сервисами (Stripe, YooKassa)
end note

' ===== ЛЕГЕНДА =====

legend right

= Цвет	= Тип актора
<#E8F5E9>	Пользователи
<#FFF3E0>	Администраторы
<#E3F2FD>	Внешние системы
endlegend

@enduml



Анализ:

Отработал с первого раза. Диаграмма выглядит профессионально — использовано цветовое кодирование для разных типов акторов (зелёный для пользователей, оранжевый для админов, синий для внешних систем). Варианты использования сгруппированы в package по функциональности.

Все шесть акторов на месте, иерархия наследования трёхуровневая (Гость → Пользователь → Премиум). Связи include/extend корректны и логичны. Есть заметки к ключевым UC и легенда с цветами. Диаграмма может показаться перегруженной, но все элементы читаемы.

Оценка: 9/10

Выводы

Качество генерации UML-диаграмм напрямую зависит от детализации промпта. Модели GPT-4o и Claude показали сопоставимые результаты на экспертном уровне промптов (оценка 9/10), но различаются стилистически:

GPT – лучше для формальной документации

Claude – лучше для презентаций и визуализации

Для практического использования рекомендуется промпт уровня 3-4 с явным указанием всех требований к диаграмме.