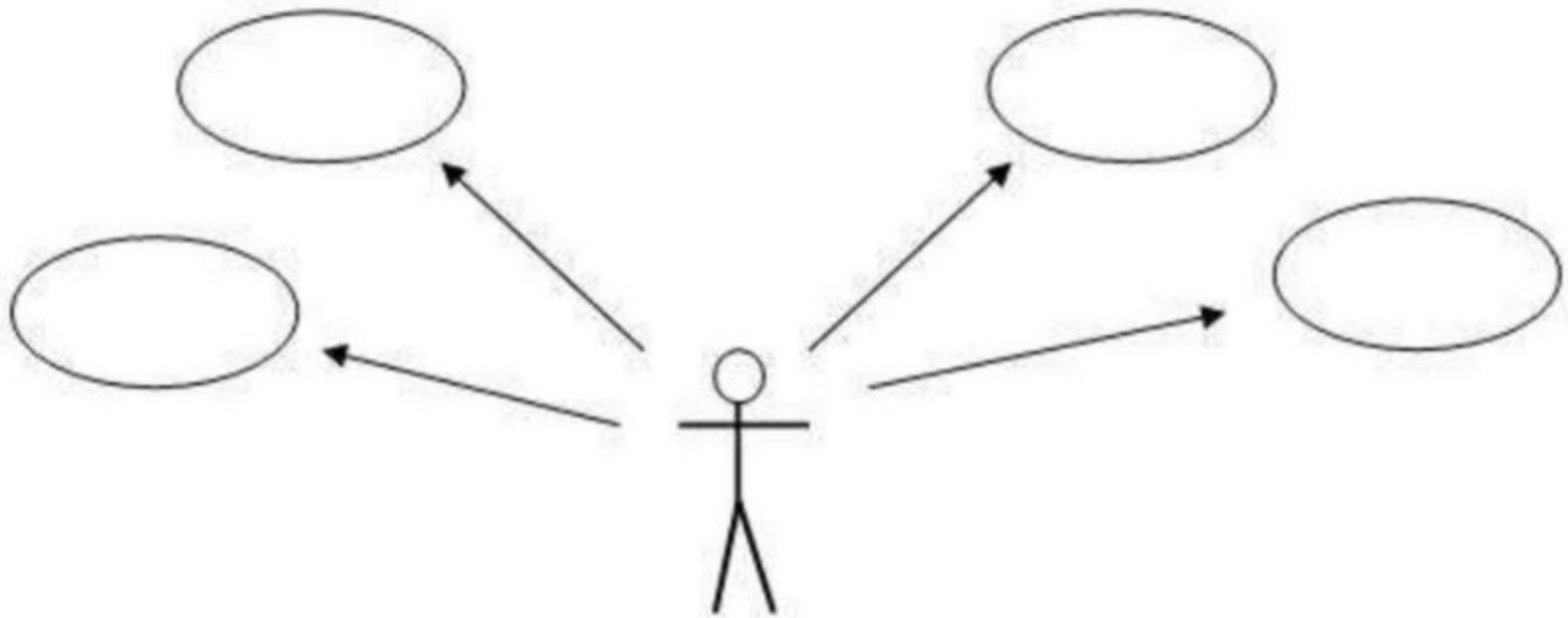
Введение в UseCase

Диаграмма прецедентов



UseCase-диаграммы

1. Диаграммы вариантов использования (Use Case)

- Для пошагового построения модели от наиболее общей и абстрактной концептуальной модели исходной системы к логической, а затем и к физической модели, после определения объектов и классов, строится модель в форме так называемой диаграммы вариантов использования (use case diagram), которая описывает функциональное назначение системы или, другими словами, то, что система будет делать в процессе своего функционирования.

- Диаграмма вариантов использования является исходным концептуальным представлением или концептуальной моделью системы в процессе ее проектирования и разработки.

Назначение диаграмм Варианты использования

Разработка диаграммы вариантов использования преследует цели:

- Определить общие границы и контекст моделируемой предметной области на начальных этапах проектирования системы.
- Сформулировать общие требования к функциональному поведению проектируемой системы.
- Разработать исходную концептуальную модель системы для ее последующей детализации в форме логических и физических моделей.
- Подготовить исходную документацию для взаимодействия разработчиков системы с ее заказчиками и пользователями.

Суть диаграммы **use case**

- Проектируемая система представляется в виде множества сущностей или актеров, взаимодействующих с системой с помощью так называемых вариантов использования.
- Актером (actor) или действующим лицом называется любая сущность, взаимодействующая с системой извне. Это может быть человек, техническое устройство, программа или любая другая система, которая может служить источником воздействия на моделируемую систему так, как определит сам разработчик.
- В свою очередь, вариант использования (use case) служит для описания сервисов, которые система предоставляет актеру.
- Другими словами, каждый вариант использования определяет некоторый набор действий, совершаемый системой при диалоге с актером. При этом ничего не говорится о том, каким образом будет реализовано взаимодействие актеров с системой.

Диаграмма вариантов использования представляет собой граф специального вида, который является графической нотацией для представления конкретных вариантов использования, актеров, возможно некоторых интерфейсов, и отношений между этими элементами.

Базовые элементы:

вариант использования и актер.

Какие типы документов используются в схеме

1. Техническое задание.
2. Частное техническое задание (опционально).
3. Сценарий использования (Use Case).
4. Сценарий тестирования (Test Case).
5. Отчет об ошибке (Bug Report).
6. Руководство пользователя.
7. Руководство администратора (опционально).

Какие типы документов используются в схеме

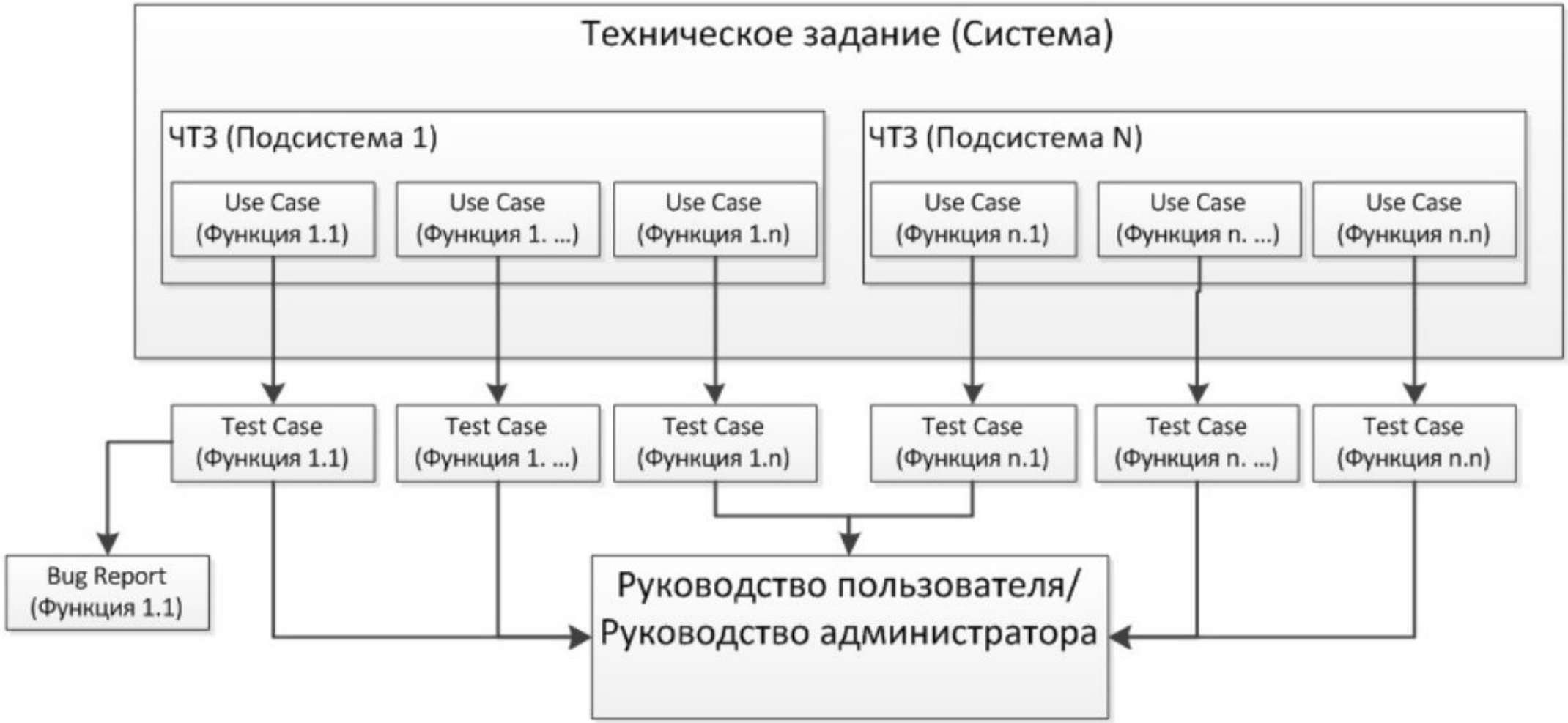


Диаграмма строится базе компонентов следующих типов :

- – ***действующие лица*** или ***актеры*** *(actors)* .
- – ***варианты использования*** *(use case).*
- – ***связи*** или ***отношения*** *(relationships)* .
- – ***интерфейсы*** *(interface).*
- – ***примечания*** (notes).

Стандартные элементы

Отдельный ***вариант использования*** обозначается на диаграмме эллипсом, внутри которого или рядом с ним содержится его краткое название или имя в форме глагола с пояснительными словами.

Проверить состояние текущего счета

- Цель варианта использования заключается в том, чтобы определить законченный аспект или фрагмент поведения некоторой сущности без раскрытия внутренней структуры этой сущности.
- Каждый вариант использования соответствует отдельному сервису, который предоставляет моделируемую сущность или систему по запросу пользователя (актера), т. е. определяет способ применения этой сущности.
- Сервис, который инициализируется по запросу пользователя, представляет собой законченную последовательность действий.

- Варианты использования описывают не только взаимодействия между пользователями и сущностью, но также реакции сущности на получение отдельных сообщений от пользователей и восприятие этих сообщений за пределами сущности.
- Варианты использования могут включать в себя описание особенностей способов реализации сервиса и различных исключительных ситуаций, таких как корректная обработка ошибок системы.
- Множество вариантов использования в целом должно определять все возможные стороны ожидаемого поведения системы.
- Примерами вариантов использования могут являться следующие действия: проверка состояния текущего счета клиента, оформление заказа на покупку товара, получение дополнительной информации о кредитоспособности клиента, отображение графической формы на экране монитора и другие действия.

Актеры

Актер представляет собой любую ***внешнюю по отношению к моделируемой системе сущность***, которая взаимодействует с системой и использует ее функциональные возможности для достижения определенных целей или решения частных задач.

Каждый актер может рассматриваться как некая отдельная роль относительно конкретного варианта использования.

В некоторых случаях актер может обозначаться в виде прямоугольника класса с ключевым словом "актер" и обычными составляющими элементами класса. Имена актеров должны записываться заглавными буквами и следовать рекомендациям использования имен для типов и классов модели.

Примерами актеров могут быть: клиент банка, банковский служащий, продавец магазина, менеджер отдела продаж, пассажир авиарейса, водитель автомобиля, администратор гостиницы, сотовый телефон и другие сущности, имеющие отношение к концептуальной модели соответствующей предметной области.

- В качестве актеров могут выступать другие системы, подсистемы проектируемой системы или отдельные классы.
- Каждый актер определяет некоторое согласованное множество ролей, в которых могут выступать пользователи данной системы в процессе взаимодействия с ней. В каждый момент времени с системой взаимодействует вполне определенный пользователь, при этом он играет или выступает в одной из таких ролей.
- Наиболее наглядный пример актера — конкретный пользователь системы со своими собственными параметрами аутентификации.

Интерфейсы

Интерфейс (interface) служит для спецификации параметров модели, которые видимы извне без указания их внутренней структуры.

В языке UML интерфейс является классификатором и характеризует только ограниченную часть поведения моделируемой сущности. Применительно к диаграммам вариантов использования, интерфейсы определяют совокупность операций, которые обеспечивают необходимый набор сервисов или функциональности для актеров.

Примечания

Примечания (notes) в языке UML предназначены для включения в модель произвольной текстовой информации, имеющей непосредственное отношение к контексту разрабатываемого проекта. В качестве такой информации могут быть комментарии разработчика (например, дата и версия разработки диаграммы или ее отдельных компонентов), ограничения (например, на значения отдельных связей или экземпляры сущностей) и помеченные значения.

Применительно к диаграммам вариантов использования примечание может носить самую общую информацию, относящуюся к общему контексту системы.

Отношения на диаграмме вариантов использования

Между компонентами диаграммы вариантов использования могут существовать различные отношения, которые описывают взаимодействие экземпляров одних актеров и вариантов использования с экземплярами других актеров и вариантов. Один актер может взаимодействовать с несколькими вариантами использования. В этом случае этот актер обращается к нескольким сервисам данной системы. В свою очередь один вариант использования может взаимодействовать с несколькими актерами, предоставляя для всех них свой сервис.

Два варианта использования, определенные для одной и той же сущности, не могут взаимодействовать друг с другом, поскольку каждый из них самостоятельно описывает законченный вариант использования этой сущности.

Варианты использования всегда предусматривают некоторые сигналы или сообщения, когда взаимодействуют с актерами за пределами системы.

- В языке UML имеется несколько стандартных видов отношений между актерами и вариантами использования:
- Отношение ассоциации (association relationship)
- Отношение расширения (extend relationship)
- Отношение обобщения (generalization relationship)
- Отношение включения (include relationship)

Отношение ассоциации

Отношение ассоциации является одним из фундаментальных понятий в языке UML.

Применительно к диаграммам вариантов использования оно служит для обозначения специфической роли актера в отдельном варианте использования.

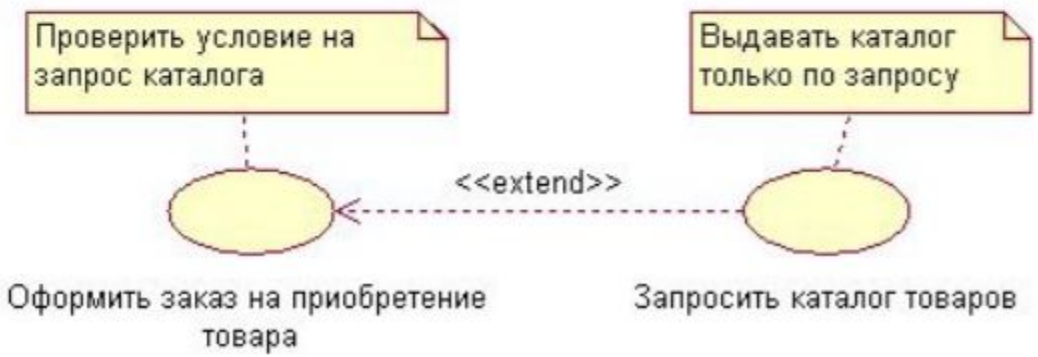
Ассоциация специфицирует семантические особенности взаимодействия актеров и вариантов использования в графической модели системы. Таким образом, это отношение устанавливает, какую конкретную роль играет актер при взаимодействии с экземпляром вари

Кратность характеризует общее количество конкретных экземпляров данного компонента, которые могут выступать в качестве элементов данной ассоциации.

Отношение расширения

Отношение расширения определяет взаимосвязь экземпляров отдельного варианта использования с более общим вариантом, свойства которого определяются на основе способа совместного объединения данных экземпляров.

Расширение является направленным и указывает, что применительно к отдельным примерам некоторого варианта использования должны быть выполнены конкретные условия, определенные для расширения данного варианта использования. Так, если имеет место отношение расширения от варианта использования А к варианту использования В, то это означает, что свойства экземпляра варианта использования В могут быть дополнены благодаря наличию свойств у расширенного варианта использования А.

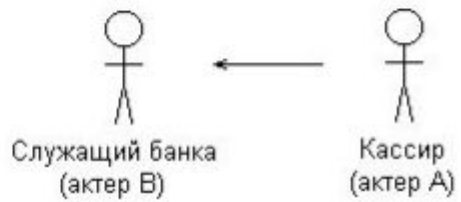


Отношение обобщения

Отношение обобщения служит для указания того факта, что некоторый вариант использования A может быть обобщен до варианта использования B. В этом случае вариант A будет являться специализацией варианта B. При этом B называется предком или родителем по отношению A, а вариант A — потомком по отношению к варианту использования B.

Потомок наследует все свойства и поведение своего родителя, а также может быть дополнен новыми свойствами и особенностями поведения.

Отношение обобщения между вариантами использования применяется в том случае, когда необходимо отметить, что дочерние варианты использования обладают всеми атрибутами и особенностями поведения родительских вариантов. При этом дочерние варианты использования участвуют во всех отношениях родительских вариантов.

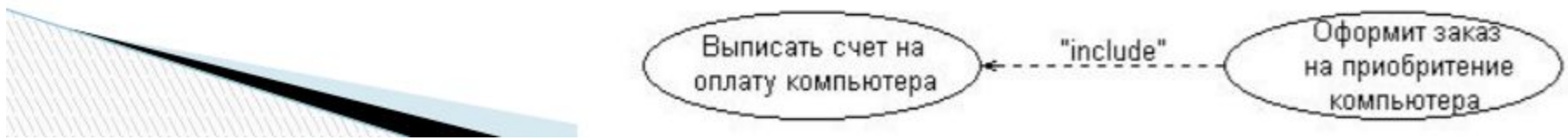


Отношение включения

Отношение включения между двумя вариантами использования указывает, что некоторое заданное поведение для одного варианта использования включается в качестве составного компонента в последовательность поведения другого варианта использования. Данное отношение является направленным бинарным отношением в том смысле, что пара экземпляров вариантов использования всегда упорядочена в отношении включения.

Семантика этого отношения определяется следующим образом. Когда экземпляр первого варианта использования в процессе своего выполнения достигает точки включения в последовательность поведения экземпляра второго варианта использования, экземпляр первого варианта использования выполняет последовательность действий, определяющую поведение экземпляра второго варианта использования, после чего продолжает выполнение действий своего поведения.

Отношение включения, направленное от варианта использования A к варианту использования B, указывает, что каждый экземпляр варианта A включает в себя функциональные свойства, заданные для варианта B. Эти свойства специализируют поведение соответствующего варианта A на диаграмме.

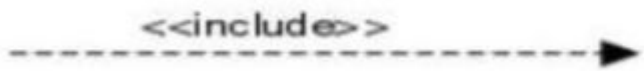


Вариант использования (use case)

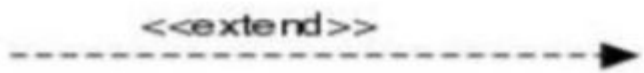
Актер (actor)

Примечание (note)

Ассоциация (association relationship)



Включение (include relationship)



Расширение (extend relationship)

Пример диаграммы вариантов использования

В качестве примера рассмотрим процесс моделирования системы продажи товаров по каталогу, которая может быть использована при создании соответствующих информационных систем.

В качестве актеров данной системы могут выступать два субъекта, один из которых является продавцом, а другой — покупателем. Каждый из этих актеров взаимодействует с рассматриваемой системой продажи товаров по каталогу и является ее пользователем, т. е. они оба обращаются к соответствующему сервису "Оформить заказ на покупку товара".

Первоначальная структура диаграммы может включать в себя только двух указанных актеров и единственный вариант использования

На следующем этапе разработки данной диаграммы вариант использования "Оформить заказ на покупку товара" может быть уточнен на основе введения в рассмотрение четырех дополнительных вариантов использования. Это следует из более детального анализа процесса продажи товаров, что позволяет выделить в качестве отдельных сервисов такие действия, как обеспечить покупателя информацией о товаре, согласовать условия оплаты товара и заказать товар со склада.

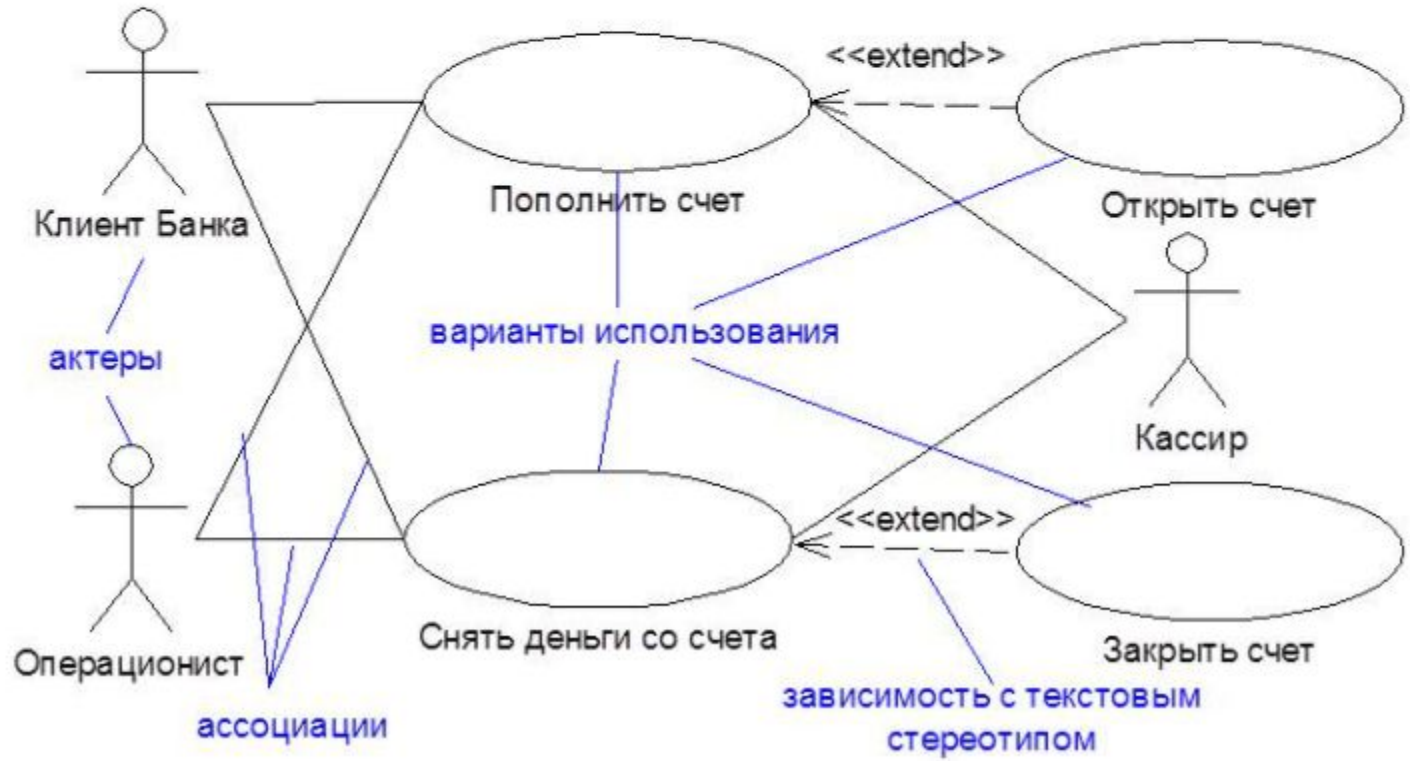
С другой стороны, продажа товаров по каталогу предполагает наличие самостоятельного информационного объекта — каталога товаров, который в некотором смысле не зависит от реализации сервиса по обслуживанию покупателей. В нашем случае, каталог товаров может запрашиваться покупателем или продавцом при необходимости выбора товара и уточнения деталей его продажи. Вполне резонно представить сервис "Запросить каталог товаров" в качестве самостоятельного варианта использования.

Приведенная диаграмма вариантов использования, в свою очередь, может быть детализирована далее с целью более глубокого уточнения предъявляемых к системе требований и конкретизации деталей ее последующей реализации. В рамках общей парадигмы ООАП подобная детализация может выполняться в двух основных направлениях.

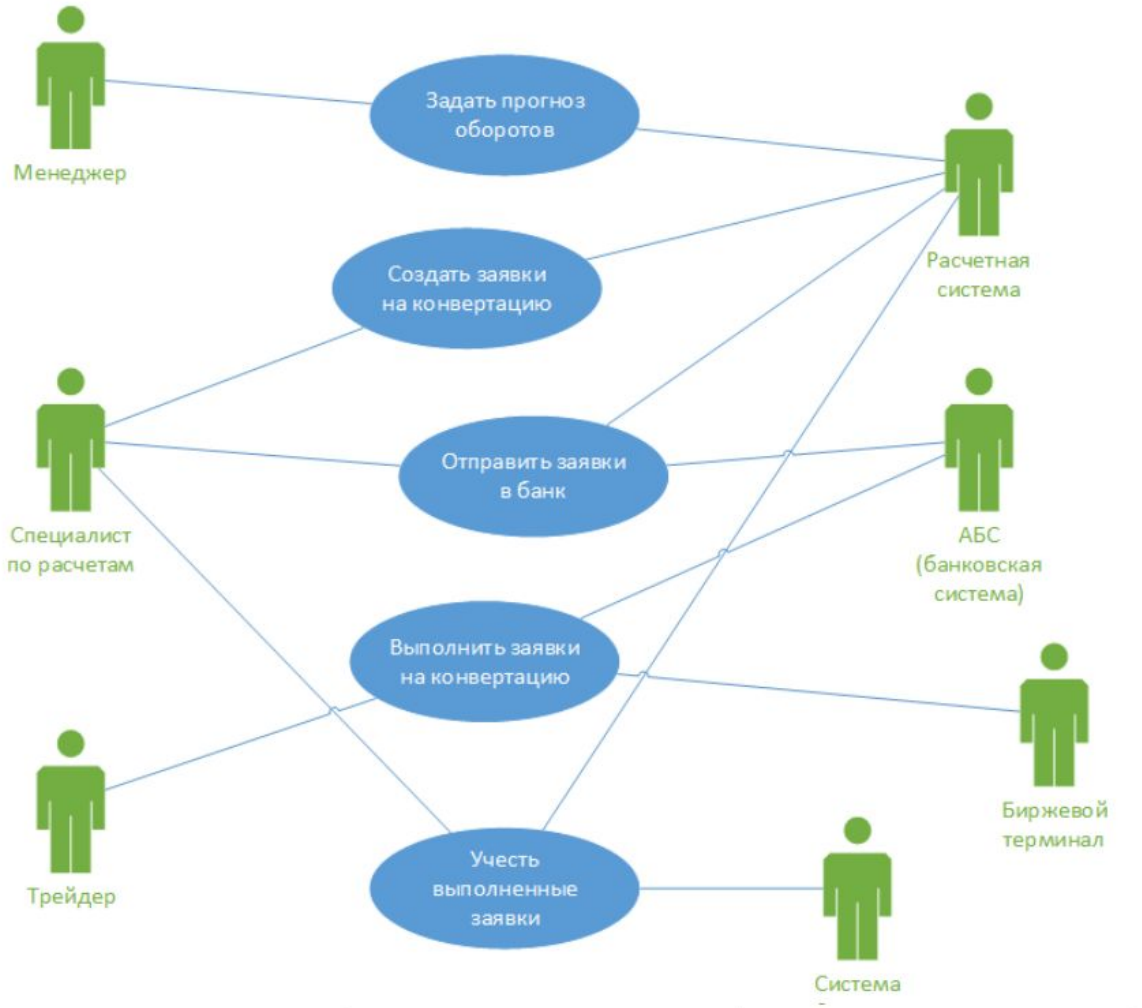
В рамках рассматриваемой системы продажи товаров может иметь самостоятельное значение и специфические особенности отдельная категория товаров — компьютеры. В этом случае диаграмма может быть дополнена вариантом использования "Оформить заказ на покупку компьютера" и актерами "Покупатель компьютера" и "Продавец компьютеров", которые связаны с соответствующими компонентами диаграммы отношением обобщения

Диаграмма вариантов использования *(use case diagram)*

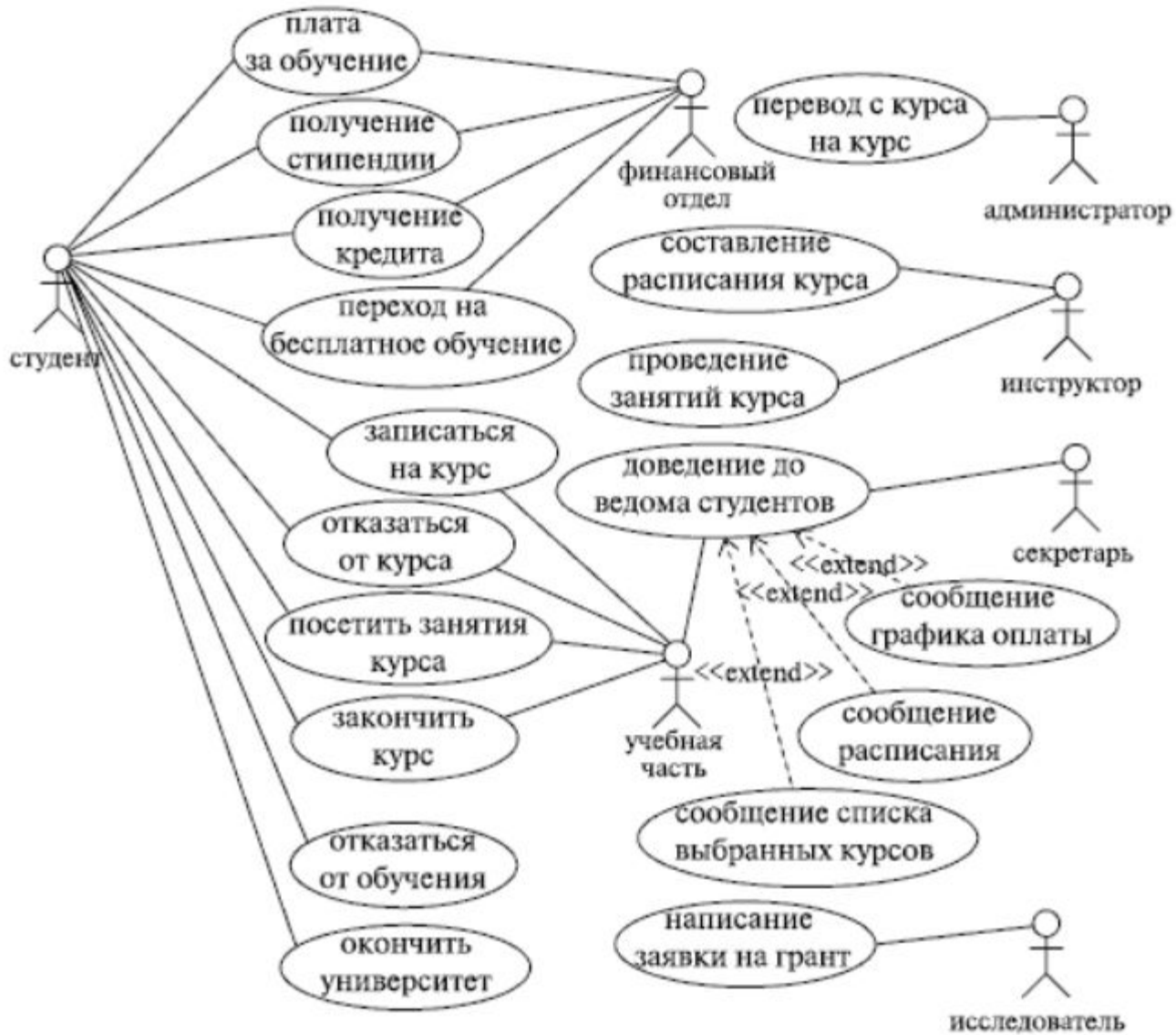
- диаграмма, на которой изображаются варианты использования проектируемой системы, заключенные в границу системы, и внешние актеры, а также определенные отношения между актерами и вариантами использования



Платежная система обращается в банк чтобы обменять валюту на такую сумму, которой хватит держателям валютных кошельков и получателям платежей.



1. Задать прогноз оборотов
2. Создать заявки на конвертацию
3. Отправить заявки в банк
4. Выполнить заявки на конвертацию
5. Учесть выполненные заявки


плата за обучение
получение стипендии
получение кредита
переход на бесплатное обучение
студент
финансовый отдел
перевод с курса на курс
администратор
составление расписания курса
проведение занятий курса
инструктор
записаться на курс
отказаться от курса
посетить занятия курса
закончить курс
отказаться от обучения
окончить университет
доведение до ведома студентов
секретарь
<<extend>>
<<extend>>
сообщение графика оплаты
<<extend>>
учебная часть
сообщение расписания
сообщение списка выбранных курсов
написание заявки на грант
исследователь

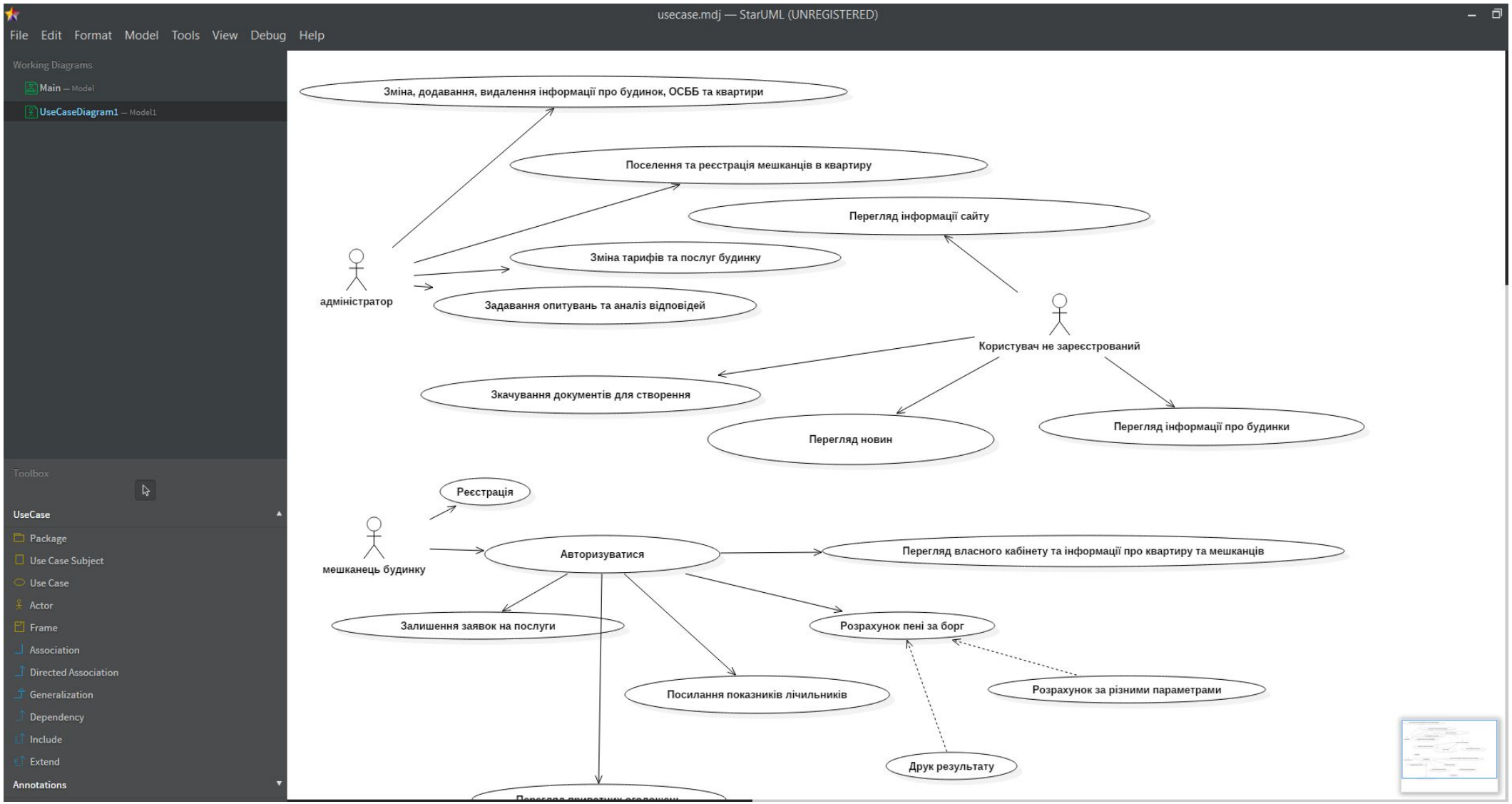
Порядок построения Usecase Diagram

1. Создать usecase диаграмму с именем «Основная функциональность»
2. Проанализировать, какие активные субъекты должны взаимодействовать с будущей системой.
3. Создать actor'ов. (Например, Менеджер, Бухгалтер и Кладовщик).
4. Создать прецеденты. Например,
    - Оформление заказа.
    - Оформление счёта.
    - Оформление накладной.
    - Выдача товара.
5. Для пояснения можно использовать комментарии.
6. Расставить связи, обозначающие зависимость (необходимо продумать, какие прецеденты находятся в отношении зависимости).
7. Результатом является подобная диаграмма:


Менеджер
Оформление заказа
Сначала проверяет наличие продуктов на складе
Оформление счёта
Бухгалтер
Оформление накладных
После оплаты клиентом
Кладовщик
Выдача товара
По накладной, предоставленной клиентом

Инструменты проектирования

StarUML™ - программная платформа моделирования, которая поддерживает UML (Унифицированный Язык Моделирования). Она основана на версии UML 1.4 и поддерживает нотацию UML версии 2.0 и одиннадцать различных типов диаграмм. Она активно поддерживает подход MDA (Архитектура Управляемая Моделью) и концепцию профилей UML. StarUML™ превосходен в отношении настройки окружения пользователя и имеет высокую степень расширяемости в том, что касается его функциональных возможностей.


usecase.mdj — StarUML (UNREGISTERED)
File Edit Format Model Tools View Debug Help
Working Diagrams
Main — Model
UseCaseDiagram1 — Model1
Toolbox
UseCase
Package
Use Case Subject
Use Case
Actor
Frame
Association
Directed Association
Generalization
Dependency
Include
Extend
Annotations
Зміна, додавання, видалення інформації про будинок, ОСББ та квартири
Поселення та реєстрація мешканців в квартиру
Перегляд інформації сайту
Зміна тарифів та послуг будинку
Задавання опитувань та аналіз відповідей
адміністратор
Користувач не зареєстрований
Скачування документів для створення
Перегляд новин
Перегляд інформації про будинки
Реєстрація
мешканець будинку
Авторизуватися
Перегляд власного кабінету та інформації про квартиру та мешканців
Залишення заявок на послуги
Розрахунок пені за борг
Посилання показників лічильників
Розрахунок за різними параметрами
Друк результату

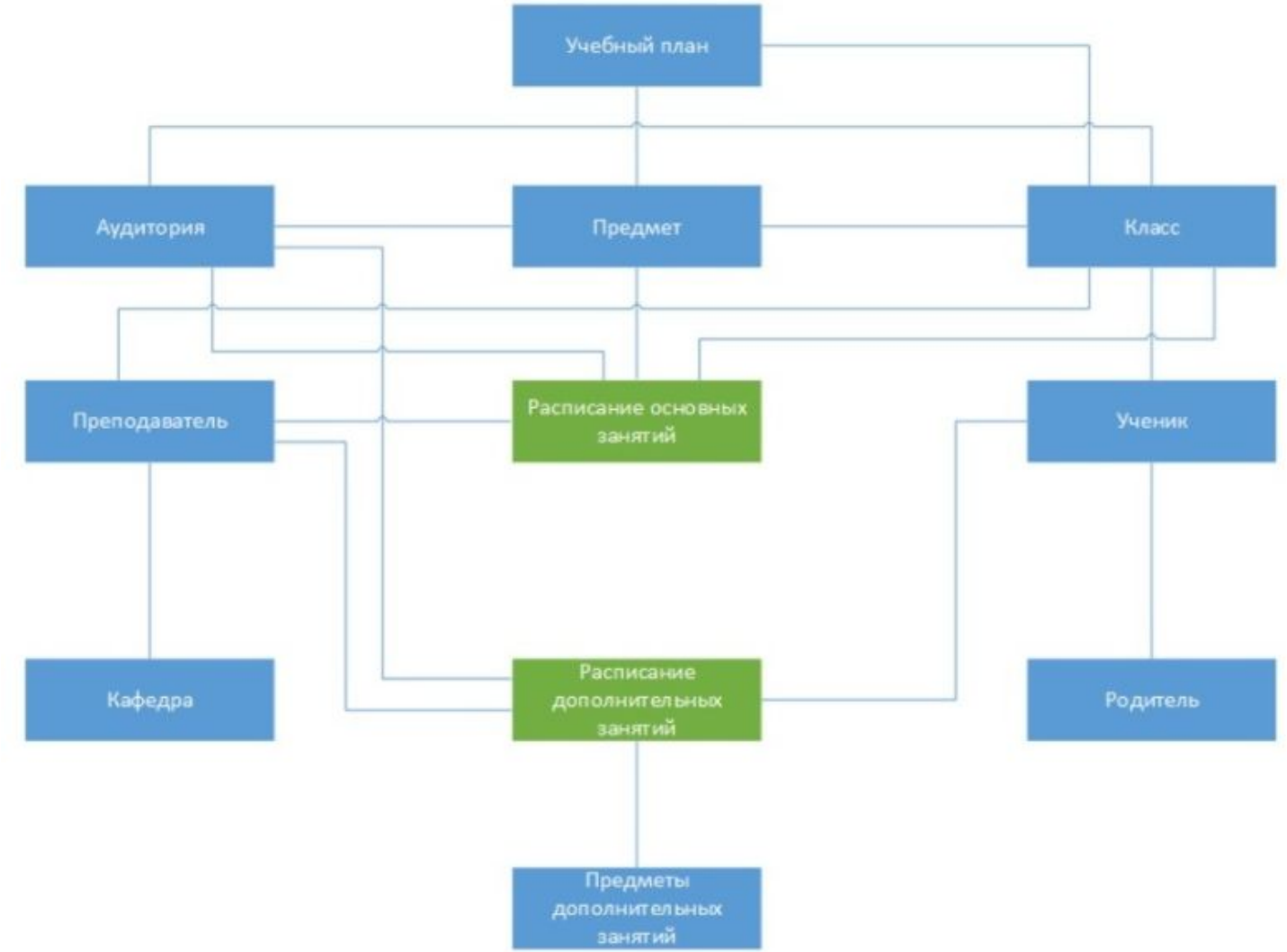
4. Пример базовой части Use Case. Регистрация пользователя в информационной системе

4.1. Постановка задачи

Система	Система входа пользователя в информационную систему
Основное действующее лицо	Пользователь
Цель	Войти в систему
Триггер	Пользователь решает зарегистрироваться в системе и заходит на страницу регистрации информационной системы
Результат	Информация о регистрации пассажира сохранена. Пользователь входит в систему
	У пользователя есть логин и пароль. Пользователь входит в систему

Практический пример

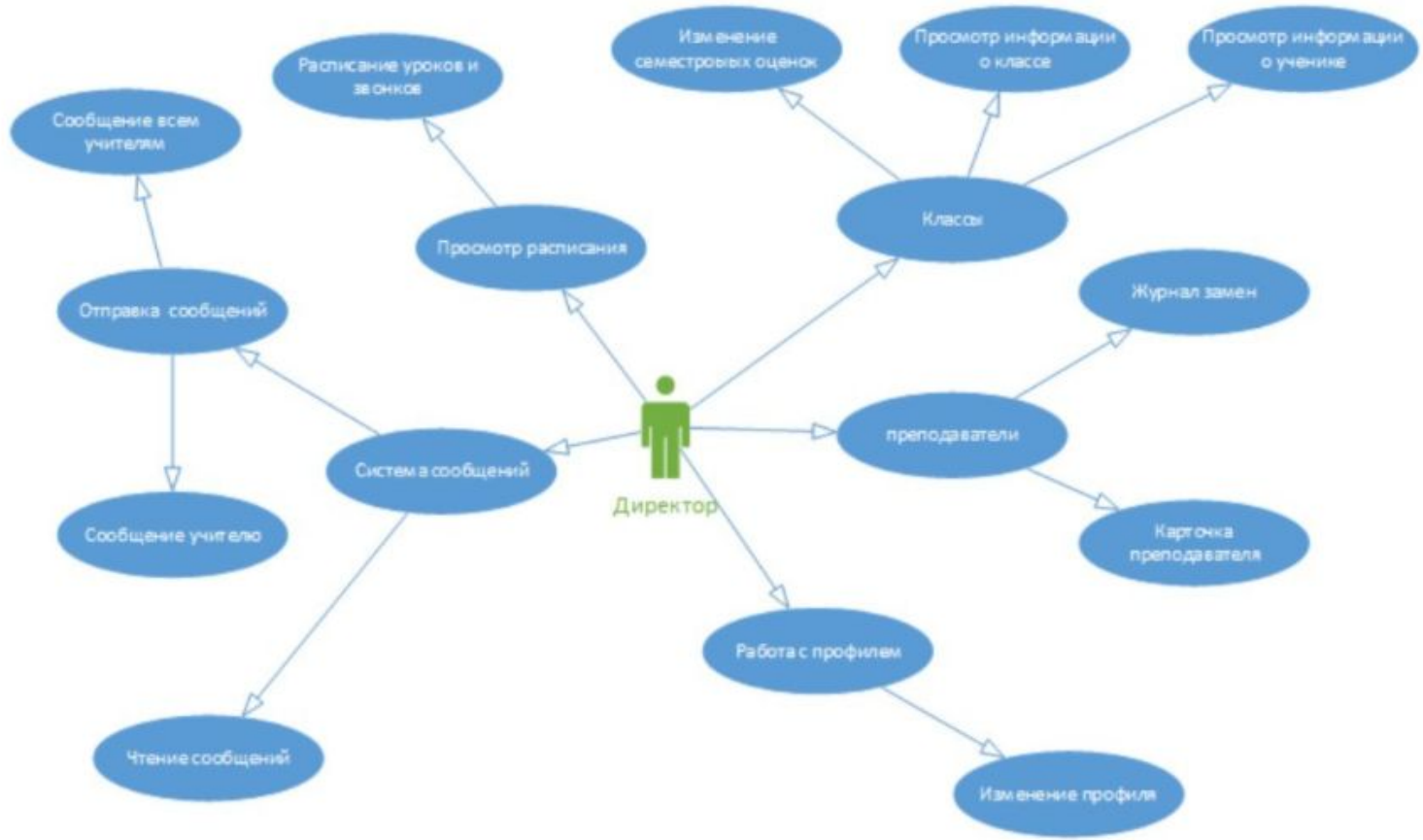
Элементы предметной области



АРМ "Директор школы"

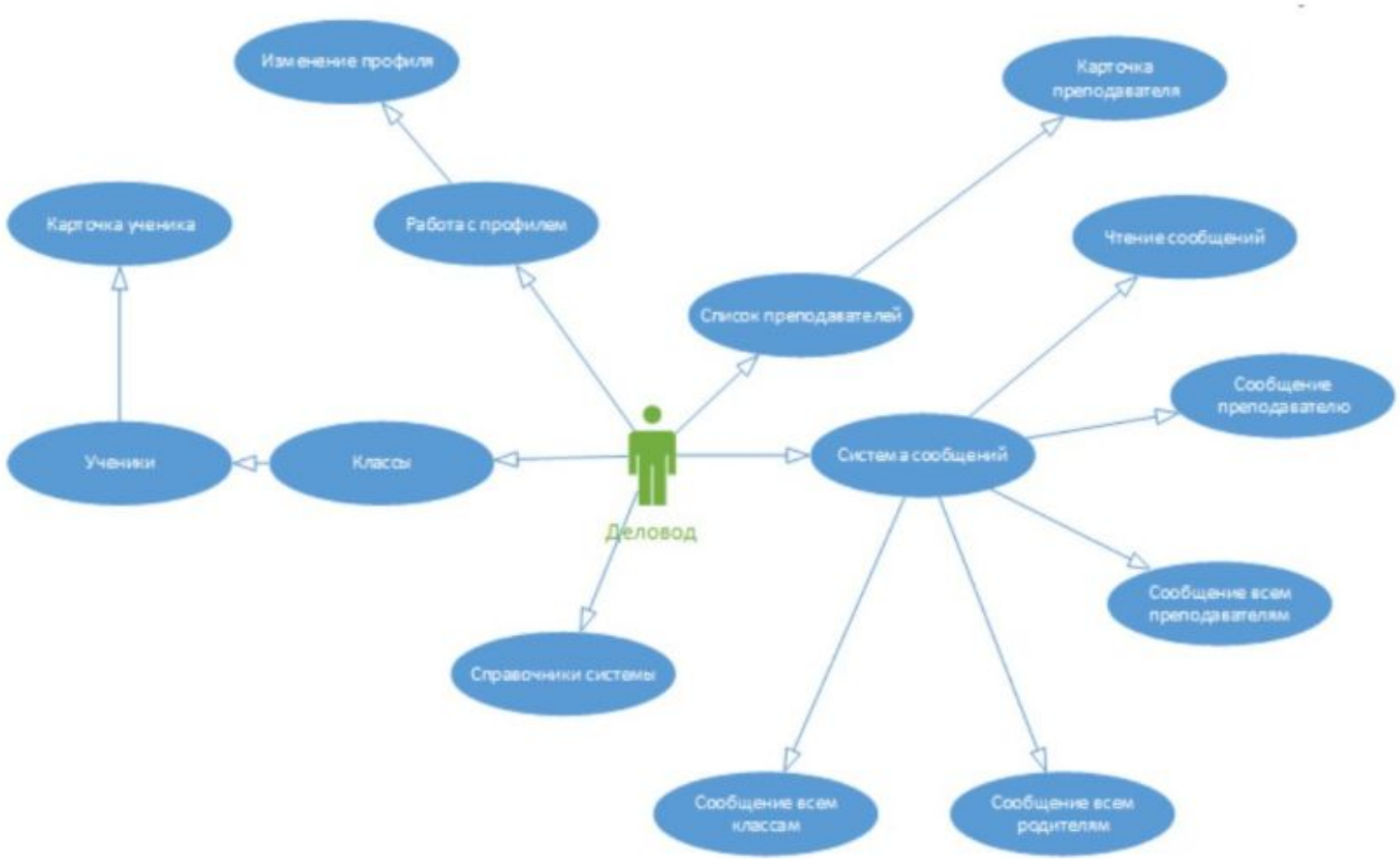
Разработка рабочего места АРМ "Директор школы" для работы через web-интерфейс:

- расписание уроков и звонков для всей ш
- "список сотрудников",
- "Личная карточка работника",
- "Список классов",
- "Список учеников в классе"
- "Личное дело ученика".
- Доступ к "общешкольным документам".



АРМ "Деловод"

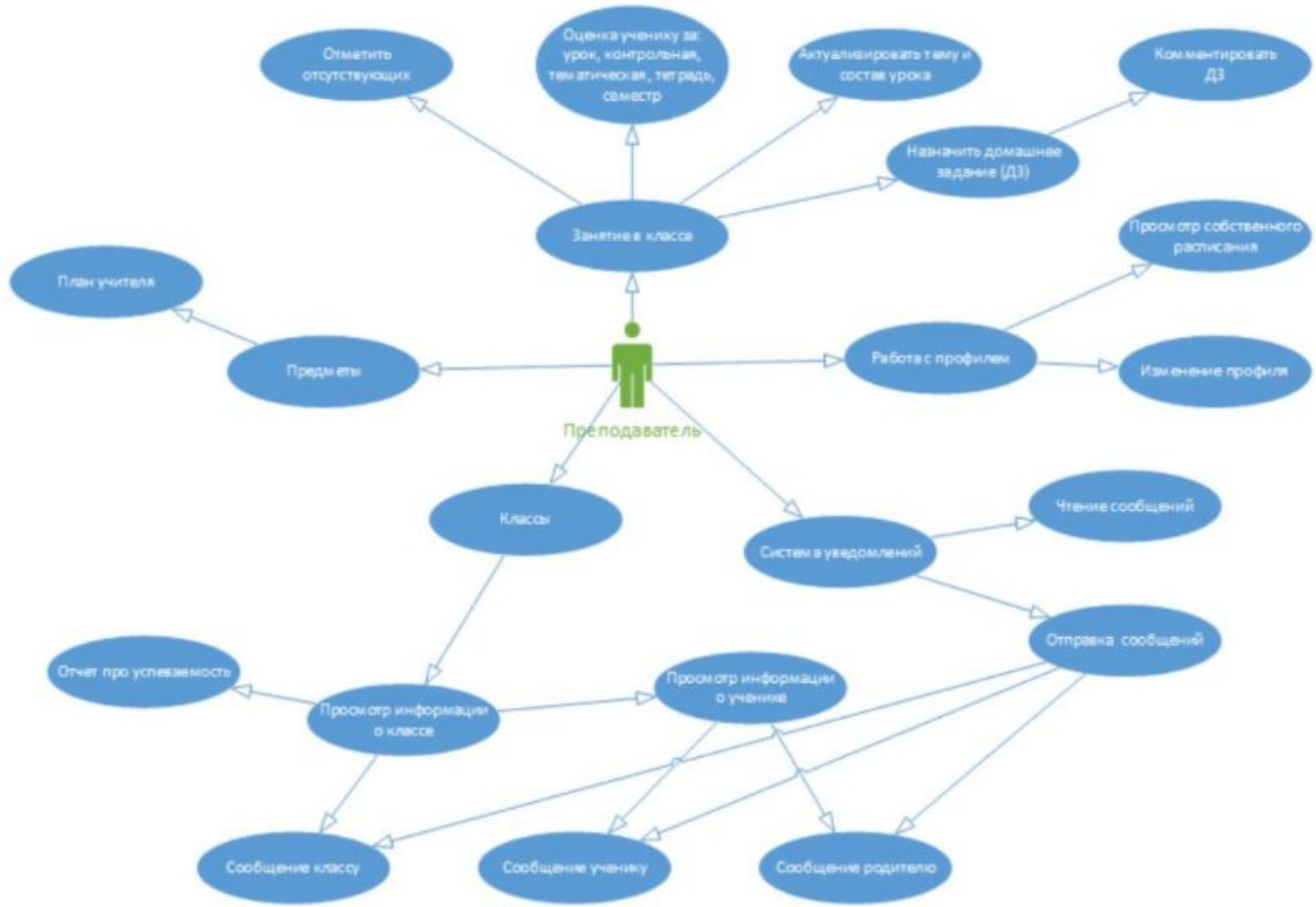
Разработка рабочего места АРМ "Деловод" через web-интерфейс: возможность



АРМ "Преподаватель"

Разработка рабочего места АРМ "Преподаватель"

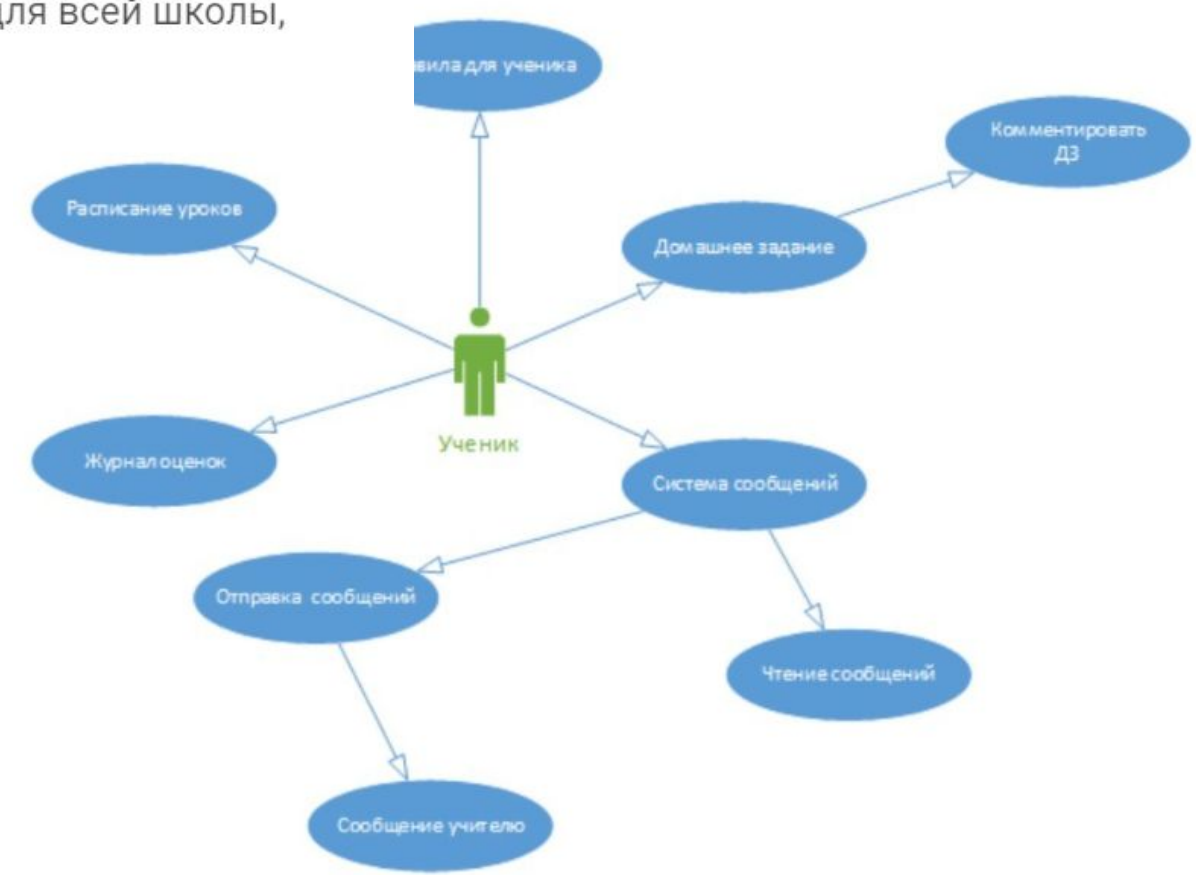
- План учителя
- Занятие в классе
- Расписание преподавателя
- Система уведомлений
- Список классов
- Ученики в классе
- Личная карточка ученика
- Обсуждение домашнего задания



АРМ "Ученик"

- Расписание уроков ученика
- Журнал оценок
- Домашнее задание
- Система сообщений

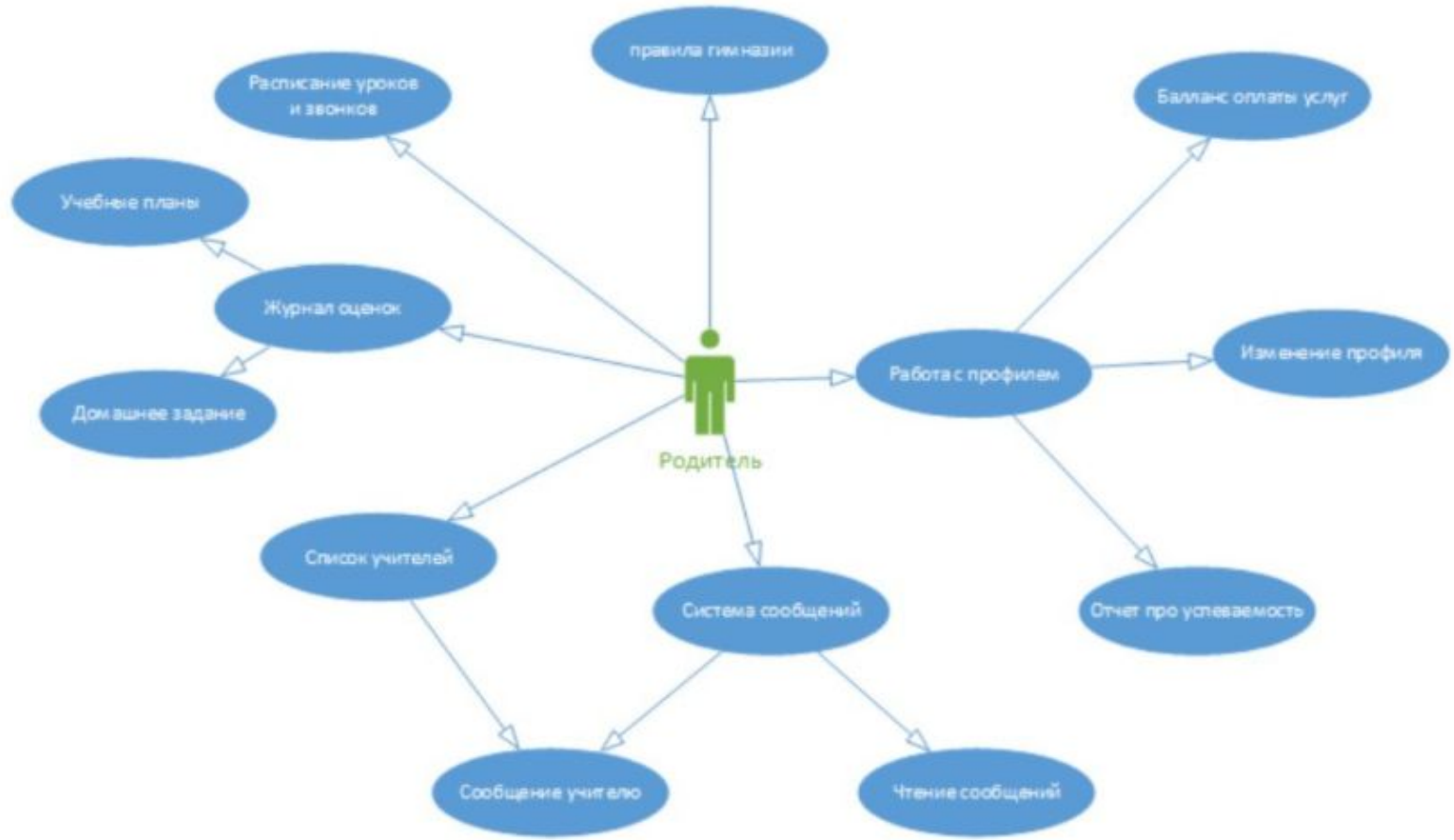
Правила для ученика - статическая страница, единая для всей школы, доступна для адаптивного просмотра



АРМ "Родитель"

- Расписание уроков ученика
- Журнал оценок
- Домашнее задание
- Отчет по успеваемости
- Список преподавателей
- Учебные планы (План учителя)
- Система сообщений

Правила для ученика - статическая страница, едина
доступна для адаптивного просмотра



ID and Name:	**UC-5 Поддержка системных справочников**		
Primary Actor:	Контент-менеджер (секретарь) Администратор	Secondary Actors:	System
Description:	В системе несколько базовых справочников, которые заполняются изначально, редко изменяются. Данные из этих справочников используются в системе повсеместно, но изменяются редко. Наиболее частое действие при использовании - это выбор нужного элемента из справочника в котором порядок вывода элементов пред настроен. Выбору иногда может предшествовать предварительный поиск по первым символам названия элемента. Редко используемые действия, выполняемые в специальном разделе настроек: создание новой записи, редактирование записи, упорядочивание и реорганизация - задать порядок показа по умолчанию. Заданный порядок показа элементов по умолчанию используется во всех местах использования справочника в системе. Типовой набор полей системных справочников: • Активность • **Название** • Сортировка При редактировании система проверяет уникальность обязательного поля **Название** Системные справочники: • Типы аудиторий • Области знаний		

	• Области знаний • Кафедры • Тип преподавания • Год обучения 1. 2016/2017 2. 2017/2018 • Образование • Уровень владения иностранными языками • Квалификационная категория
Trigger:	Пользователь инициирует необходимость изменить выбранный системный справочник.
Ограничения:	CON-1 Название элемента должно быть уникальным для справочника. CON-2 Элемент справочника не может быть удален, если где-либо используется в системе
Preconditions:	PRE-1 Пользователь аутентифицирован в системе и обладает достаточными правами для управления системными справочниками
Postconditions:	
Бизнес-правила	
Priority:	Low

ID and Name:	**UC-10 Добавить новую аудиторию, изменить существующую**		
Primary Actor:	Контент-менеджер (секретарь) Администратор	Secondary Actors:	System
Description:	Пользователь должен заполнить как минимум обязательные поля **“название аудитории”, “количество мест для учеников”**. “Название аудитории” вводится в виде текста и должно отражать назначение аудитории, может содержать ее особые признаки для однозначной идентификации: номер аудитории, читаемый предмет, для каких классов чаще всего используется. Пользователь может ограничить **читаемые предметы** в аудитории конкретным списком, выбирая допустимые из справочника “предмет”. Для удобства использования, в справочнике аудиторий предусмотрена классификация системным справочником **типы аудиторий** Аудитория может принадлежать только к одной группе. Признак **“Активность”** - по-умолчанию устанавливается в Да. Если признак не активный - аудитория не будет учитываться при автоматическом составлении расписания. Для сохранения изменений в элементе аудитория, пользователь выполняет действие “Сохранить”. Система осуществляет проверку на уникальность **“названия аудитории”**, блокирует создание аудиторий с идентичными названиями. Сохраняет данные либо выдает информацию о возникших ограничениях. Данный функционал доступен для десктопной версии.		

Trigger:	Пользователь инициирует необходимость добавления нового или изменения существующего элемента **аудитория**.
Ограничения:	CON-1 Название аудитории должно быть уникальным в системе CON-2 Количество мест для учеников - натуральное число. Контролируется диапазон [1 - 500]
Preconditions:	PRE-1 Пользователь аутентифицирован в системе и обладает достаточными правами для управления данной сущностью и связанными с ней справочниками. PRE-2 Все необходимые связанные справочники существуют в системе
Postconditions:	POST-1 В системе появляется новый экземпляр сущности с введенными характеристиками.
Бизнес-правила	BR-26, BR-28, BR-30, BR-32, BR-34
Priority:	Low


АУДИТОРИИ > Младшие классы 4/12
СИНХРОНИЗИРОВАТЬ
Добавить группу аудиторий
Добавить аудиторию
1
НАСТРОИТЬ ВИД
ФИЛЬТРЫ
Группы аудиторий
Младшие классы (4)
Общий фонд (2)
Специализированные аудитории (6)
N п/п
Активность
Название
Номер аудитории
Количество мест
1 Да 101 "Кабинет информатики" 101 20
2 Да 106 "Кабинет математики" 106 22
3 Нет 205 "Кабинет географии" 205 16
4 Да 159 "Кабинет рисования" 159 20
Показывать на странице 10
> Сохранить
2
Создать копию
Название
106 "Кабинет математики"
Группа аудиторий
Включить активность
Номер аудитории 101
Вместимость, чел 20
Закрепить за классом 4А
Читаемые предметы
1 Высшая математика


АУДИТОРИИ > Младшие классы 4/12
СИНХРОНИЗИРОВАТЬ
Добавить группу аудиторий
Добавить аудиторию
НАСТРОИТЬ ВИД
ФИЛЬТРЫ
Группы аудиторий
Младшие классы (4)
Общий фонд (2)
Специализированные аудитории (6)
N п/п
Активность
Название
Номер аудитории
Количество мест
1 Да 101 "Кабинет информатики" 101 20
2 Да 106 "Кабинет математики" 106 22
3 Нет 205 "Кабинет географии" 205 16
4 Да 159 "Кабинет рисования" 159 20
Показывать на странице 10
> Сохранить
Создать копию
Название
101 "Кабинет информатики"
Группа аудиторий
Младшие классы
Выключить активность
Номер аудитории 101
Вместимость, чел 20
Закрепить за классом 4А
Читаемые предметы
1 Робототехника
2 Основы программирования

ID and Name:	**UC-20 Добавить новый, изменить имеющийся, учебный предмет**		
Primary Actor:	ЗАВУЧ Администратор	Secondary Actors:	System
Description:	Пользователь должен заполнить как минимум обязательные поля **“Код предмета”, “Название предмета”, “Тип предмета”, “Область знаний”**. “Название предмета” вводится в виде текста и должно быть уникальным в системе. Значения свойств “Тип предмета“ и “Область знаний” устанавливается из соответствующих справочников. Для удобства использования, в справочнике учебных предметов в свойстве **“Область знаний”** использует соответствующий системный справочник. Признак **“Активность”** - по-умолчанию устанавливается в Да. Признак “сортировка” по-умолчанию устанавливается в 500, признак “сложность предмета” по-умолчанию устанавливается в 5. Для сохранения изменений или добавления нового “учебного предмета”, пользователь выполняет действие “Сохранить”. Система осуществляет проверку на уникальность **“Код предмета”** и **“название предмета”**, проводит валидацию обязательных полей, блокирует создание учебных предметов с идентичными названиями и кодами. Сохраняет данные либо выдает информацию о возникших ограничениях. Данный функционал доступен для десктопной версии.		
Trigger:	Пользователь инициирует необходимость добавления нового или изменения имеющегося учебного предмета.		

Trigger:	Пользователь инициирует необходимость добавления нового или изменения имеющегося учебного предмета.
Ограничения:	CON-1 Название учебного предмета и кода предмета должны быть уникальными в системе CON-2 Обязательные поля должны быть заполнены
Preconditions:	PRE-1 Пользователь аутентифицирован в системе и обладает достаточными правами для управления данной сущностью и связанными с ней справочниками. PRE-2 Все необходимые связанные справочники существуют в системе
Postconditions:	POST-1 В системе появляется новый экземпляр сущности с введенными характеристиками.
Бизнес-правила	
Priority:	Low

ID and Name:	**UC-35 Добавить нового преподавателя, редактировать существующего**		
Primary Actor:	ЗАВУЧ Администратор	Secondary Actors:	System
Description:	Пользователь должен заполнить обязательные поля: **ФИО**, **“Кафедра”**, **“Область знаний”**. “ФИО” вводится в виде текста и должно быть уникальным в системе. Значения свойств “Кафедра“ и “Область знаний” устанавливается из соответствующих системных справочников. Для преподавателя могут быть заданы условия **доступности** по дням недели (методические дни) и/или определенным урокам в конкретные дни недели. По-умолчанию преподаватель доступен для всех уроков. Пользователь может заполнить или отредактировать необязательные данные: • **Инициалы** - используются в расписании • “Читаемые предметы” - множество элементов “предмет”. • “Телефон” - множественное значение, один установлен как основной. • “EMAIL” - множественное значение, один установлен как основной. • Дата рождения • Образование - заполняется из системного справочника “Образование” • Пол • Квалификационная категория - заполняется из системного справочника “Квалификационная категория” • Уровень владения иностранными языками - заполняется из системного справочника “Уровень владения иностранными языками” • Примечания • Сортировка - число, применяется для упорядочивания при показе в списке		

	Для сохранения изменений элемента “Преподаватель”, пользователь выполняет действие “Сохранить”. Данный функционал доступен для десктопной версии.
Trigger:	Пользователь инициирует необходимость добавления нового преподавателя или редактирование существующего.
Ограничения:	CON-1 **ФИО** преподавателя должно быть уникальным CON-2 Обязательные поля должны быть заполнены
Preconditions:	PRE-1 Пользователь аутентифицирован в системе и обладает достаточными правами для управления данной сущностью и связанными с ней справочниками. PRE-2 Все необходимые связанные справочники существуют в системе
Postconditions:	POST-1 В системе появляется новый экземпляр сущности с введенными характеристиками.
Бизнес-правила	
Priority:	Low

ПРЕПОДАВАТЕЛИ > Кафедра украинского языка и литературы 4/12

СИНХРОНИЗИРОВАТЬ Добавить нового преподавателя НАСТРОИТЬ ВИД ФИЛЬТРЫ

Кафедры

- Кафедра украинского языка и литературы (4)
- Кафедра иностранных языков (4)
- Кафедра истории (4)

	N п/п	Активность	ФИО	Область знаний	Читаемые предметы	Квалификация	Кон тел
☐	1	☑	Иванова Мария Ивановна	украинский язык	украинский язык	преподаватель I категории	+38
☐	2	☑	Сидорова Галина Николаевна	украинский язык	украинский язык; правописание	преподаватель высшей категории	+38
☐	3	☑	Войтенко Александр Васильевич	литература	украинская литература	преподаватель I категории	+38
☐	4	☑	Петриченко Людмила Николаевна	литература	украинская литература	преподаватель высшей категории	+38

Показывать на странице 10

> Сохранить Создать копию

ФИО
Иванова Мария Ивановна

Инициалы
Иванова М. И.

Кафедра
украинского языка и литературы

Область знаний
украинский язык

Читаемые предметы
украинский язык ⊕

Квалификация
преподаватель I категории

Контактный телефон
основной +38 050 555-10-20 ⊕
+38 093 658-02-05

E-mail
ivanova@school.com ⊕

Дата рождения (ДД / ММ / ГОД): 20 / 05 / 1978

Образование
Высшее образование, Аспирантура

Пол ☑ Ж ☐ М

Программы для UML

1) StarUML

StarUML — это инструмент моделирования программного обеспечения с открытым исходным кодом. Это обеспечивает одиннадцать типов диаграмм. StartUML 2 совместим с версиями UML 2.x.

Особенности:

- Позволяет создавать диаграммы Obje3ct, Use case, Deployment, Seque3nce, Communication, Activity и профиля.
- Позволяет обнаруживать и устанавливать сторонние расширения.
- Работайте с одним и тем же UX на нескольких платформах, включая macOS, Windows и Linux.
- Нет ограничений для использования этого коммерческого программного обеспечения для оценки.

Ссылка для скачивания: http://staruml.io/

2) Умбрелло:

Umbrello — это инструмент моделирования UML. Работает под KDE и Linux. Инструмент также поддерживает генерацию кода и реверс-инжиниринг для C ++ и Java.

Особенности:

- Позволяет создавать схемы программного обеспечения и другой системы в стандартном формате.
- Это поможет вам проверить скриншот, чтобы увидеть umbrello в действии.
- Предлагает руководство по обучению Umbrello и UML-моделированию.

Ссылка для скачивания: https: //umbrello.kde.org/

3) Эдро Макс

Edraw Max — это программа для построения UML, которая помогает вам создавать диаграммы с использованием готовых символов и шаблонов. Это позволяет вам импортировать ваши рисунки в форматы файлов, такие как PDF, PPT, Word, HTML и т. Д.

Особенности:

- Вы можете создать блок-схему, интеллектуальную карту, UML, электрические схемы, сетевые диаграммы и т. Д.
- Он предоставляет удобный интерфейс, похожий на MS Word.
- Edraw Max поможет вам поделиться дизайном в любое время и в любом месте.
- Этот инструмент предоставляет более 280 новейших решений для схем и диаграмм.

4) UML дизайнерский инструмент:

Инструмент UML Designer предлагает набор общих диаграмм для работы с моделями UML 2.5. Этот инструмент предоставляет простой способ перехода от UML к предметно-ориентированному моделированию.

Особенности:

- Позволяет пользователю повторно использовать предоставленные представления и работать с полной прозрачностью как на моделях DSL, так и на моделях UML.
- Помогает вам создать диаграмму классов, диаграмму компонентов и составную диаграмму структуры
- Позволяет использовать устаревшие модели UML и начать работу с DSL.

Ссылка для скачивания: http://www.umldesigner.org/

5) Альтова

Altova UModel — это еще один полезный инструмент UML, который делает визуальный дизайн программного обеспечения практичным для любого проекта. Визуально проектируйте модели приложений в UML, которые могут быть сгенерированы с использованием Java, C ++, C # или Visual Basic.

Особенности:

- Интуитивное визуальное моделирование для всех диаграмм UML
- Вспомогательные окна позволяют строить нужные модели.
- Это позволяет добавлять гиперссылки к любому элементу в любой диаграмме UML.
- Вы можете назначить элемент определенному слою, и слои могут быть заблокированы, чтобы предотвратить изменения.

Ссылка для скачивания: https://www.altova.com/umodel

6) Umple

Umple — это модель с открытым исходным кодом для интеграции текстовых конструкций UML в языки программирования, генерации кода или использования простого метода моделирования UML.

Особенности:

- Это позволяет разработчикам встраивать шаблоны концепций моделирования, шаблоны генерации и другие абстракции в традиционный код.
- Инструмент Umple помогает пользователям быстрее изучать UML.
- Инструмент может работать онлайн, как плагин Eclipse, а также автономная командная строка Jar.

Ссылка для скачивания: https://cruise.eecs.uottawa.ca/umple/

7) Визуальная Парадигма

Visual Paradigm — это инструмент разработки программного обеспечения, разработанный специально для программных проектов двигателей. Этот инструмент UML помогает команде разработчиков программного обеспечения моделировать информационную систему бизнеса и процессы разработки.

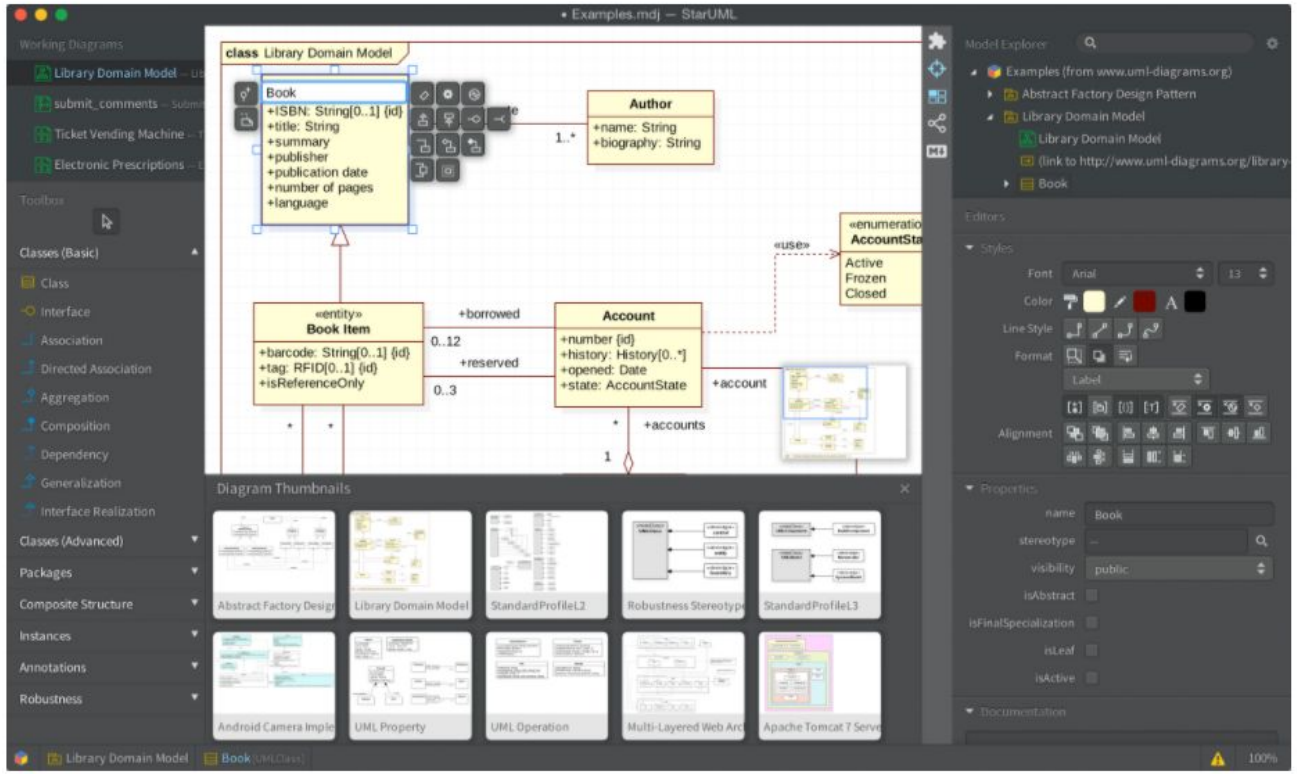
Особенности:

- Он предлагает поддержку BPMN, UML, ERD, DFD, SysML.
- Он предлагает полный инструмент для анализа процессов, проектирования систем, проектирования баз данных и т. Д.
- Предлагает функцию пользовательских историй для захвата и поддержания потребностей пользователей.

Ссылка для скачивания: https://www.visual-paradigm.com/

Star UML

StarUML - это сложный программный разработчик, предназначенный для поддержки *гибкого* и *краткого* моделирования.



Основными целями пользователей являются:

- Гибкие и небольшие команды разработчиков
- Профессиональные люди
- Учебные заведения.

Ключевые особенности StarUML:

- Многоплатформенная поддержка (MacOS, Windows и Linux)
- Совместимость со стандартом UML 2.x
- Поддержка SysML
- Диаграмма сущность-связь (ERD)
- Диаграмма потока данных (DFD)
- Блок-схема
- Несколько окон
- Современный UX
- Темные и светлые темы
- Поддержка дисплея Retina (High-DPI)
- Поддержка сенсорной панели MacPro Pro
- Модельно-ориентированная разработка
- Открытые API
- Различные сторонние расширения
- Проверка асинхронной модели
- Экспорт в HTML-документы
- Автоматические обновления.

Задания.

- 1. Создать use case диаграмму по варианту.
- 2. Описать.

Варианты

1. Веб додаток, системи управління закладом харчування створюється з метою максимально автоматизувати роботу персоналу закладів харчування. Робота із системою поділяється на дві частини: робота офіціянта або менеджера та робота адміністратора або бухгалтера.

Вимоги користувачів.

Менеджери мають доступ до наступних функцій:

- Авторизація в системі за кодом;
- Вибір столика для замовлення;
- Формування замовлення та розрахунок клієнтів;
- Зміна замовлень;
- Друк рахунку замовлень;
- Перегляд списку замовлень за дату;
- Перегляд фінансового стану на сьогодні;
- Формування залишків коштів;
- Зміна статусу замовлень;
- Пошук блюд за категоріями;
- Облік блюд за наявністю.

Вимоги адміністратора:

1. Авторизація за логіном та паролем.
2. Робота із постачальниками;
3. Робота із обліком сировини;
4. Формування приходних накладних за сировиною;
5. Формування блюд за включенням сировини;
6. Розрахунок вартості блюд з урахуванням собівартості, роботи та націнки;
7. Робота із категоріями блюд та сировини;
8. Формування звіту за замовленнями;
9. Формування звітів за датою, популярністю блюд та доходами та витратами;
10. Друк інформації.

Вариант 2.

Веб орієнтована система розрахунків коммунальних платежів створюється з метою максимально автоматизувати роботу персоналу житлово комунальних господарств з нарахування та обліку послуг. Робота із системою поділяється на дві частини: робота робітника житлово комунального господарства та користувача послуг.

Вимоги користувачів.

1. Отримання інформації про ЖКГ.

3. Зареєструвати новий будинок.

Внутрішні користувачі – адміністратори мають доступ до додаткових функцій, таких як:

1. Редагування інформації про будинки.

2. Редагування інформації про мешканців та тарифи.

3. Інформаційний обмін з зовнішніми користувачами за допомогою опитувань.

Зовнішній користувач авторизований та зареєстрований (мешканець) –мають доступ до додаткових функцій, таких як:

1. Посилання показників лічильників.

2. Перегляд архіву власних проплат.

3. Відповідати на опитування;

4. Перегляд інформації про квартиру та її мешканців;

5. Залишення заявок на послуги ЖКГ (довідки паспортного столу)

6. Розрахунок вартості комунальних послуг з урахування прострочених платежів та нарахування пені.

Вариант 3

Бізнес-вимоги

Основні цілі: проект створюється з метою донесення інформації до користувача про кредитні програми , кредитні графіки та платежі, кредитування під заставу різні розрахунки за кредитними програмами.

Представлення проекту: проект буде реалізовано у вигляді сайту, що містить структуровану інформацію про програми кредитування.

Розрахунок часу відповіді на запитання користувачів: відповідь виконується за короткий термін часу.

Вимоги користувачів

- зовнішні користувачі

1. Отримання інформації про кредитні програми.
2. Отримання інформації про проценти за кредитними програмами та умови виплат.
3. Перегляд інформації про новини та можливість задавати запитання.
4. Перегляд власного кабінету та графіку виплат
5. Можливість придбати кредит

- внутрішній користувач – адміністратор

1. Редагування структури сайту.
2. Редагування сторінок сайту.
3. Робота з запитаннями.
4. Обробка інформації(коментарів).
5. Робота з замовленнями(зміна статусу)
6. Робота з новинами
7. Робота з довідниками кредитної компанії

- внутрішній користувач – менеджер з продажу

1. Підбор індивідуальної кредитної програми
2. Зміна новин
3. Реєстрація клієнтів в систему
4. Розрахунок графіків кредитних виплат
5. Відповідь на запитання