# Задание

Нужно разработать систему для бизнеса по решению математических заданий.

Заказчик сообщил:

"Добрый день, я создатель фирмы "УчисьЛегко", мы решаем различный работы связанные с математикой. Наша клиентура - студенты-заочники и очники различных ВУЗов, техникумов, колледжей, шаражек, и прочих нор. У нас такая организация: нам звонит клиент, наш человек с ним общается, уточняет объём работы, срок, темы заданий. После обсуждения и озвучивания примерной цены (наш человек в лучшем случае имеет математическое образование, но в основном весьма далёк от темы, поэтому примерную цену он называет исходя из прайс листа, подготовленного специалистами).  
    После этого заказчик отправляет нам фотографии или файлы с заданиями, и мы начинаем искать того, кто решит эту работу. Решает работы постоянно-меняющийся штат, состоящий, как ни парадоксально, тоже из студентов. Только эти студенты обучаются по математическим специальностям. Тот, кто соглашается взять эту работу, корректирует уже озвученную заказчику цену, и в таком случае мы сообщаем последнему новую цифру.   
      Бывает так, что исполнитель попадается ненадёжный, а именно: срывает сроки, выполняет задания неверно, врёт нам о месте учёбы. Это приводит к срыву заказов и потере клиентов. Поэтому нам нужна система, в которой все исполнители будут иметь свой рейтинг надёжности, у каждого из них будет свой список тем, задания по которым они в состоянии решить. Хотелось бы, чтобы при регистрации исполнителю высылалось несколько пробных заданий по темам, которые, по его словам, он в состоянии решать. Чтобы исключить повторную регистрацию исполнителей с плохим рейтингом, хорошо бы иметь систему верификации.  
      В этой же системе должна быть предусмотрена регистрации заказчиков, и особая группа - модераторы. Задача модератора - компетентно оценить работу, которую нужно выполнить, проверить, нет ли в этой самой работе контактных данных заказчика, которые могут попасть к исполнителю. Последнее нужно для того, чтобы исключить связь исполнителя напрямую с заказчиком, потому как, вы сами понимаете, это не в наших интересах.  
      Мы также надеемся, что подобная система позволит избежать таких технических редких накладок, когда происходила потеря части файлов или всех заданий при передаче их от нашего человека к исполнителю.   
      Забыл упомянуть, но, мне кажется, это очевидно: наша выгода - процент от стоимости выполнения работ. Возьмётесь за это задание?"

Ниже будут использоваться следующие обозначения:

* Система – разрабатываемое приложение
* Заказчик – пользователь системы, использующий её с целью получить решение своих заданий.
* Исполнитель – пользователь системы, использующий её с целью обогащения, который получает задания от заказчика посредством системы.
* Заказ – набор заданий (может включать и 1 задание), решение которых заказчик хочет получить. Заказ выполняется исполнителем.
* Выполнение заказа – решение заданий, входящих в заказ исполнителем.

Этап сбора и анализа требований

Акторы и цели

Системный Администратор

* Установить систему
* Проверить состояние системы
* Устранить технический сбой

Незарегистрированный пользователь

* Зарегистрироваться (UC#1)

Исполнитель

* Получить заказ (UC#2)
* Отправить выполненный заказ (UC#3)
* Вывести деньги (UC#4)

Заказчик

* + - Отправить заказ (UC#5)
    - Получить выполненный заказ
    - Подать жалобу (UC#7)
    - Пополнить персональный счёт (UC #11)

Модератор

* + - Проверить задания (первичная модерация) (UC#8)
    - Проверить задания (вторичная модерация) (UC#10)
    - Рассмотреть жалобу (UC#9)

Банк

UC #1: Зарегистрироваться

Первичный актор: незарегистрированный пользователь

Другие акторы: система, модератор

Предусловия: пользователь не зарегистрирован

Триггер: новый пользователь хочет зарегистрироваться

Основной успешный сценарий:

1. Пользователь инициирует процесс регистрации
2. Система предоставляет выбор типа пользователя
3. Пользователь выбирает, каким пользователем он хочет быть, «Заказчиком» или «Исполнителем»
4. В зависимости от сделанного выбора, система предоставляет возможность для введения регистрационных данных
5. Пользователь вводит необходимые данные
6. Система проверяет возможность регистрации пользователя с такими данными
7. Система добавляет информацию о пользователе

Расширения (нештатные ситуации):

6а. Пользователь с таким именем уже существует в системе

6а1. Система предлагает пользователю ввести другой логин

6а2. Выполнение сценария продолжается с шага 5

6б. Система обнаружила попытку повторной регистрации исполнителя с низким рейтингом

6б1. Система уведомляет пользователя о невозможности продолжения регистрации

6б2. Выполнение сценария прекращается

UC #2: Получить заказ

Первичный актор: исполнитель

Другие акторы: система

Предусловия: в системе есть невыполненный заказ, и у исполнителя нет заказа «на руках»

Триггер:

Основной успешный сценарий:

1. Исполнитель инициирует просмотр доступных ему заказов
2. Система выводит данные о заказах, доступных данному исполнителю (кол-во заданий, темы заданий, стоимость, сроки)
3. Исполнитель выбирает определённый заказ и сообщает системе о своём выборе
4. Система сохраняет информацию о том, что пользователь взял заказ
5. Система предоставляет исполнителю файл с заданиями заказа

Расширения (Внештатные ситуации):

3а. Исполнителя не устраивает ни один из вариантов

3а1. Исполнитель инициирует отмену выбора

3а2. Выполнение сценария прекращается

UC #3: Отправить выполненный заказ

Первичный актор: исполнитель

Другие акторы: система, модератор

Предусловия: исполнитель имеет незавершённый заказ

Триггер: исполнитель завершил заказ

Основной успешный сценарий:

1. Исполнитель инициирует процесс отправки выполненного заказа
2. Система предоставляет средства для отправки заказа
3. Исполнитель вводит данные, прикрепляет файлы с выполненным заказом
4. Исполнитель инициирует отправку заказа на модерацию
5. Система сохраняет выполненный заказ и информацию о нём(дата, логин исполнителя)

UC #4: Вывести деньги

Первичный актор: исполнитель

Другие акторы: система, банк

Предусловия: исполнитель вывод средств возможен

Триггер: исполнитель хочет вывести средства

Основной успешный сценарий:

1. Исполнитель инициирует вывод средств
2. Система предоставляет средства для выбора способа вывода денег
3. Исполнитель выбирает нужный способ вывода и вводит необходимые данные(номер счёта/сумму и тд.)
4. Система проверяет данные
5. Система связывается с банком и инициирует перевод средств на счёт, введённый пользователем
6. Банк уведомляет систему об успешном проведении операции
7. Система сохраняет информацию об операции
8. Система уведомляет исполнителя об успешном проведении операции

Расширения (внештатные ситуации):

4а. Сумма на персональном счёте исполнителя меньше запрошенной им на вывод

4а1. Система уведомляет исполнителя о неправильных данных и даёт повторную возможность ввода данных.

4а2. Выполнение сценария продолжается с шага 3

UC #5: Отправить заказ

Первичный актор: заказчик

Другие акторы: система

Предусловия: заказчик авторизован в системе

Триггер: заказчик хочет отправить заказ

Основной успешный сценарий:

1. Заказчик инициализирует отправку заказа
2. Система предоставляет средства для ввода данных о заказе
3. Заказчик вводит необходимые данные
4. Система проверяет данные
5. Система выводит данные о необходимой предоплате (сумму предоплаты)
6. Заказчик выражает согласие на списание со счёта суммы, равной сумме предоплаты
7. Система списывает указанную сумму со счёта заказчика
8. Система сохраняет заказ и данные о предоплате

Расширения (внештатные ситуации):

4а. Некоторые данные не удовлетворяют некоторым условиям

4а1. Система уведомляет исполнителя о неправильно заполненных полях и даёт возможность ввести данные заново.

4а2. Выполнение сценария продолжается с шага 3

6а. Заказчик отказывается вносить предоплату в указанном размере

6а1. Система сохраняет заказ как неоплаченный

6а2. Выполнение сценария прекращается

7а. Сумма на счёте заказчика меньше суммы предоплаты

7а1. Система уведомляет заказчика о нехватке средств на счёте

7а2. Система сохраняет заказ как неоплаченный

7а3. Выполнение сценария прекращается

UC #7: Подать жалобу

Первичный актор: заказчик

Другие акторы: система, модератор

Предусловия: с момента последнего полученного заказчиком заказа прошло не больше недели

Триггер: заказчик не удовлетворён выполненным заказом

Основной успешный сценарий:

1. Заказчик инициирует подачу жалобы
2. Система предоставляет средства для подачи жалобы
3. Заказчик вводит данные о жалобе (номер заказа, текст претензии) и инициализирует их отправку
4. Система сохраняет жалобу как нерассмотренную

UC #8: Проверить задания (первичная модерация)

Первичный актор: Модератор

Другие акторы: исполнитель, система, заказчик

Предусловия: в системе зарегистрированы хотя бы один заказчик и исполнитель

Триггер: в систему поступил заказ от заказчика

Основной успешный сценарий:

1. Модератор инициирует просмотр списка непроверенных заказов
2. Система выводит список заказов
3. Модератор выбирает заказ
4. Система предоставляет возможность скачать выбранный заказ
5. Модератор проверяет и оценивает заказ
6. Модератор подтверждает корректность заказа и сообщает системе стоимость заказа
7. Система сохраняет данные о заказе(стоимость) уведомляет заказчика о полной цене заказа

Расширения (Внештатные ситуации):

6а. Модератор помечает заказ как не корректный

6а1. Модератор формирует сообщение о том, почему заказ получил отказ

6а2. Система уведомляет пользователя, подавшего заказ о том, что он не прошёл модерацию с текстом, составленным модератором в пункте 6а1

6а3. Выполнение сценария прекращается

UC #10: Проверить задания (вторичная модерация)

Первичный актор: Модератор

Другие акторы: исполнитель, система, заказчик

Предусловия: в системе зарегистрированы хотя бы один заказчик и исполнитель

Триггер: в систему поступил заказ от исполнителя

Основной успешный сценарий:

1. Модератор инициирует просмотр списка выполненных заказов, не прошедших вторичную модерацию
2. Система выводит список заказов
3. Модератор выбирает заказ
4. Система предоставляет возможность скачать выбранный заказ
5. Модератор проверяет заказ и подтверждает его корректность
6. Система сохраняет информацию о заказе(дату проверки, статус) и уведомляет заказчика

Расширения (Внештатные ситуации):

5а. Заказ не проходит модерацию

5а1. Модератор формирует сообщение о том, почему заказ получил отказ

5а2. Система уведомляет исполнителя, подавшего заказ о том, что он не прошёл модерацию с текстом, составленным модератором в пункте 5а1

5а3. Выполнение сценария прекращается

UC #9: Рассмотреть жалобу

Первичный актор: модератор

Другие акторы: исполнитель, система, заказчик

Предусловия: В системе есть хотя бы одна жалоба, требующая рассмотрения

Триггер:

Основной успешный сценарий:

1. Модератор инициирует просмотр списка нерассмотренных жалоб
2. Система выводит этот список
3. Модератор выбирает один из пунктов
4. Система выводит подробную информацию о жалобе (Дату, логин заказчика, текст претензии заказчика, файлы с выполненным заказом)
5. Модератор ознакамливается с предметом споров и удовлетворяет жалобу
6. Система сохраняет жалобу как рассмотренную
7. Система возвращает заказчику на счёт оплату за заказ

Расширения(Внештатные ситуации):

5а. Модератор отклоняет жалобу

5а1. Система уведомляет пользователя об отказе удовлетворения жалобы

5а1. Выполнение сценария прекращается

UC #11: Пополнить персональный счёт

Первичный актор: заказчик

Другие акторы: система, банк

Предусловия:

Триггер: Заказчик инициирует пополнение счёта

Основной успешный сценарий:

1. Заказчик инициирует пополнение счёта
2. Система предоставляет средства для ввода суммы которую заказчик хочет внести
3. Заказчик вводит сумму и подтверждает введённую информацию
4. Система предоставляет список вариантов оплаты
5. Заказчик выбирает способ оплаты из списка, предложенного системой
6. Согласно выбранному методу оплаты, система предоставляет пользователю средства для ввода информации (номер карты/ номер телефона/ и тд.)
7. Пользователь вводит необходимую информацию и инициирует её отправку
8. Система передаёт данные в банк и ждёт ответа
9. Банк списывает деньги со счёта пользователя (банковского или телефонного)
10. Банк переводит деньги на счёт, указанный в системе (счёт владельца системы)
11. Система увеличивает сумму на персональном счёте заказчика
12. Система сохраняет информацию об операции
13. Система уведомляет заказчика об успешном проведении операции

Расширения (внештатные ситуации):

9а. Банк отказывает системе в проведении операции

9а1. Система уведомляет пользователя об отказе банка

9а2. Выполнение сценария прекращается

Диаграмма вариантов использования (прецедентов)

разрабатываемой системы.

Диаграмма вариантов использования (Usecase Diagram)Концептуальная модель.

(Диаграмма классов разрабатываемой системы)

КМ

Диаграмма деятельностей.

Процесс обработки заказа в разрабатываемой системе.

Диаграмма активностей (Activity Diagram)

Диаграмма состояний объекта «Заказ»

в разрабатываемой системе.

Диаграмма состояний (Statechart Diagram)

Диаграмма последовательности выполнения

UC #5: Отправить заказ.

Диаграмма последовательности (Sequence Diagram)

**Пояснения:**

Вся Бизнес логика для данного сценария разбита на Регистратор заказов и Обработчик денежных операций.

*CreateOrder()* – проверка данных и создание заказа, согласно данным, внесённым заказчиком. Это занятие регистратора заказов.

*Result:Boolean* – true or false, успешно или нет прошла операция.

Регистратор заказов после создания записи в БД считает сумму предоплаты (CountPrepayment())

И возвращает *Prepayinfo:real* – сумму необходимой предоплаты.

*Writeoff(sum:real,ID\_order:int)* – списание со счёта заказчика предоплаты, соответственно размера sum и за заказ с id ID\_order; Этим занимается Обработчик денежных операций, ему выделены все функции, связанные с денежными операциями в системе.

*CreateLine()* – создаёт в БД запись об списании предоплаты.

Бд одна, в ней содержатся информация о заказах, в том числе и инфо о внесённой предоплате.