# 1.1. ЭСКИЗНЫЙ ПРОЕКТ

# 1.1.1 ПРЕЗЕНТАЦИЯ ИДЕИ ПРОЕКТА

## НАЗВАНИЕ ПРОЕКТА

BuyMyPoem - проект, направленный на реализацию взаимодействия авторов литературных произведений с читателями. Он упростит процесс покупки и продажи результатов творческой деятельности, позволит выставлять любым писателям свои работы для просмотра и оценки сообществу, а также поможет им разрабатывать новые проекты с помощью системы антиплагиата, черновиков и других специальных возможностей.

## ЦЕЛЬ И ЗАДАЧИ

Разработать программную систему целью которой является хранение текстовых данных и автоматизация процесса продажи и покупки результатов творческой деятельности, таким образом чтобы любой желающий мог просматривать выгруженные работы в режиме онлайн, взаимодействовать с другими пользователями, организовывая сделки, и загружать собственные работы.

Задачи, которые необходимо решить для достижения поставленной цели:

* организовать структуру веб-приложения и базы данных таким образом, чтобы избежать ошибок на низком уровне в ходе пользования системой
* добавить в веб-приложение функции многопоточности
* разработать систему регистрации, идентификации, аутентификации и авторизации в веб-приложении
* разработать удобный пользовательский интерфейс

## СПИСОК АНАЛОГОВ

* Сервис интернет-рекрумента - headhunter.ru
* Архив хранения текстовых данных, размещаемых пользователями на безвозмездной основе – ficbook.ru
* Сервис проверки текста на уникальность – antiplagiat.ru

## РОЛИ КОМАНДЫ. РАСПРЕДЕЛЕНИЕ ЗАДАЧ ПО ИСПОЛНИТЕЛЯМ.

(на этап проектирования)

Журавлев Н.М. – Анализ предметной области, создание диаграммы прецедентов и диаграммы состояний.  
Хлызова В.Г. – Анализ предметной области, создание эскиза интерфейса, создание диаграммы классов.

# 1.1.2. КАЛЕНДАРНЫЙ ПЛАН ПРОЕКТА

## СПИСОК ОСНОВНЫХ ЗАДАЧ ПО СРОКАМ ВЫПОЛНЕНИЯ

пункт 1.1.1. Презентация идеи проекта: deadline - 01.11.2019

пункт 1.1.2. Календарный план проекта: deadline - 08.11.2019

пункт 1.1.3. State of art: deadline - 08.11.2019

пункт 1.1.4. Анализ предметной области: deadline - 08.11.2019

пункт 1.1.5. Макет программной системы: deadline - 15.11.2019

# 1.1.3. STATE OF ART

## ОБЗОР АКТУАЛЬНЫХ ТЕХНОЛОГИЙ. АНАЛИЗ АНАЛОГОВ

За пример взята популярная система проверки текста на уникальность “Антиплагиат”, аналог которой предстоит реализовать в программной системе.

Извлечение текста

Система поддерживает возможность загрузки документов в docx, doc, txt, pdf, rtf, odt, html, pptx и еще нескольких форматах. Вместе с текстом система извлекает и позиции слов на страницах, чтобы в дальнейшем иметь возможность показывать пользователям разметку отчета о заимствовании на самом документе. Одним из способом извлечения текста является OCR. Текст печатается на виртуальном принтере, а потом распознается его. Также тексты могут извлекаться с помощью IFilter'ов.

Поиск заимствований

1. Разбиваем предложения на слова, выкидываем числа, знаки препинания, стоп-слова. Лемматизируем (приводим в нормальную форму) все слова.
2. Превращаем слова в целые числа путем хеширования, получаем массив чисел.
3. Берем первую тройку хешей, затем 2, 3, 4-ый хеш, затем 3, 4, 5-ый и так до конца массива хешей. Это и есть шинглы — черепички. Свое название такой способ получил из-за такого черепичного перекрытия наборов. Каждую черепичку сливаем в один объект и хешируем еще раз.
4. Сортируем полученные числа, получаем упорядоченный массив целых чисел. Это и есть основа для поиска.

Теперь для поиска нужна функция, которая по списку хешей превращает документы, отранжированные по убыванию числа совпавших хешей, в документ-источник. Для того, чтобы быстро находить такой набор, нужен обратный индекс, который по хешу возвращает список документов, в которых этот хеш есть. Реализована гигантская хеш-таблица, которая хранится на ssd, а не в памяти. Поиск по индексу занимает малую часть времени от всего цикла обработки документа. Проходит поиск так:

Этап 1. Поиск по индексу

Для каждого хеша текста-запроса получается список идентификаторов документов-источников, в которых он встречается. Дальше ранжируется список идентификаторов документов-источников по числу встретившихся хешей из текста-запроса. Получается ранжированный список документов-кандидатов на источник заимствования.

Этап 2. Построение ревизии

Для большого текста-запроса кандидатов может быть порядка 10 тыс. Это много. Берется первый документ-источник, делается сравнение с текстом-запросом и исключается из всех остальных кандидатов те хеши, которые уже были в этом первом документе. Удаляются из списка кандидатов те, у которых остался ноль хешей, кандидатов пересортируются по новому числу хешей. Берется первый документ из нового списка, сравнивается с текстом-источником, удаляются хеши, удаляются нулевые кандидаты, кандидаты пересортируются. Повторяем цикл 15-20 раз, чтобы список иссяк или в нем остались только те документы, у которых есть совпадение по нескольким хешам.

Использование хешей слов позволяет нам проводить операции сравнения быстрее, экономить на памяти и хранить не тексты документов-источников, а их цифровые слепки, полученные при индексации, тем самым не нарушая авторских прав. Выделение конкретных фрагментов заимствования делается с помощью суффиксного дерева.

В результате проверки одним модулем поиска получается ревизия, в которой находится список источников, их метаданные и координаты блоков заимствований относительно текста-запроса.

Сборка отчета

Здесь используется подход аналогичный подготовке одной ревизии, с учетом того, что совпадения могут быть двух видов: заимствования и цитирования. В результате отчета формируется два показателя для каждого источника:

Доля в отчете — отношение объема заимствований, которое учитывается из данного источника, к общему объему документа. Если один и тот же текст был найден в нескольких источниках, то учитывается он только в одном из них. В сумме дает процент заимствований и цитирований.

Доля в тексте — отношение объема, заимствованного из данного источника текста к общему объему документа. Доли в тексте по источникам суммировать нет смысла. Данный показатель не изменяется при изменении отчета.

Индексация интернета

Интернет индексируется направлено — поиск больших скоплени1 научных текстов, рефератов, статей, диссертаций, научных журналов и т.п. Индексация происходит так:

1. Алгоритм загружает документы с разумной нагрузкой на каждый хост (в работе одновременно сотни сайтов);
2. Алгоритм передает документ и его метаданные в очередь на обработку, из документа извлекается текст;
3. Текст анализируется на «качество» — добавляются сюда простые эвристики на объем и определяется качество пришедшего текста
4. Качественный текст проходит дальше и превращается в хеши. Хеши и метаданные отправляются в основной индекс интернета;
5. Сравниваем пришедший текст с ранее проиндексированными текстами. Новичок добавляется, только если он реально новый, т.е. 90% его хешей не содержится целиком в каком-то другом уже проиндексированном тексте. Если же документ уже есть, он добавляется в атрибуты архива.

# 1.1.4. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

## ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ

Предметная область подразумевает хранилище текстов с реализацией функции автоматизации процессов обмена результатами творческой деятельности. (Процесс купли-продажи литературных произведений).

## СЛОВАРЬ ПРЕДМЕТНОЙ ОБЛАСТИ

Author(Автор) – пользователь системы, просматривающий и добавляющий работы, обладает атрибутами логин, пароль, фио, дата рождения, личная информация а также рейтинг оценки работ

Customer(Заказчик) – пользователь системы просматривающий и заказывающий работы, обладает атрибутами логин, пароль, фио, дата рождения, личная информация.

PaymentResourse(Платежное средство) – сущноть, содержащая данные о платежных ресурсах пользователя, содержит атрибуты номер карты, номер телефона.

Order(Заказ) – сущность, описывающая и реализующая процесс обмена выполненной работы на денежные средства, содержит информацию об авторе, заказчике, заявке, на основе которой он создан, работе, и состоянии.

Request(Заявка) – сущность, описывающая требования заказчика к будущей сделки, содержит информацию о жанре, теме, дедлайне, стоимости, заказчике и об авторах, откликнувшихся на эту заявку

Comment(Комментарий) – сущность, описывающая процесс взаимодействия и общения между автором и заказчиком в процессе бизнес-логики, а также сообщение пользователя привязанное к произведению. Содержит информацию о произведении, пользователе и содержимом комментария.

Composition(Произведение) – сущность, описывающая свойства результата творческой деятельности. Содержит информацию об авторе, названии, жанре, тегах, количестве лайков и дизлайков, статусе и самом содержимом произведения

Status(Статус) – статус произведения. Может содержать одно из следующих значений (черновик, открытая работа, заказная работа)

Genre(Жанр) – жанр произведения. Может содержать одно из следующих значений (комедия, трагедия, детектив и т.д.)

Tag(Тег) - абстракция, описывающая кодовые слова для краткого и общего описания и поиска произведения.

Type(Тип) – тип произведения произведения. Может содержать одно из следующих значений (стих, рассказ, пьесса, басня и т.д.)

## ПОЛЬЗОВАТЕЛИ СИСТЕМЫ И РОЛИ

Гость – неавторизованный пользователь, может просматривать авторизированных пользователей, произведения авторов и заявки. Имеет возможность авторизоваться или создать учётную запись.

Авторизованный пользователь:

Заказчик – пользователь, авторизованный в системе. Представляет физическое лицо, которое выступает в качестве покупателя результата творческой деятельности

Автор - пользователь, авторизованный в системе. Представляет физическое лицо, которое выступает в качестве продавца результата творческой деятельности

Администратор - пользователь, авторизованный в системе. Представляет физическое лицо, которое управляет жизнедеятельностью системы, может вносить изменения.

## ПРЕЦЕДЕНТЫ



СПЕЦИФИКАЦИЯ ПРЕЦЕДЕНТОВ

**1. “Подтвердить выполнение заказа”**

**Предусловие.** Автор откликнулся на заявку, заказчик выбрал автора, автор выполнил работу.

**Актеры:** “Заказчик”

**Основной поток.** Актер просматривает работу выполненную автором и, если его устраивает результат, он нажимает на кнопку подтверждения выполнения.

**Альтернативный поток.** Актера не устраивает результат выполнения работы после предварительного просмотра. Он нажимает на кнопку отправки заказа на редактирование, после чего ему предлагается написать комментарий к работе.

**Постусловие.** Актер начинает выполнение прецедента “Оплатить заказ”.

**2. “Оценить работу”**

**Предусловие:** Актер просматривает опубликованную работу.

**Актеры:** “Автор”, “Заказчик”

**Основной поток:** Актер нажимает на кнопку “Понравилось”, количество отметок “Понравилось” увеличивается на 1; или же актер нажимает на кнопку “Не понравилось”, количество отметок “Не понравилось” увеличивается на 1.

**Альтернативный поток:** Источник данных временно не доступен, Актёр получает уведомление с сообщением о возникших проблемах.

**Постусловие:** На экран выводится список всех опубликованных работ.

3.**“Посмотреть список заявок”**

**Предусловие: -**

**Актеры:** “Автор”

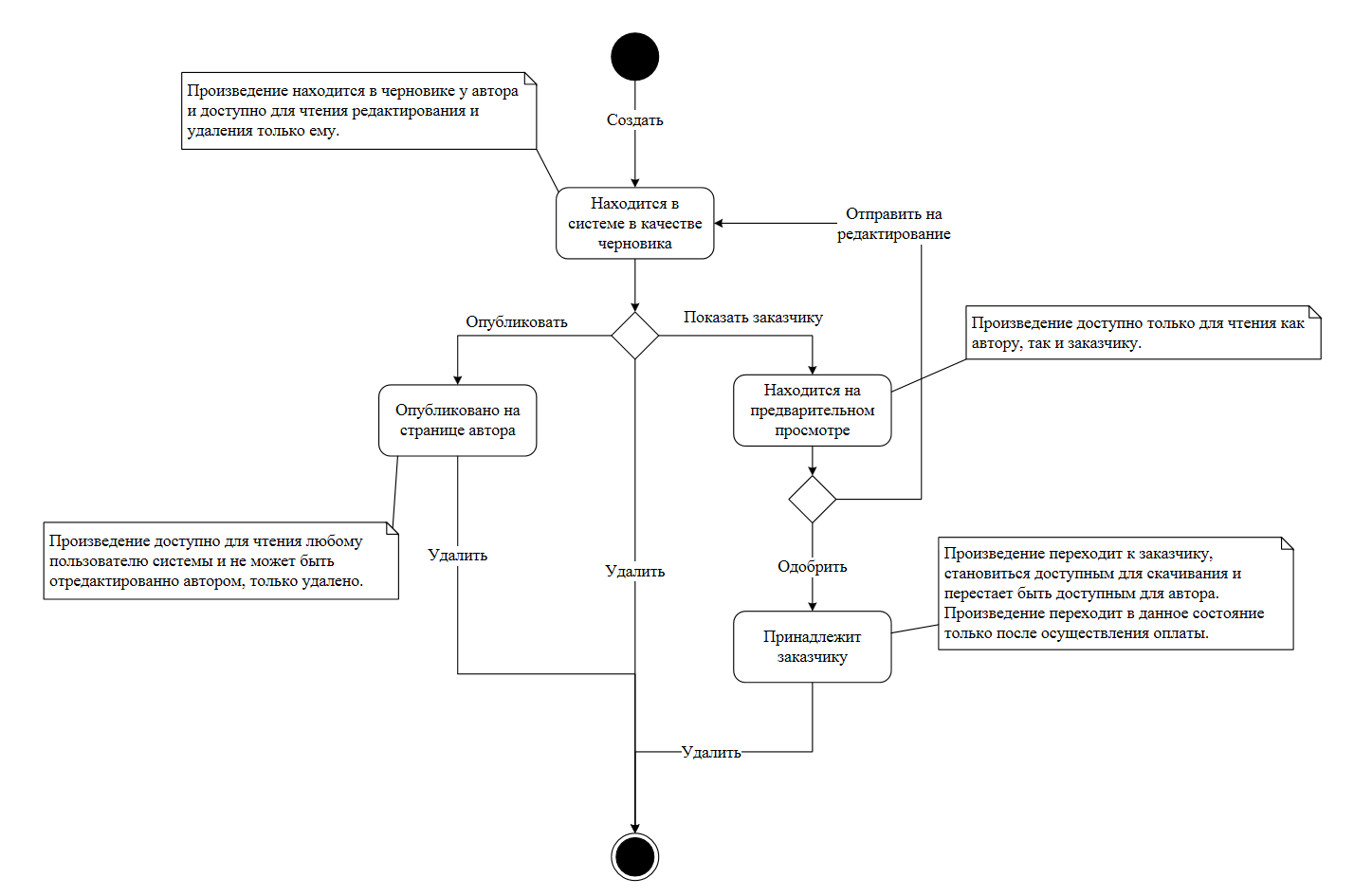
**Основной поток:** Актёр с главной страницы переходит по кнопке “Список заявок” на страницу со списком актуальных заявок от Заказчиков.

**Альтернативный поток:** Источник данных временно не доступен, Актёр получает уведомление с сообщением о возникших проблемах.

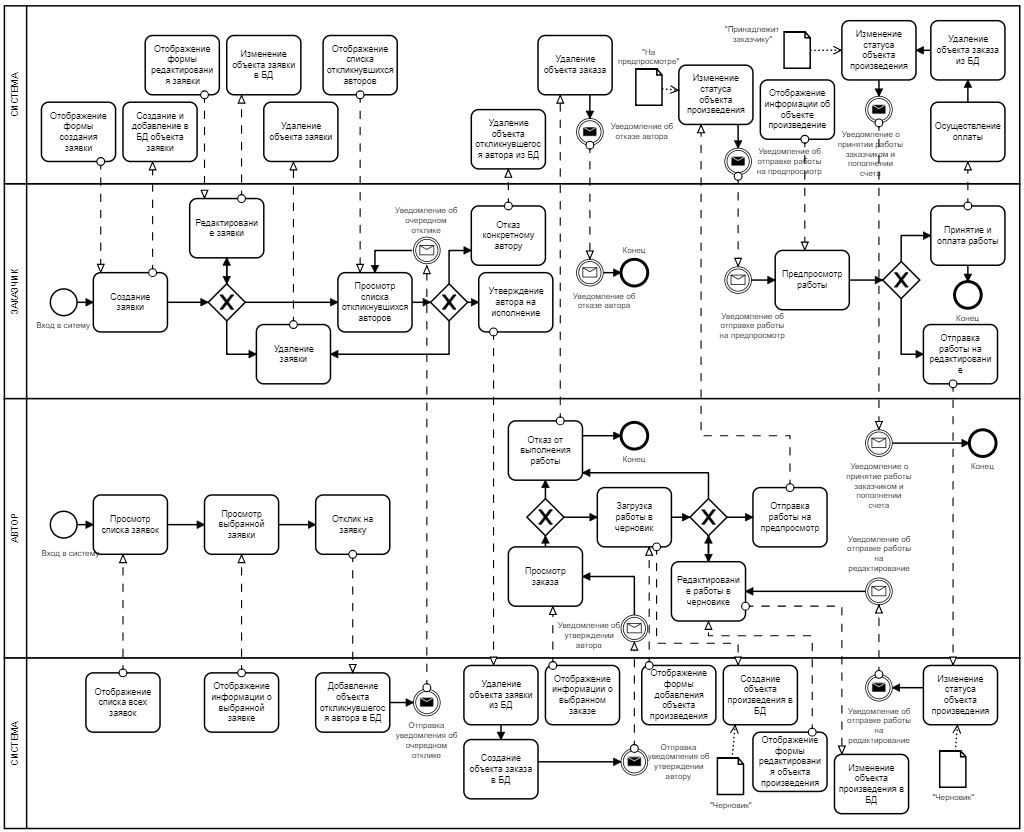
**Постусловие:** Актёр пролистывает список всех заявок и выбирает понравившуюся заявку либо осуществляет поиск по заявкам и ищет подходящую.

## ОПИСАНИЕ ДОКУМЕНТОВ/СУЩНОСТЕЙ ПРЕДМЕТНОЙ ОБЛАСТИ

1. Концептуальная диаграмма классов (UML Class Diagram)

2. Диаграмма состояний произведения 

## АНАЛИЗ БИЗНЕС-ПРОЦЕССОВ



## ФУНКЦИОНАЛЬНЫЕ ТРЕБОВАНИЯ К СИСТЕМЕ

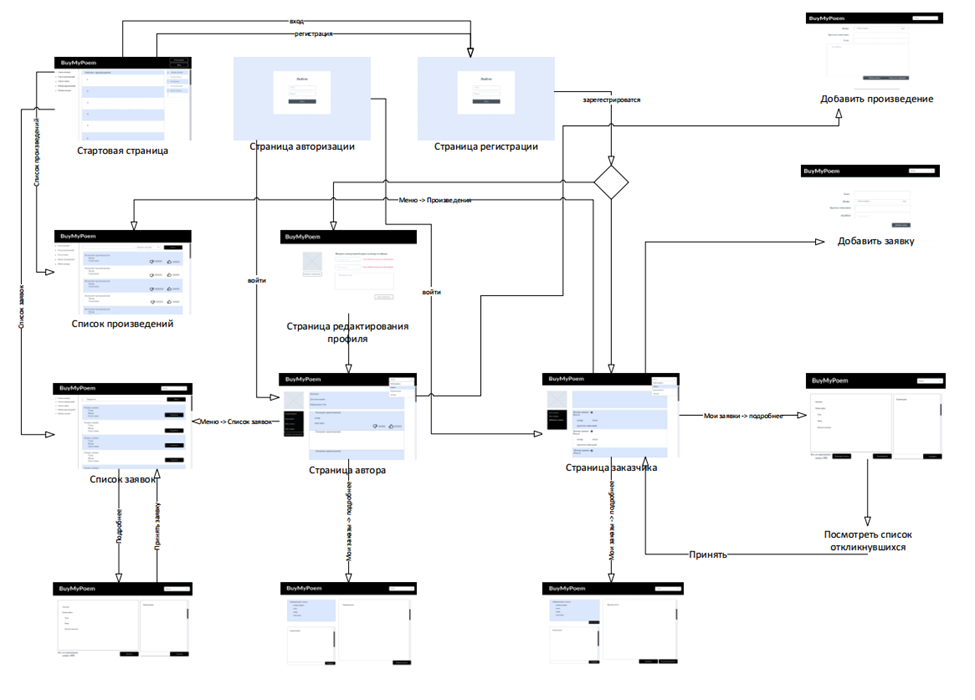
* Система должна обеспечивать возможность регистрации, авторизации, аутентификации и идентификации пользователей, а также хранение, изменение, добавление и удаление их личных данных.
* Система должна обеспечивать хранение текстовых данных.
* Система должна обеспечивать организацию бизнес-логики для процесса купли и продажи результата творческой деятельности, гарантировать конфиденциальность и надежность выполнения этого процесса.
* Система должна включать в себя функцию проверки текста на уникальность.

## НЕФУНКЦИОНАЛЬНЫЕ ТРЕБОВАНИЯ К СИСТЕМЕ

* Система должна быть реализована в виде web-приложения
* Система должна предусматривать базовую защиту от SQL-инъекций
* Система должна иметь простой и понятный пользователю интерфейс
* Система должна предусматривать валидацию данных.

# 1.1.5. МАКЕТ ПРОГРАММНОЙ СИСТЕМЫ

## СТРУКТУРА ИНТЕРФЕЙСА, НАВИГАЦИЯ



## ЭСКИЗЫ ИНТЕРФЕЙСА ПОЛЬЗОВАТЕЛЯ

