

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

## **ЛАБОРАТОРНАЯ РАБОТА №3**

**Генерация диаграмм UML**  
по дисциплине «Автоматизация научных исследований»

Выполнил

Студент гр. 5040102/50201 <подпись>

Тухватуллина Г.М.

Проверил

проф.,д.т.н.

<подпись>

Новиков Ф.А.

«28» декабря 2025 г.

Санкт-Петербург

2025

## **Цель**

Разработать и проанализировать диаграмму UML (а или б) для информационной системы по выбору студента.

## **Задание**

### 1. Часть 1а. Диаграмма вариантов использования

- Выбрать предметную область для проектирования информационной системы
- Определить основных действующих лиц системы
- Выявить основные функциональные требования
- Построить диаграмму вариантов использования с учетом следующих требований:
  1. Отобразить все действующие лица
  2. Определить основные и вспомогательные варианты использования
  3. Установить отношения между вариантами использования (include, extend)
  4. Добавить границу системы
  5. Обеспечить логическую целостность диаграммы

### 2. 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

### **Критерии оценивания**

- Корректность синтаксиса PlantUML
- Соответствие стандартам UML
- Полнота отображения функциональных требований
- Качество визуализации
- Логическая целостность диаграмм
- Обоснованность выбора предметной области

### **Форма отчёта**

1. Исходный код PlantUML для обеих диаграмм
2. Сгенерированная диаграмма
3. Пояснительная записка с описанием:
  - 1) Выбранной предметной области
  - 2) Обоснования выбора действующих лиц
  - 3) Функциональных требований
  - 4) Структуры классов
4. Анализ результатов проектирования

## **Ход работы**

Для выполнения данной лабораторной работы использована большая языковая модель – DeepSeek.

Реализация кода PlantUML производилась в онлайн-сервисе:  
<https://editor.plantuml.com/>

### **1. Описание выбранной информационной системы**

Для более корректного оценивания и сравнения сгенерированных диаграмм UML предметная область и информационная система взяты из курса «Формализация моделирования», на котором были предоставлены оба типа полностью правильных диаграмм.

Информационная система «Отдел кадров» (сокращённо ИС ОК) предназначена для ввода, хранения и обработки информации о сотрудниках и движении кадров.

Система должна обеспечивать выполнение следующих основных функций:

1. Приём, перевод и увольнение сотрудников;
2. Создание и ликвидация подразделений;
3. Создание вакансий и сокращение должностей.

Информационная система отдела кадров должна учитывать и отражать следующие структурные особенности:

1. Наличие и изменчивость иерархической организационной структуры предприятия;
2. Наличие отношений субординации, определяемых организационной структурой и номенклатурой должностей;
3. Наличие временных рабочих групп, образуемых для выполнения проектов.

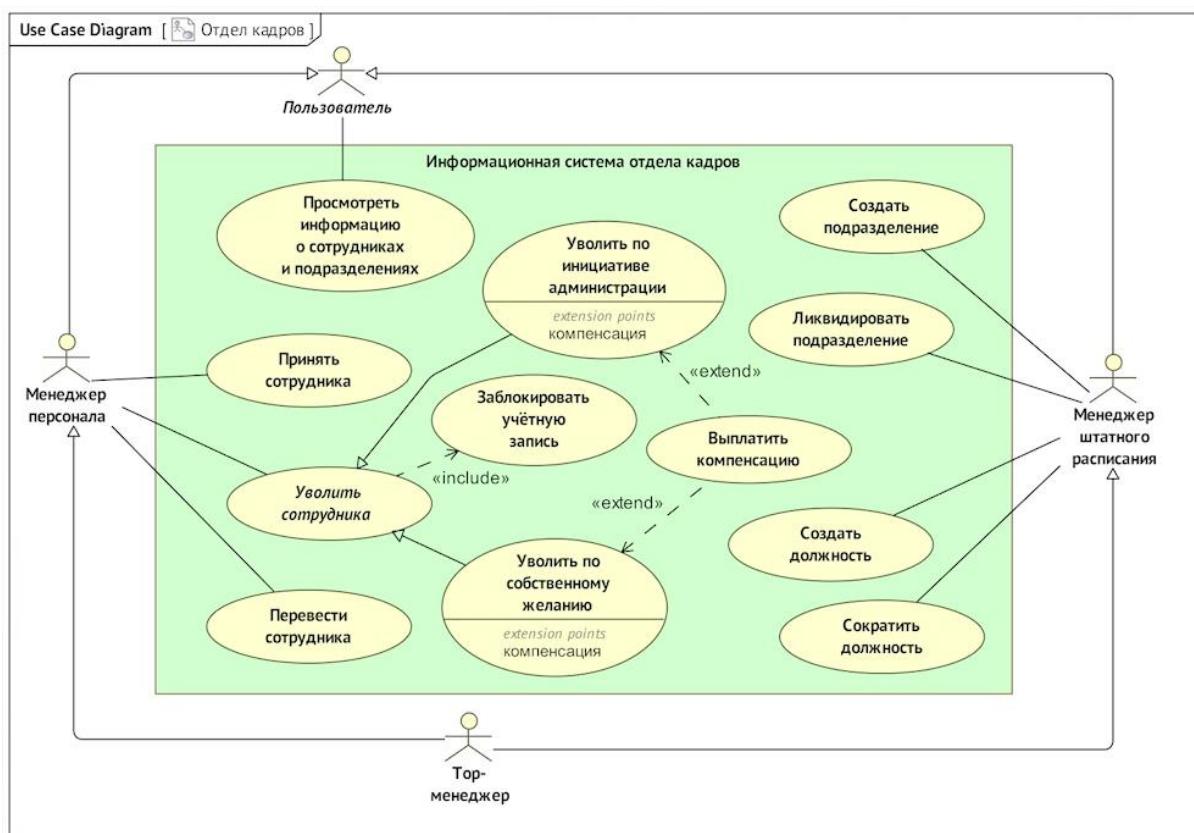
### **Основные действующие лица системы**

Менеджер персонала – принимает, увольняет и переводит сотрудников, а также просматривает информацию о сотрудниках и подразделениях.

Менеджер штатного расписания – создаёт подразделения и должности, ликвидирует подразделения и сокращает должности, а также просматривает информацию о сотрудниках и подразделениях.

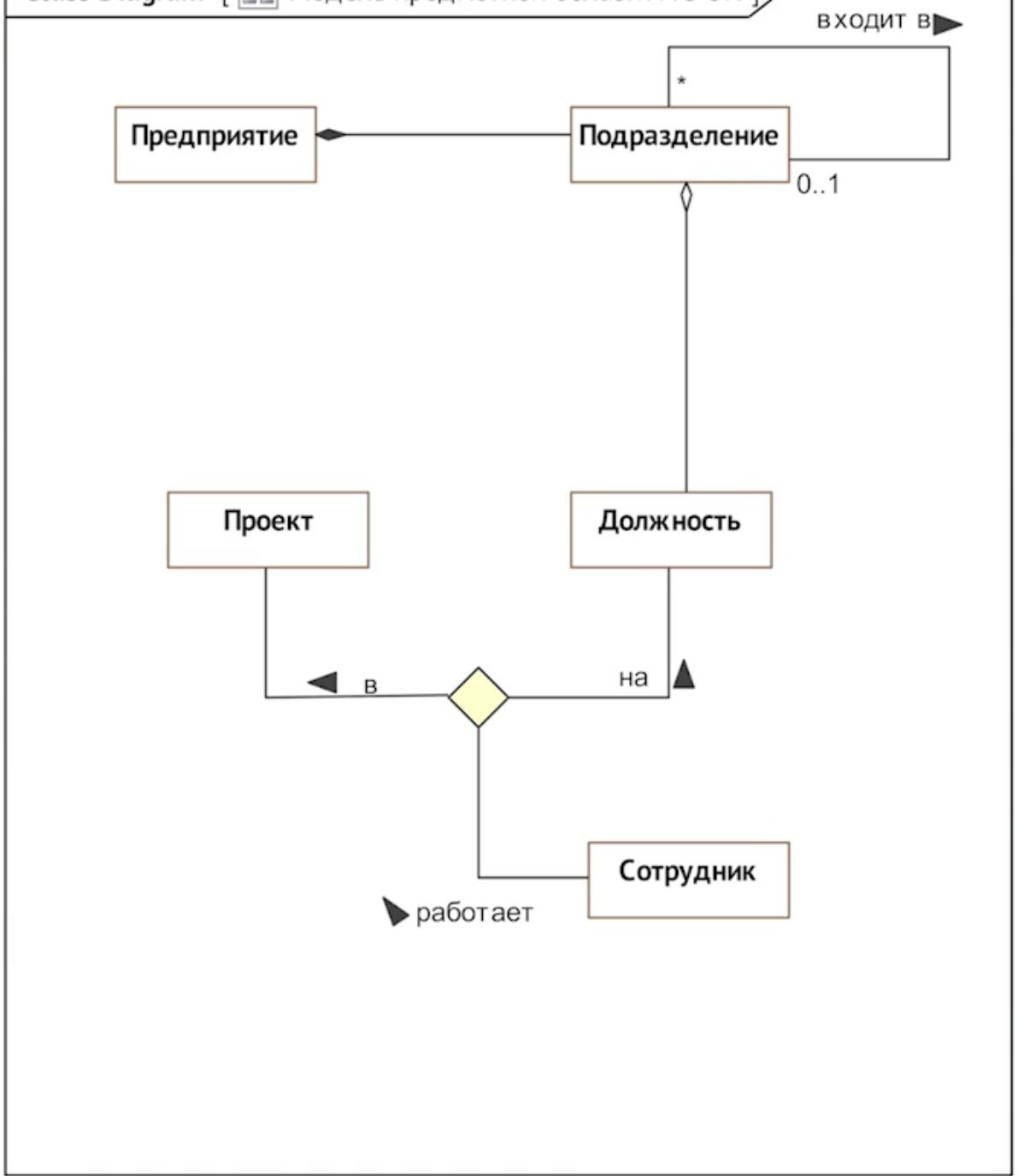
Топ-менеджер – может выполнять функции менеджера персонала и менеджера штатного расписания.

**Исходная диаграмма вариантов использования:**



**Исходная диаграмма классов:**

Class Diagram [ Модель предметной области ИС ОК ]



## 2. Генерация диаграмм UML

### Промпт 1 – ДИ

Сгенерируй PlantUML код для диаграммы вариантов использования информационной системы отдела кадров. Укажи действующие лица, перечисли основные случаи использования.

### Ответ

plantuml

```
@startuml
left to right direction
skinparam packageStyle rectangle
skinparam usecaseBackgroundColor #F9F7F4
skinparam usecaseBorderColor #333333

actor "Сотрудник отдела кадров" as hr
actor "Сотрудник" as employee
actor "Руководитель" as manager
actor "Система бухгалтерии" as accounting
actor "Администратор системы" as admin

rectangle "Информационная система отдела кадров" {
    usecase "Управление персоналом" as UC1
    usecase "Ведение кадрового документооборота" as UC2
    usecase "Оформление приема на работу" as UC3
    usecase "Оформление перевода" as UC4
    usecase "Оформление увольнения" as UC5
    usecase "Учет отпусков и отгулов" as UC6
    usecase "Учет больничных листов" as UC7
    usecase "Учет командировок" as UC8
    usecase "Формирование отчетов" as UC9
    usecase "Просмотр кадровой статистики" as UC10
    usecase "Планирование отпусков" as UC11
    usecase "Подача заявлений" as UC12
    usecase "Просмотр личных данных" as UC13
    usecase "Согласование заявлений" as UC14
    usecase "Просмотр отчетов" as UC15
    usecase "Интеграция с бухгалтерией" as UC16}
```

```
usecase "Управление правами доступа" as UC17
usecase "Резервное копирование данных" as UC18
}
```

' Связи с сотрудником отдела кадров

hr --> UC1

hr --> UC2

hr --> UC3

hr --> UC4

hr --> UC5

hr --> UC6

hr --> UC7

hr --> UC8

hr --> UC9

hr --> UC10

hr --> UC11

' Связи с сотрудником

employee --> UC12

employee --> UC13

employee --> UC6

' Связи с руководителем

manager --> UC14

manager --> UC15

manager --> UC11

' Связи с внешней системой

accounting --> UC16

UC16 --> accounting

' Связи с администратором

admin --> UC17

admin --> UC18

' Включения и расширения

UC3 .> UC2 : включает

UC4 .> UC2 : включает

UC5 .> UC2 : включает

UC9 .> UC1 : включает

UC12 .> UC6 : расширяет

UC12 .> UC7 : расширяет

UC12 .> UC8 : расширяет  
UC14 .> UC12 : расширяет

```
note top of UC2
    Приказы, трудовые договоры,
    дополнительные соглашения,
    личные карточки
end note

note right of UC16
    Передача данных о
    отработанном времени,
    отпусках, больничных
end note
@enduml
```

Описание основных элементов диаграммы:

Действующие лица (акторы):

1. Сотрудник отдела кадров — основной пользователь системы.
2. Сотрудник — просматривает личные данные и подает заявления.
3. Руководитель — согласует заявления и просматривает отчеты.
4. Система бухгалтерии — внешняя система для обмена данными.
5. Администратор системы — управляет доступом и резервным копированием.

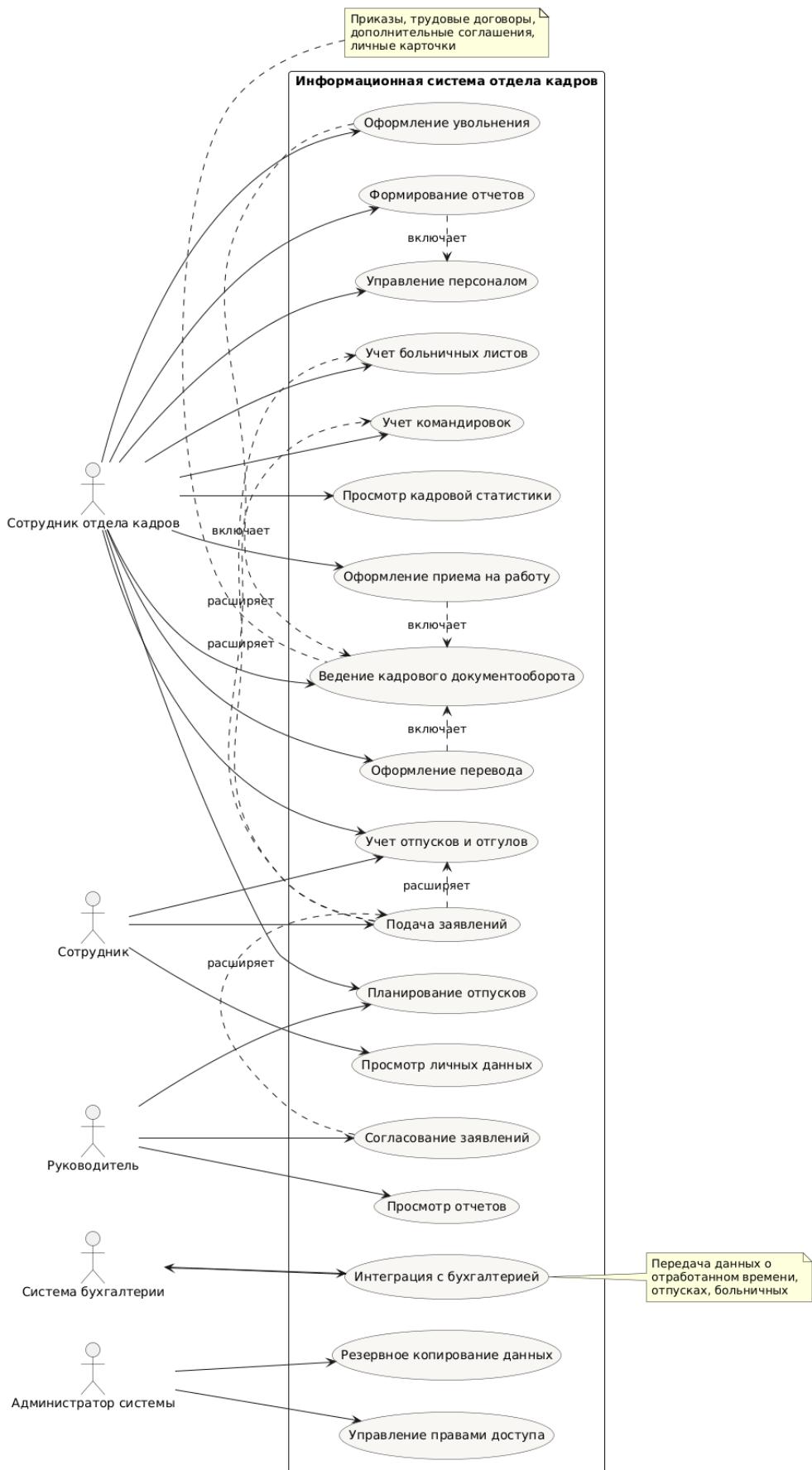
Основные варианты использования:

- Управление персоналом (базовый сценарий).
- Ведение кадрового документооборота (оформление приказов, договоров).
- Оформление кадровых действий (прием, перевод, увольнение).
- Учет отсутствий (отпуска, больничные, командировки).
- Формирование отчетности и аналитика.

- Планирование отпусков.
- Подача и согласование заявлений сотрудниками и руководителями.
- Интеграция с бухгалтерией для передачи данных.
- Администрирование системы (права доступа, резервное копирование).

Диаграмма отображает границы системы, основных участников и ключевые бизнес-процессы типовой HR-системы.

## Диаграмма



## Промпт 2 – ДИ

Сгенерируй PlantUML код для диаграммы вариантов использования информационной системы отдела кадров. Укажи действующие лица: менеджер персонала, менеджер штатного расписания, топ-менеджер. Перечисли основные случаи использования и добавь границу системы.

## Ответ

Plantuml

```
@startuml
left to right direction
skinparam usecase {
    backgroundColor LightSkyBlue
    borderColor DarkSlateGray
}

rectangle "Информационная система отдела кадров" {
    (Управление данными сотрудников) as (UC1)
    (Оформление кадровых документов) as (UC2)
    (Формирование отчетности) as (UC3)
    (Управление отпусками и больничными) as (UC4)
    (Ведение штатного расписания) as (UC5)
    (Утверждение штатного расписания) as (UC6)
    (Анализ персонала) as (UC7)
    (Утверждение приема/увольнения) as (UC8)

    (UC2) .> (UC1) : включает
    (UC3) .> (UC1) : включает
}

actor "Менеджер персонала" as HRManager
actor "Менеджер штатного расписания" as StaffingManager
actor "Топ-менеджер" as TopManager

HRManager --> (UC1)
HRManager --> (UC2)
HRManager --> (UC4)
HRManager --> (UC7)

StaffingManager --> (UC5)
```

StaffingManager --> (UC3)

TopManager --> (UC6)

TopManager --> (UC3)

TopManager --> (UC8)

TopManager --> (UC7)

note right of HRManager

Работает с личными данными,

документами и отпусками

сотрудников

end note

note left of TopManager

Утверждает ключевые изменения

и анализирует отчеты

end note

@enduml

Описание диаграммы:

Граница системы: Прямоугольник "Информационная система отдела кадров".

Основные варианты использования:

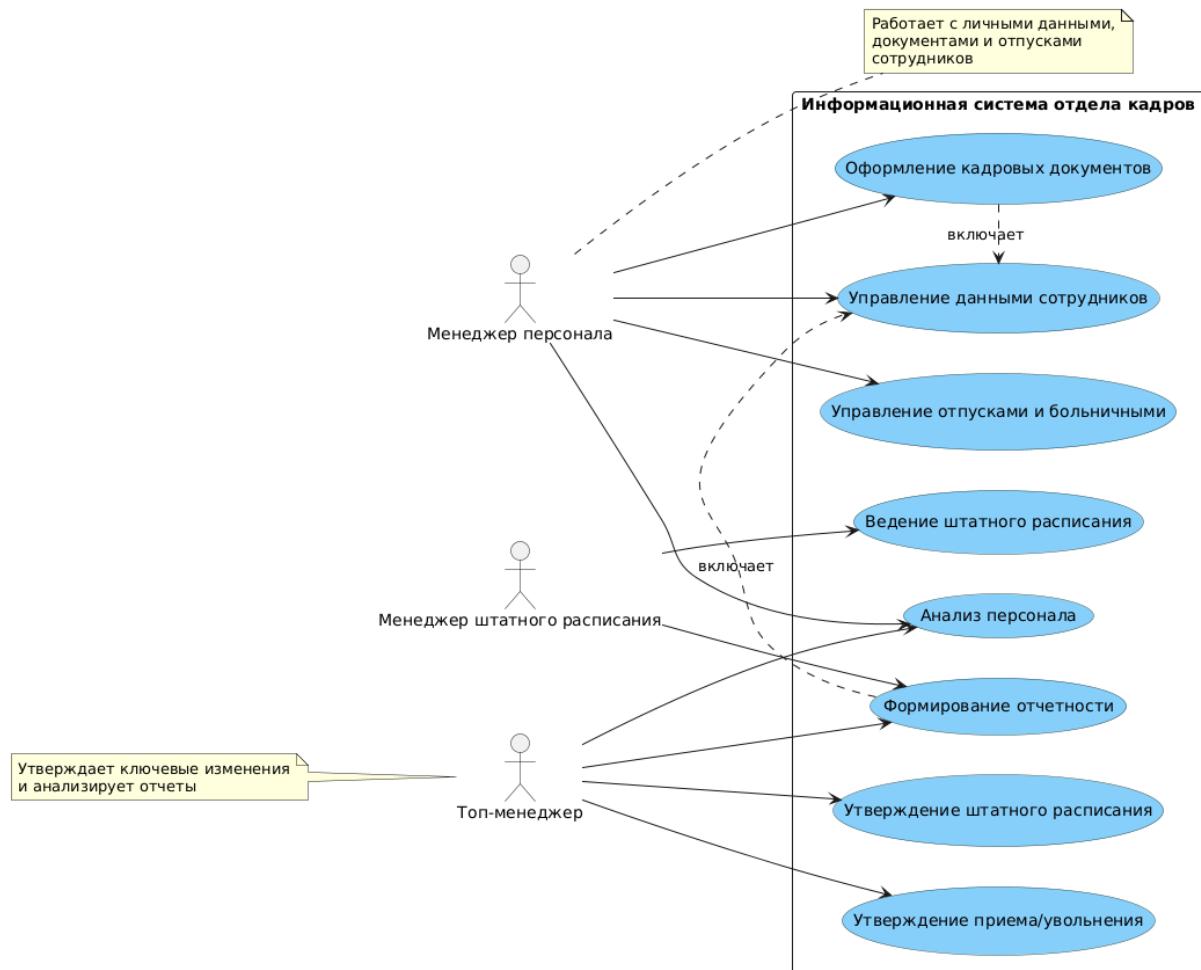
1. Управление данными сотрудниками - ведение личных дел, контактной информации, истории работы.
2. Оформление кадровых документов - приказы о приеме, переводе, увольнении.
3. Формирование отчетности - создание регламентированных и аналитических отчетов.
4. Управление отпусками и больничными - учет и планирование времени отсутствия.
5. Ведение штатного расписания - управление должностями, ставками, структурами подразделений.
6. Утверждение штатного расписания - согласование изменений в организационной структуре.

7. Анализ персонала - получение аналитических данных по кадровому составу.
8. Утверждение приема/увольнения - согласование ключевых кадровых решений.

Связи:

- Include-отношения показывают, что для формирования отчетов и документов необходимы данные сотрудников
- Каждый актер связан с соответствующими его роли вариантами использования

### Диаграмма



### **Промпт 3 – ДИ**

Сгенерируй PlantUML код для диаграммы вариантов использования информационной системы отдела кадров. Укажи действующие лица и перечисли случаи использования по ролям.

Менеджер персонала – принимает, увольняет и переводит сотрудников, а также просматривает информацию о сотрудниках и подразделениях.

Менеджер штатного расписания – создаёт подразделения и должности, ликвидирует подразделения и сокращает должности, а также просматривает информацию о сотрудниках и подразделениях.

Топ-менеджер – может выполнять функции менеджера персонала и менеджера штатного расписания.

Отобрази все связи между действующими лицами и соответствующими вариантами использования. Включи границу системы.

### **Ответ**

#### **Plantuml**

```
@startuml
left to right direction
skinparam packageStyle rectangle

rectangle "Информационная система отдела кадров" {
    (Просмотр информации о сотрудниках) as uc1
    (Просмотр информации о подразделениях) as uc2

    (Прием сотрудника) as uc3
    (Увольнение сотрудника) as uc4
    (Перевод сотрудника) as uc5

    (Создание подразделения) as uc6
    (Создание должности) as uc7
    (Ликвидация подразделения) as uc8
    (Сокращение должности) as uc9
}
```

```

actor "Менеджер персонала" as PersonnelManager
actor "Менеджер штатного расписания" as StaffingManager
actor "Топ-менеджер" as TopManager <<генерализация>>

PersonnelManager --> uc1
PersonnelManager --> uc2
PersonnelManager --> uc3
PersonnelManager --> uc4
PersonnelManager --> uc5

StaffingManager --> uc1
StaffingManager --> uc2
StaffingManager --> uc6
StaffingManager --> uc7
StaffingManager --> uc8
StaffingManager --> uc9

TopManager --|> PersonnelManager
TopManager --|> StaffingManager

```

@enduml

Описание диаграммы:

Граница системы: Прямоугольник обозначает границы информационной системы отдела кадров.

Действующие лица (акторы):

1. Менеджер персонала – работает с кадровыми операциями.
2. Менеджер штатного расписания – управляет структурой организации.
3. Топ-менеджер – наследует полномочия обоих ролей (связь-генерализация, показана стрелкой с треугольным наконечником).

Варианты использования по ролям:

Для Менеджера персонала:

- Просмотр информации о сотрудниках (uc1)
- Просмотр информации о подразделениях (uc2)
- Прием сотрудника (uc3)

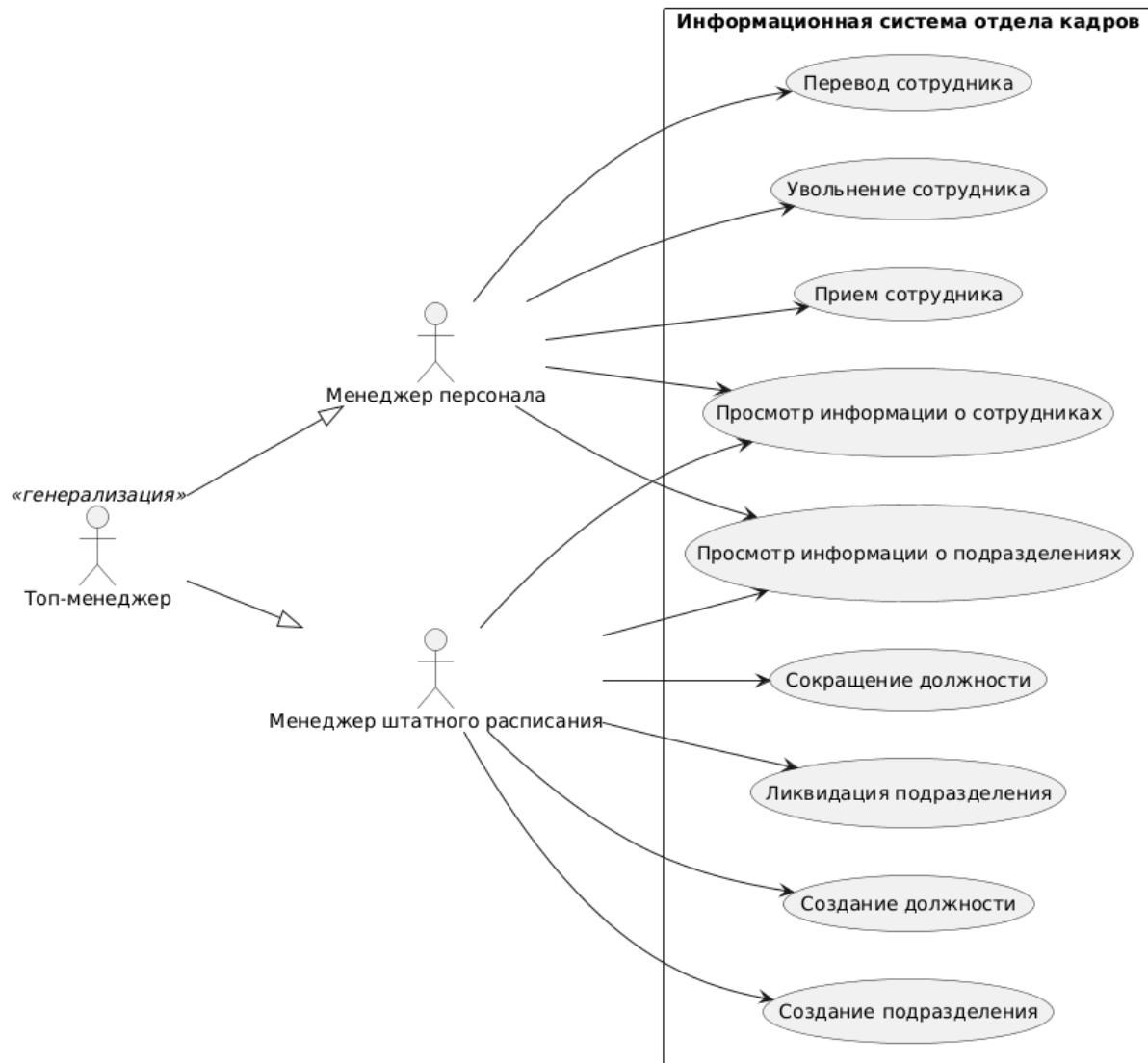
- Увольнение сотрудника (uc4)
- Перевод сотрудника (uc5)

Для Менеджера штатного расписания:

- Просмотр информации о сотрудниках (uc1)
- Просмотр информации о подразделениях (uc2)
- Создание подразделения (uc6)
- Создание должности (uc7)
- Ликвидация подразделения (uc8)
- Сокращение должности (uc9)

Для Топ-менеджера: Все вышеперечисленные варианты использования (с uc1 по uc9).

## Диаграмма



## Промпт 4 – ДИ

Сгенерируй PlantUML код для диаграммы вариантов использования информационной системы отдела кадров. Укажи действующие лица: менеджер персонала, менеджер штатного расписания, топ-менеджер. Отобрази границу системы и все связи действующих лиц с вариантами использования.

Варианты использования для менеджера персонала:

- просмотреть информацию о сотрудниках и подразделениях
- принять сотрудника

- перевести сотрудника
- уволить сотрудника:
  - «include» заблокировать учётную запись
  - уволить по инициативе администрации
  - «extend» выплатить компенсацию
  - уволить по собственному желанию
  - «extend» выплатить компенсацию

Варианты использования для менеджера штатного расписания:

- просмотреть информацию о сотрудниках и подразделениях
- создать подразделение
- ликвидировать подразделение
- создать должность
- сократить должность

Варианты использования для топ-менеджера те же, что у менеджера персонала и менеджера штатного расписания.

## Ответ

Plantuml

```
@startuml
left to right direction
skinparam packageStyle rectangle

actor "Менеджер персонала" as hr_manager
actor "Менеджер штатного расписания" as staff_manager
actor "Топ-менеджер" as top_manager

rectangle "Информационная система отдела кадров" {
    (Просмотреть информацию о сотрудниках\ни подразделениях) as view_info

    (Принять сотрудника) as hire_employee
    (Перевести сотрудника) as transfer_employee
    (Уволить сотрудника) as dismiss_employee

    (Создать подразделение) as create_department
}
```

```

(Ликвидировать подразделение) as close_department
(Создать должность) as create_position
(Сократить должность) as reduce_position

(Заблокировать учётную запись) as block_account
(Уволить по инициативе администрации) as dismiss_by_admin
(Уволить по собственному желанию) as dismiss_by_employee
(Выплатить компенсацию) as pay_compensation
}

' Связи менеджера персонала
hr_manager --> view_info
hr_manager --> hire_employee
hr_manager --> transfer_employee
hr_manager --> dismiss_employee

' Связи менеджера штатного расписания
staff_manager --> view_info
staff_manager --> create_department
staff_manager --> close_department
staff_manager --> create_position
staff_manager --> reduce_position

' Связи топ-менеджера (все варианты использования)
top_manager --> view_info
top_manager --> hire_employee
top_manager --> transfer_employee
top_manager --> dismiss_employee
top_manager --> create_department
top_manager --> close_department
top_manager --> create_position
top_manager --> reduce_position

' Включения и расширения для увольнения
dismiss_employee .> block_account : include
dismiss_employee <|-- dismiss_by_admin
dismiss_employee <|-- dismiss_by_employee
dismiss_by_admin .> pay_compensation : extend
dismiss_by_employee .> pay_compensation : extend

@enduml

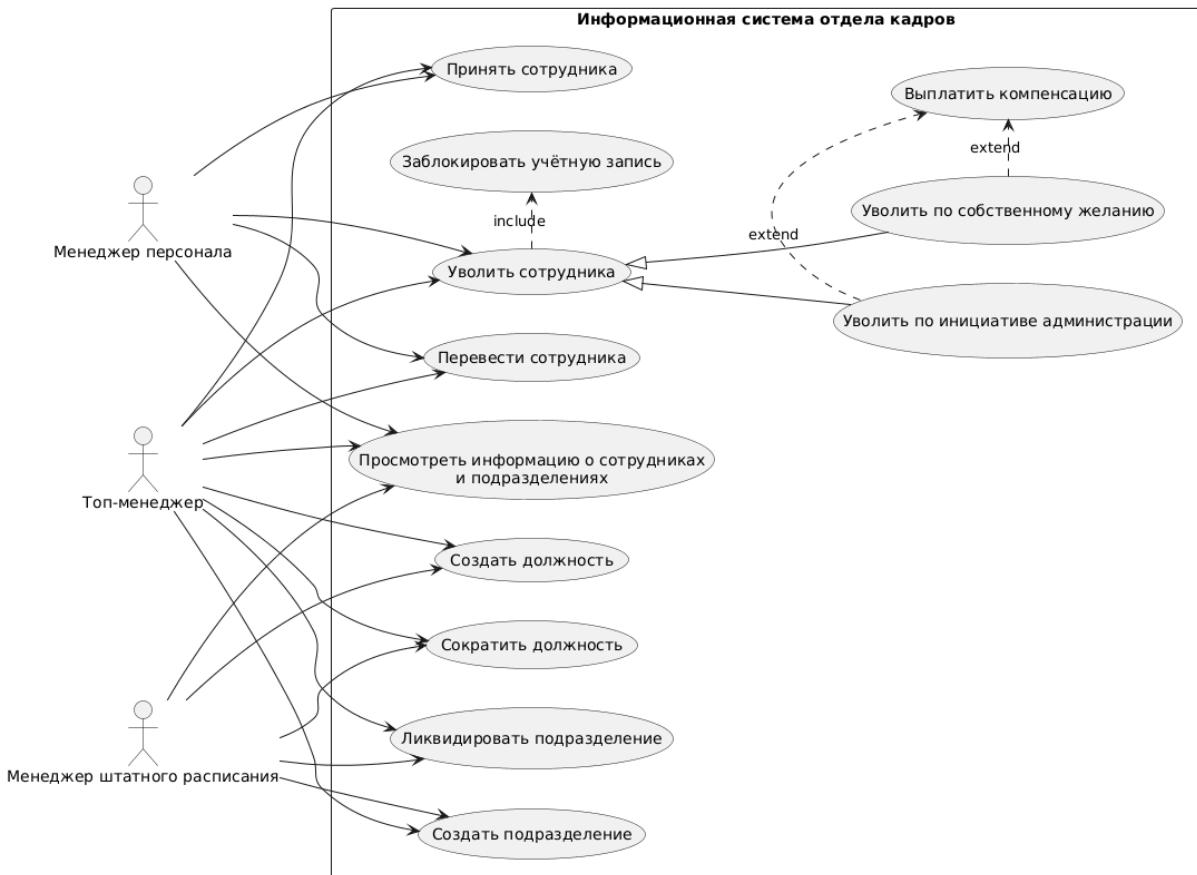
```

Этот код PlantUML создаст диаграмму вариантов использования, которая включает:

1. Трех действующих лиц:
  - Менеджер персонала
  - Менеджер штатного расписания
  - Топ-менеджер
2. Границу системы - прямоугольник, охватывающий все варианты использования.
3. Все связи между действующими лицами и вариантами использования:
  - Менеджер персонала связан с 4 основными вариантами использования
  - Менеджер штатного расписания связан с 5 вариантами использования
  - Топ-менеджер связан со всеми 9 вариантами использования
4. Отношения между вариантами использования:
  - include между "Уволить сотрудника" и "Заблокировать учётную запись"
  - Наследование между "Уволить сотрудника" и двумя специализированными вариантами
  - extend для выплаты компенсации в обоих случаях увольнения

Диаграмма будет отображена в направлении слева направо для лучшей читаемости.

## Диаграмма



## Промпт 1 – ДК

Сгенерируй PlantUML код для диаграммы классов информационной системы отдела кадров. Укажи основные классы и связи.

## Ответ

plantuml

```
@startuml
' Диаграмма классов информационной системы отдела кадров
title Диаграмма классов HR-системы
```

```
class Сотрудник {
    - id: UUID
    - табельныйНомер: String
    - фамилия: String
    - имя: String
    - отчество: String
    - датаРождения: Date
```

```
- пол: Enum  
- контактныйТелефон: String  
- email: String  
- ИНН: String  
- СНИЛС: String  
+ рассчитатьСтаж(): Integer  
+ получитьТекущуюДолжность(): Должность  
}
```

```
class Должность {  
    - id: UUID  
    - наименование: String  
    - кодДолжности: String  
    - категория: String  
    - базовыйОклад: Decimal  
    - описание: String  
    + являетсяРуководящей(): Boolean  
}
```

```
class Подразделение {  
    - id: UUID  
    - наименование: String  
    - кодПодразделения: String  
    - родительскоеПодразделение: Подразделение  
    - руководитель: Сотрудник  
    + получитьИерархию(): List<Подразделение>  
}
```

```
class ТрудовойДоговор {  
    - id: UUID  
    - номер: String  
    - датаЗаключения: Date  
    - датаНачала: Date  
    - датаОкончания: Date  
    - типДоговора: Enum  
    - условия: String  
    - статус: Enum  
    + расторгнуть(): void  
    + продлить(): void  
}
```

```
class КадровоеПеремещение {
```

```
- id: UUID
- дата: Date
- типПеремещения: Enum
- основание: String
- приказНомер: String
- приказДата: Date
+ создатьПриказ(): Документ
}
```

```
class ГрафикРаботы {
- id: UUID
- типГрафика: Enum
- нормЧасовВМесяц: Integer
- расписание: String
+ рассчитатьОтклонение(): Decimal
}
```

```
class Отпуск {
- id: UUID
- датаНачала: Date
- датаОкончания: Date
- типОтпуска: Enum
- количествоДней: Integer
- статус: Enum
- приказНомер: String
+ рассчитатьДни(): Integer
}
```

```
class БольничныйЛист {
- id: UUID
- номер: String
- датаНачала: Date
- датаОкончания: Date
- диагноз: String
- статус: Enum
+ рассчитатьПособие(): Decimal
}
```

```
class Награждение {
- id: UUID
- дата: Date
- типНаграды: Enum
```

```
- описание: String  
- сумма: Decimal  
+ выдатьПриказ(): Документ  
}
```

```
class ДисциплинарноеВзыскание {  
    - id: UUID  
    - дата: Date  
    - типВзыскания: Enum  
    - основание: String  
    - срокДействия: Date  
    + снятьВзыскание(): void  
}
```

```
class Документ {  
    - id: UUID  
    - типДокумента: Enum  
    - название: String  
    - содержание: Blob  
    - датаСоздания: Date  
    - подпись: String  
    + подписать(): void  
    + отозвать(): void  
}
```

```
class ПользовательСистемы {  
    - id: UUID  
    - логин: String  
    - хэшПароля: String  
    - роль: Enum  
    - активен: Boolean  
    + аутентифицировать(): Boolean  
    + авторизовать(): Boolean  
}
```

#### ' Основные связи

Сотрудник "1" -- "1..\*" ТрудовойДоговор : имеет

Сотрудник "1" -- "0..\*" КадровоеПеремещение : проходит

Сотрудник "1" -- "1" Подразделение : работает в

Сотрудник "1" -- "1" Должность : занимает

Сотрудник "1" -- "1" ГрафикРаботы : следует

Сотрудник "1" -- "0..\*" Отпуск : использует

Сотрудник "1" -- "0..\*" БольничныйЛист : оформляет  
Сотрудник "1" -- "0..\*" Награждение : получает  
Сотрудник "1" -- "0..\*" ДисциплинарноеВзыскание : имеет  
Сотрудник "1" -- "1" ПользовательСистемы : представляет

Подразделение "1" -- "0..1" Подразделение : включает >  
Должность "1" -- "0..\*" Сотрудник : занимается

КадровоеПеремещение "1" -- "1..\*" Документ : оформляет  
Отпуск "1" -- "1" Документ : подтверждается  
БольничныйЛист "1" -- "1" Документ : прилагается

' Интерфейсы и перечисления  
enum ТипДоговора {  
    бессрочный  
    срочный  
    по\_совместительству  
}

enum ТипПеремещения {  
    прием  
    перевод  
    увольнение  
    повышение  
    понижение  
}

enum СтатусДоговора {  
    активен  
    расторгнут  
    приостановлен  
}

enum РольПользователя {  
    администратор  
    hr\_специалист  
    руководитель  
    сотрудник  
}

@enduml

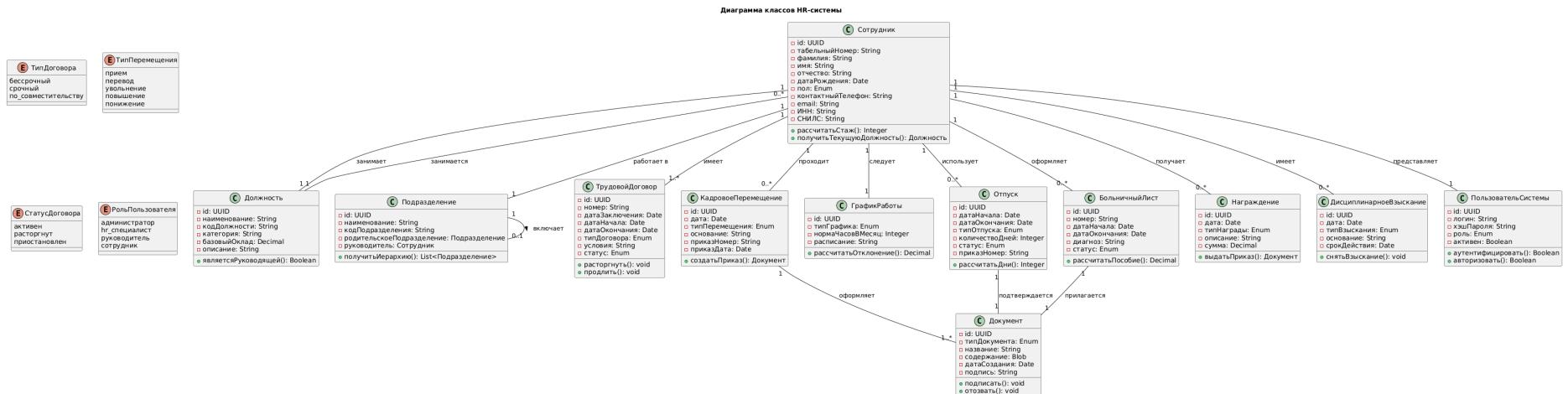
Пояснения:

1. Сотрудник - центральный класс системы, содержащий личные данные
2. Должность и Подразделение - справочники организационной структуры
3. ТрудовойДоговор - юридические основания работы
4. КадровоеПеремещение - история карьерного роста (прием, переводы, увольнения)
5. Отпуск и БольничныйЛист - модули учета отсутствий
6. Награждение и ДисциплинарноеВзыскание - учет поощрений и взысканий
7. Документ - абстракция для кадровых приказов и документов
8. ПользовательСистемы - управление доступом к системе

Основные связи:

- Композиция между сотрудником и его документами
- Агрегация между организационными единицами
- Наследование типов через перечисления
- Биективные связи для основных сущностей

**Диаграмма**



## Промпт 2 – ДК

Сгенерируй PlantUML код для диаграммы классов информационной системы отдела кадров. Классы: предприятие, подразделение, должность, проект, сотрудник. Укажи атрибуты с типами и основные связи.

## Ответ

Plantuml

```
@startuml
' Диаграмма классов для информационной системы отдела кадров
title Диаграмма классов: Информационная система отдела кадров

class Предприятие {
    -название: String
    -адрес: String
    -ИНН: String
    -телефон: String
    +добавитьПодразделение()
    +удалитьПодразделение()
}

class Подразделение {
    -название: String
    -кодПодразделения: String
    -датаСоздания: Date
    -типПодразделения: String
    +изменитьНазвание()
    +переместитьСотрудника()
}

class Должность {
    -название: String
    -оклад: BigDecimal
    -требования: String
    -кодДолжности: String
    +изменитьОклад()
    +обновитьТребования()
}

class Проект {
```

```
-название: String  
-кодПроекта: String  
-бюджет: BigDecimal  
-срокНачала: Date  
-срокОкончания: Date  
-статус: String  
+добавитьУчастника()  
+изменитьСтатус()  
}
```

```
class Сотрудник {  
    -табельныйНомер: String  
    -ФИО: String  
    -датаРождения: Date  
    -пол: String  
    -паспортныеДанные: String  
    -датаПриема: Date  
    -контактныйТелефон: String  
    -электроннаяПочта: String  
    +рассчитатьЗарплату()  
    +оформитьОтпуск()  
    +перевестиВДругоеПодразделение()  
}
```

' Основные связи между классами

Предприятие "1" \*-- "0..\*" Подразделение : содержит

Подразделение "1" \*-- "1..\*" Должность : включает

Подразделение "1" -- "0..\*" Сотрудник : штат

Должность "1" -- "1..\*" Сотрудник : занимает

Сотрудник "0..\*" -- "0..\*" Проект : участвует

Подразделение "1" -- "1" Сотрудник : руководитель

Проект "1" -- "1" Подразделение : ответственное

Проект "1" -- "1" Сотрудник : руководительПроекта

' Дополнительные заметки

note top of Предприятие : Основная организационная единица

note right of Подразделение : Структурная единица предприятия\n(отдел, управление, департамент)

note left of Должность : Штатная позиция в подразделении

note right of Проект : Временная деятельность с целями\ни бюджетом

note left of Сотрудник : Физическое лицо, работающее\на предприятии

@enduml

Пояснения к диаграмме:

1. Агрегация (\*--):

- Предприятие содержит подразделения (подразделения могут существовать отдельно)
- Подразделение включает должности (должность не существует без подразделения)

2. Ассоциации (--):

- Сотрудник работает в подразделении
- Сотрудник занимает должность
- Сотрудники участвуют в проектах (многие-ко-многим)
- Подразделение имеет руководителя (сотрудника)
- Проект имеет ответственное подразделение и руководителя проекта

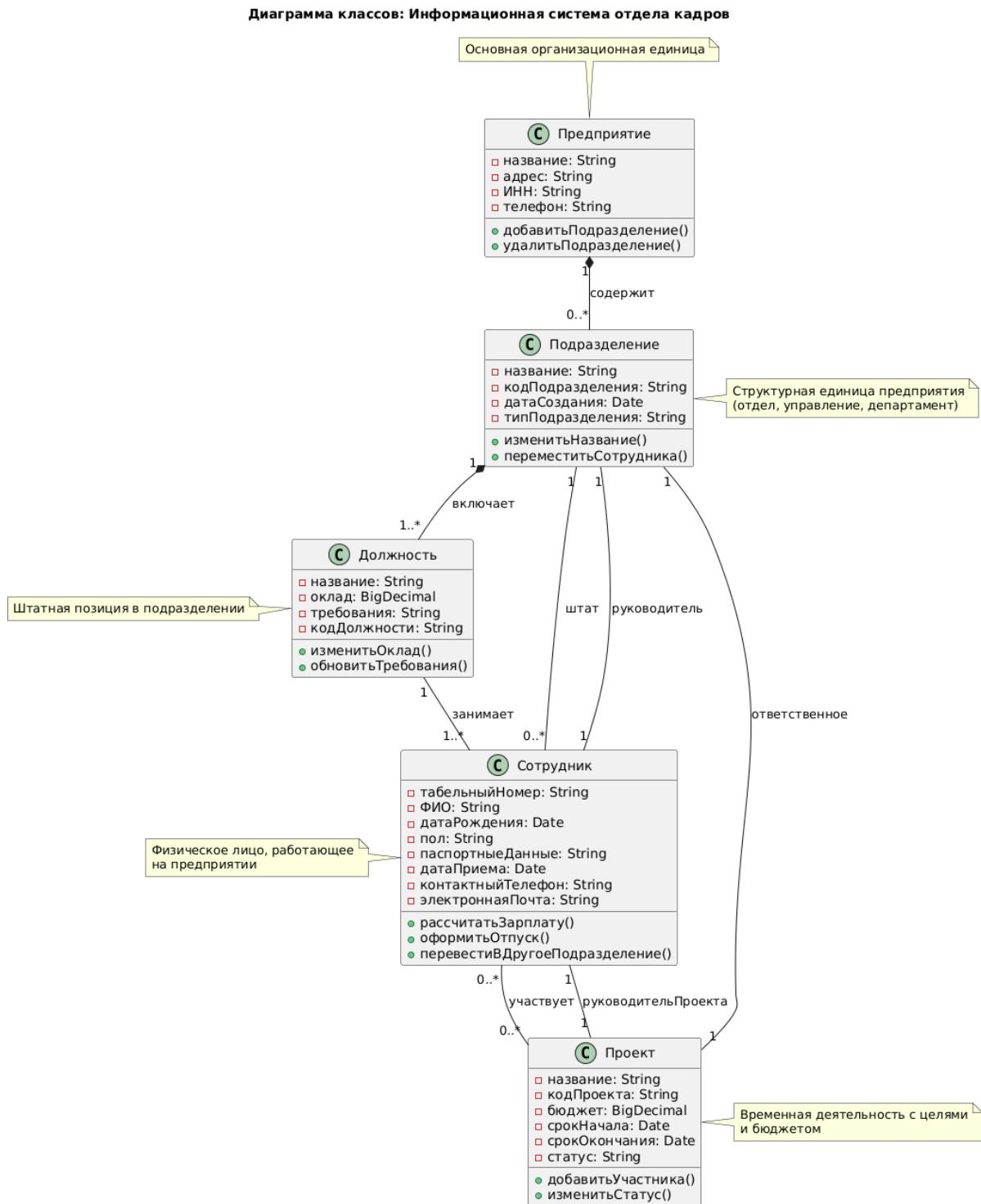
3. Атрибуты указаны с типами данных и видимостью:

- - приватные (private)
- Типы: String, Date, BigDecimal и др.

4. Основные методы для каждого класса.

Диаграмма может быть расширена дополнительными классами (Награда, Отпуск, Больничный и т.д.) и уточнением типов связей.

## Диаграмма



## Промпт 3 – ДК

Сгенерируй PlantUML код для диаграммы классов системы управления отделом кадров. Укажи классы: Предприятие, Подразделение,

Должность, Проект, Сотрудник. Покажи связи между ними (ассоциация, агрегация, композиция, ассоциация-класс).

## Ответ

### Plantuml

```
@startuml
' Класс Предприятие
class Предприятие {
    - название: String
    - ИНН: String
    - адрес: String
    + нанятьСотрудника()
    + создатьПодразделение()
    + запуститьПроект()
}

' Класс Подразделение
class Подразделение {
    - название: String
    - код: String
    - руководитель: Сотрудник
    + добавитьСотрудника()
    + назначитьРуководителя()
}

' Класс Должность
class Должность {
    - название: String
    - оклад: BigDecimal
    - требования: String
    + рассчитатьЗарплату()
}

' Класс Проект
class Проект {
    - название: String
    - бюджет: BigDecimal
    - срокИсполнения: Date
    - статус: String
    + запуститьПроект()
```

```

+ завершитьПроект()
+ добавитьУчастника()
}

' Класс Сотрудник
class Сотрудник {
    - ФИО: String
    - паспортные данные: String
    - датаПриема: Date
    - контактные данные: String
    + работатьНадПроектом()
    + взятьОтпуск()
    + получитьЗарплату()
}

' Ассоциация-класс для связи Сотрудник-Проект
class УчастиеВПроекте {
    - датаНачала: Date
    - датаОкончания: Date
    - роль: String
    - загрузка: int
    + рассчитатьЗарплату()
}

' Связи между классами

' Агрегация: Предприятие содержит Подразделения
Предприятие "1" о-- "0..*" Подразделение : содержит >

' Композиция: Подразделение состоит из Должностей
Подразделение "1" *-- "1..*" Должность : включает >

' Ассоциация: Подразделение управляет Проектами
Подразделение "1" -- "0..*" Проект : управляет >

' Ассоциация: Сотрудник занимает Должность
Сотрудник "1" -- "1" Должность : занимает >

' Ассоциация: Сотрудник работает в Подразделении
Сотрудник "1" -- "1" Подразделение : работает в >

' Ассоциация-класс: Сотрудник участвует в Проекте через УчастиеВПроекте

```

Сотрудник "1" -- "0..\*" УчастиеВПроекте

УчастиеВПроекте "0..\*" -- "1" Проект

' Ассоциация: Руководитель Подразделения (специальная связь)

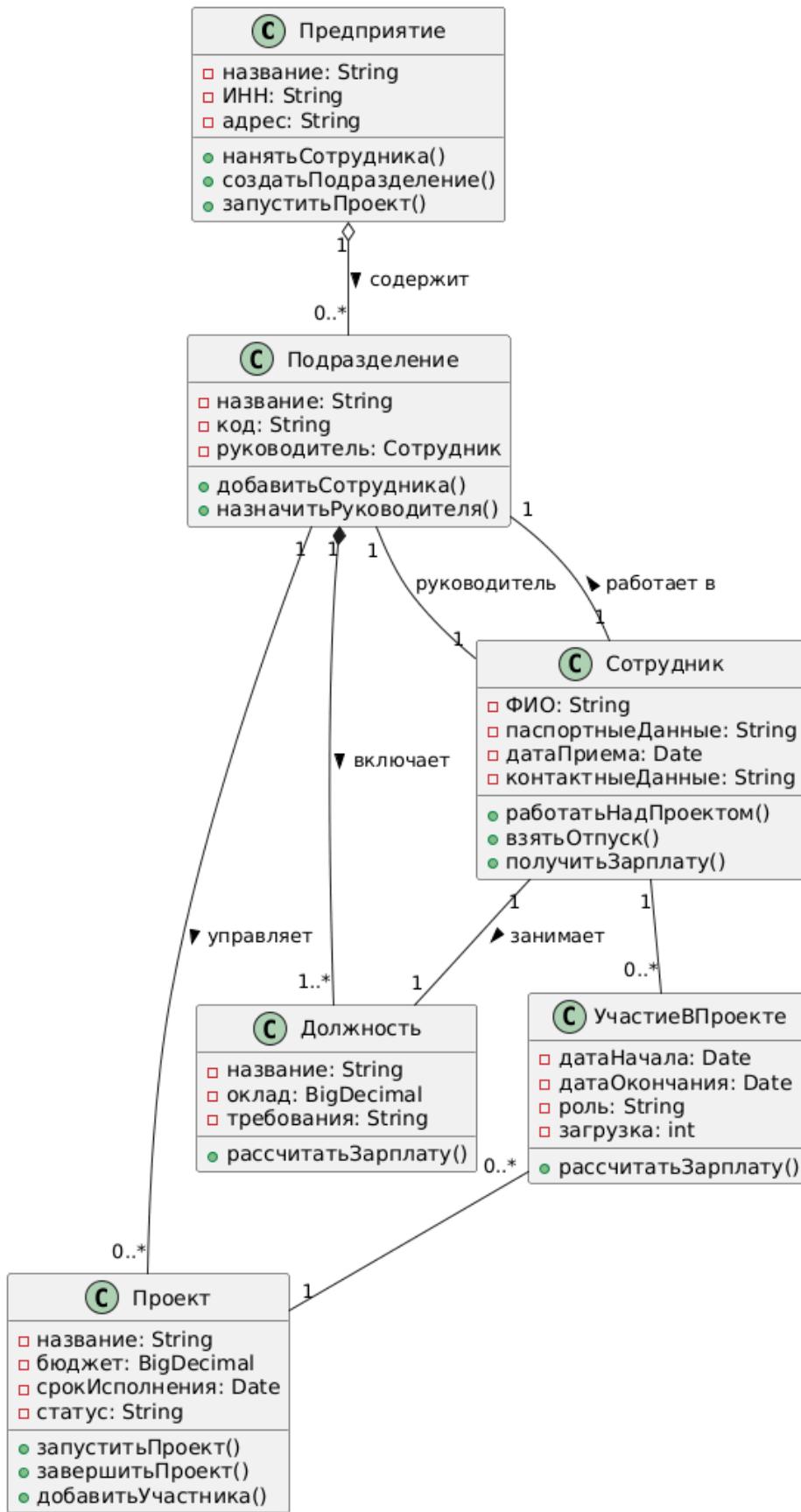
Подразделение "1" -- "1" Сотрудник : руководитель

@enduml

### **Пояснения к связям:**

1. **Агрегация** (ромб пустой): Предприятие содержит Подразделения, но подразделения могут существовать отдельно (например, при реорганизации).
  2. **Композиция** (ромб заполненный): Подразделение жестко включает Должности. Должность не может существовать без подразделения.
  3. **Ассоциация** (простая линия):
    - Подразделение управляет Проектами
    - Сотрудник занимает конкретную Должность
    - Сотрудник работает в конкретном Подразделении
    - Специальная связь "руководитель" между Подразделением и Сотрудником
  4. **Ассоциация-класс**: Связь "многие-ко-многим" между Сотрудником и Проектом реализована через класс УчастиеВПроекте, который хранит дополнительную информацию (роль, даты, загрузку).
- Дополнительные возможности для расширения:**
- Добавить класс НачислениеЗарплаты как ассоциацию-класс между Сотрудник и Должность
  - Добавить класс ИсторияКарьера для отслеживания перемещений сотрудника
  - Добавить класс Резюме или Документы в состав сотрудника (композиция)

## Диаграмма



## Промпт 4 – ДК

Сгенерируй PlantUML код для диаграммы классов системы управления отделом кадров. Укажи классы: Предприятие, Подразделение, Должность, Проект, Сотрудник. Покажи связи между ними:

Предприятие имеет подразделения (композиция).

Подразделений может быть много (рефлексивная ассоциация).

Должность часть подразделения (агрегация).

Сотрудник работает на должности и в проекте (многополюсная ассоциация).

## Ответ

plantuml

```
@startuml
```

```
class Предприятие {  
    -название: String  
    -адрес: String  
    +добавитьПодразделение()  
    +удалитьПодразделение()  
}
```

```
class Подразделение {  
    -название: String  
    -код: String  
    +изменитьНазвание()  
    +переместитьВ()  
}
```

```
class Должность {  
    -название: String  
    -оклад: Decimal  
    -требования: String  
    +изменитьОклад()  
}
```

```
class Проект {  
    -название: String
```

```

-бюджет: Decimal
-срок: Date
+завершитьПроект()
}

class Сотрудник {
    -ФИО: String
    -паспортные данные: String
    -датаПриема: Date
    +назначитьДолжность()
    +добавитьВПроект()
    +рассчитатьЗарплату()
}

```

' Композиция: Предприятие → Подразделение  
Предприятие "1" \*-- "1..\*" Подразделение : содержит

' Рефлексивная ассоциация для иерархии подразделений  
Подразделение "1" --o "0..\*" Подразделение : входит в

' Агрегация: Подразделение → Должность  
Подразделение "1" o-- "1..\*" Должность : включает

' Многополюсная ассоциация между Сотрудник, Должность и Проект  
Сотрудник "1" -- "1" Должность : занимает  
Сотрудник "1..\*" -- "0..\*" Проект : участвует в

@enduml

### **Пояснения к связям:**

#### **1. Композиция (Предприятие ↔ Подразделение):**

- \*-- (сплошной ромб) означает, что подразделения не могут существовать без предприятия
- Множественность 1..\* показывает, что предприятие имеет минимум одно подразделение

#### **2. Рефлексивная ассоциация (Подразделение ↔ Подразделение):**

- --o (пустой ромб) показывает, что подразделение может состоять из других подразделений
- Используется для создания иерархической структуры отделов

### **3. Агрегация (Подразделение ↔ Должность):**

- o-- (пустой ромб) означает, что должности могут существовать независимо от подразделения
- Каждое подразделение содержит одну или несколько должностей

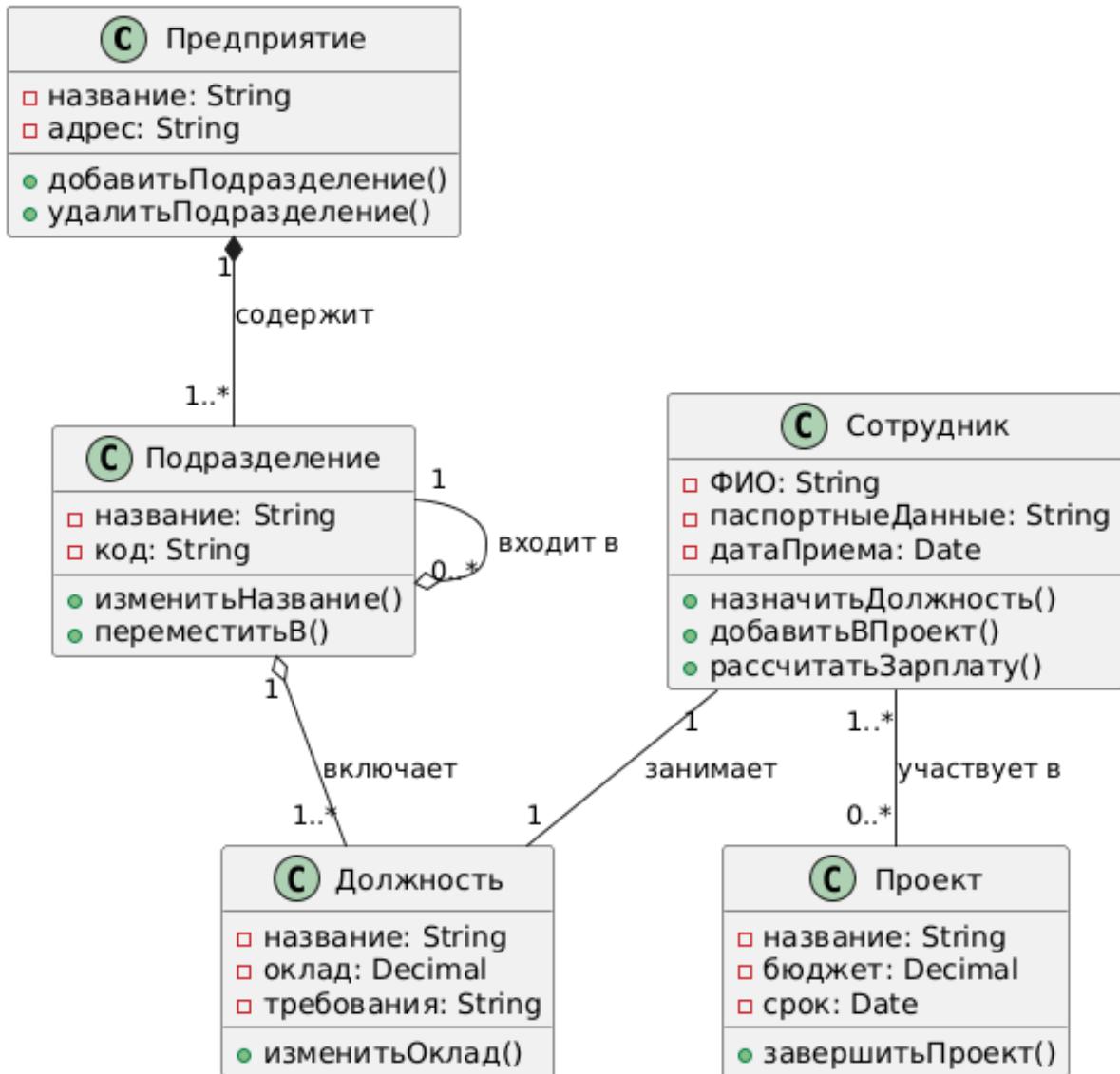
### **4. Ассоциации для Сотрудника:**

- Связь с Должностью: каждый сотрудник занимает одну должность
- Связь с Проектом: сотрудник может участвовать в 0 или нескольких проектах, проект может иметь 0 или нескольких сотрудников

#### **Дополнительные возможности для расширения диаграммы:**

- Добавить классы Контракт, Отпуск, Навыки
- Включить интерфейсы для расчета зарплаты
- Добавить перечисления для типов должностей или статусов проекта.

## Диаграмма



### 3. Сравнительный анализ полученных результатов

Критерии оценки:

1. Корректность синтаксиса PlantUML
2. Соответствие стандартам UML
3. Полнота отображения функциональных требований
4. Качество визуализации
5. Логическая целостность диаграмм
6. Обоснованность выбора предметной области

Диаграмма	К 1	К 2	К 3	К 4	К 5	К 6	Средний балл
ДИ1	8	7	9	7	8	8	<b>7.8</b>
ДИ2	9	8	8	8	7	9	<b>8.2</b>
ДИ3	7	6	7	6	7	8	<b>6.8</b>
ДИ4	10	9	9	9	9	9	<b>9.2</b>
ДК1	8	7	9	7	8	9	<b>8.0</b>
ДК2	9	8	8	8	8	9	<b>8.3</b>
ДК3	7	6	7	6	7	7	<b>6.7</b>
ДК4	10	9	10	9	9	10	<b>9.5</b>

Лучшие диаграммы:

Лучшая ДИ (диаграмма вариантов использования): ДИ4 – обладает высокой степенью детализации, корректно отображает все роли, включает отношения include и extend, а также границу системы. Диаграмма логически целостна и полностью соответствует функциональным требованиям.

Лучшая ДК (диаграмма классов): ДК4 – демонстрирует исключительную полноту, включает все запрошенные классы, правильно отображает типы связей (композиция, агрегация, рефлексивная ассоциация), а также содержит атрибуты и методы. Полностью соответствует предметной области и стандартам UML.

Обоснование оценок:

ДИ1 – Хорошая полнота функционала, множество акторов и вариантов использования, но перегружена связями и содержит нестандартные стереотипы.

ДИ2 – Сбалансированная диаграмма с четкой структурой, хорошим охватом функционала и корректными связями, но не включает все отношения между вариантами использования.

ДИ3 – Упрощенная диаграмма с минимальным набором связей, недостаточно детализирована для полного отражения функциональности.

ДИ4 – Наиболее полная и логичная диаграмма, включает все роли, специализированные сценарии увольнения, отношения include/extend и границу системы. Визуально понятна и соответствует стандартам UML.

ДК1 – Полноценная диаграмма с хорошим охватом сущностей, но недостаточно структурирована, частично перегружена атрибутами.

ДК2 – Хорошо организованная диаграмма с четкой иерархией классов и адекватными связями, но отсутствуют некоторые типы отношений (например, рефлексивная ассоциация).

ДК3 – Использование кириллицы в коде снижает читаемость, недостаточно атрибутов и методов, упрощенные связи.

ДК4 – Идеально детализированная диаграмма с полным набором атрибутов, методов, корректными типами связей (композиция, агрегация, рефлексивная ассоциация), полностью соответствующая предметной области и стандартам UML.

Вывод:

Наилучшие результаты действительно получены в промптах 4 ДИ и 4 ДК, что подтверждается их высокой оценкой по всем критериям, логической целостностью и полнотой отображения функциональных требований системы отдела кадров.

## 4. Итоговые промпты

Для диаграммы вариантов использования лучший промпт Промпт 4 –  
ДИ:

«Сгенерируй PlantUML код для диаграммы вариантов использования информационной системы отдела кадров. Укажи действующие лица: менеджер персонала, менеджер штатного расписания, топ-менеджер. Отобрази границу системы и все связи действующих лиц с вариантами использования.

Варианты использования для менеджера персонала:

- просмотреть информацию о сотрудниках и подразделениях
- принять сотрудника
- перевести сотрудника
- уволить сотрудника:
  - «include» заблокировать учётную запись
  - уволить по инициативе администрации
  - «extend» выплатить компенсацию
  - уволить по собственному желанию
  - «extend» выплатить компенсацию

Варианты использования для менеджера штатного расписания:

- просмотреть информацию о сотрудниках и подразделениях
- создать подразделение
- ликвидировать подразделение
- создать должность
- сократить должность

Варианты использования для топ-менеджера те же, что у менеджера персонала и менеджера штатного расписания.»

«Для диаграммы классов лучший промпт Промпт 4 – ДК:

Сгенерируй PlantUML код для диаграммы классов системы управления отделом кадров. Укажи классы: Предприятие, Подразделение, Должность, Проект, Сотрудник. Покажи связи между ними:

Предприятие имеет подразделения (композиция).

Подразделений может быть много (рефлексивная ассоциация).

Должность часть подразделения (агрегация).

Сотрудник работает на должности и в проекте (многополюсная ассоциация).»