**Отчет**

**Аукцион**

## Задание

Реализовать работу электронного аукциона.

## Роли

В системе существует 3 вида пользователей:

* Покупатель - клиент, который хочет приобрести какой-либо лот.
* Администратор - контролирует аукцион.
* Продавец - клиент, который является поставщиком новых товаров для аукциона.

## Функциональные требования

**Для покупателя:**

* Регистрация и авторизация в системе.
* Покупка лота.
* Участие в аукционе в качестве продавца.
* Поиск лота по различным категориям.
* Возможность выставления оценки продавцу.

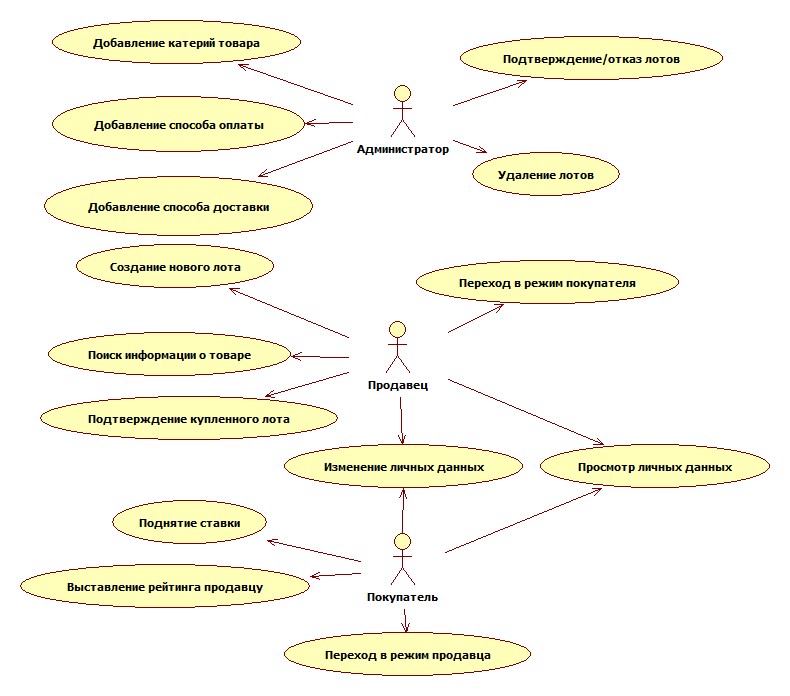
**Для продавца:**

* Регистрация и авторизация в системе.
* Создание заявок на добавление нового лота.
* Задание желаемой стоимости лота.
* Возможность участвовать в аукционе в качестве покупателя..

**Для администратора:**

* Обработка заявок от продавца.
* Обеспечивать защиту продавца и покупателя.
* Редактирование списка товаров.

## Варианты использования



## Подробное описание всех вариантов использования

### Варианты использования для "Администратора"

* Подтверждение заявки о добавлении нового лота.

Администратор может одобрить или отклонить заявку в зависимости от внутренних правил магазина.

* Обеспечение надежной и бесперебойной работы аукциона

Администратор может следить за ходом аукциона. Если лот, предоставленный на аукционе, нарушает правила, администратор вправе снять слот.,

### Варианты использования для "Покупателя"

Регистрация

* Пользователь заполняет регистрационную форму  (логин и пароль).
* Если логин еще не зарегистрирован, то регистрация проходит успешно, иначе пользователю предлагается ввести другой логин.

Покупка лота

* Пользователь ищет необходимый лот в каталоге.
* Если соблюдены все требования для участия в аукционе, т.е. лот успешно найден и пользователь авторизован, то при нажатии на кнопку "Установить ставку" , всплывает окошко, где нужно ввести сумму. При необходимости пользователь может установить несколько ставок.
* Если пользователь не прошел авторизацию. Появляется окно с предложением ввести логин и пароль или зарегистрироваться.
* Если Аукцион был выигран, то пользователь связывается с продавцом и договаривается об оплате и доставке.

Написание комментария и выставление оценки

* После успешной покупки пользователь может зайти на страницу продавца и выставить оценку

Продажа лота

* Пользователь может выставить свой лот на аукцион, после этого пользователю устанавливаются права «Продавца»

### Варианты использования для "Продавца"

Регистрация

* Пользователь заполняет регистрационную форму (логин и пароль).
* Если логин еще не зарегистрирован, то регистрация проходит успешно, иначе пользователю предлагается ввести другой логин.

Добавление лота

* Продавца в специальной форме заполняет заявку и устанавливает минимальную стоимость.
* Если продавец авторизован, то заявка встает в очередь на подтверждение от администратора.
* Администратор одобряет заявку, лот добавляется в магазин.
* Если администратор отклоняет заявку, продавцу отправляется письмо с объяснениями причины отказа.

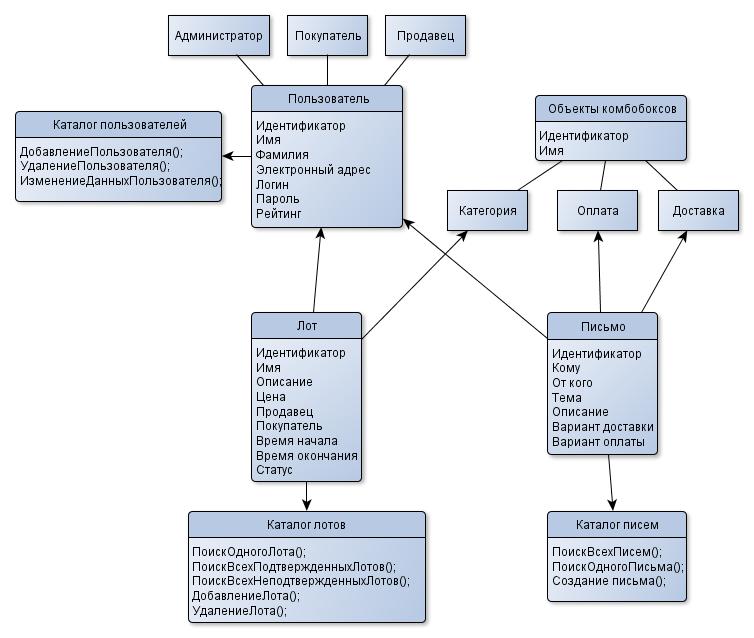
Задание стоимости лота

Если продавцу утвердили заявку, то лот появляется в каталоге.

Когда лот будет куплен, продавец связывается с покупателем и договаривается об оплате и доставке.

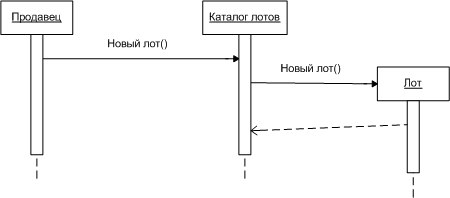
Если покупатель не выполняет требования, то продавец обращается к Администратору для дальнейших действий.

## Разработка статической объектной модели предметной области

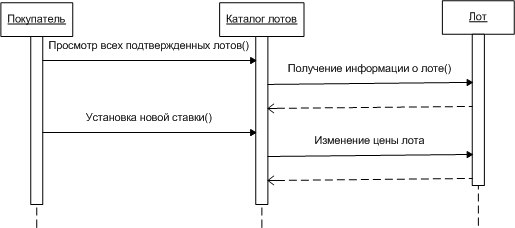


## Разработка динамической объектной модели предметной области

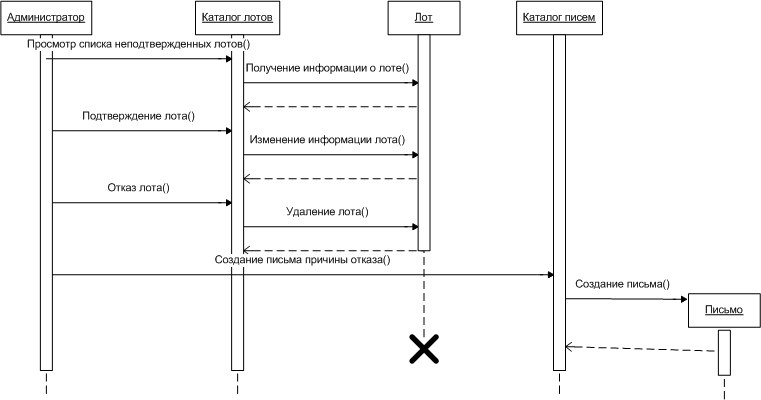
**Добавление нового лота:**



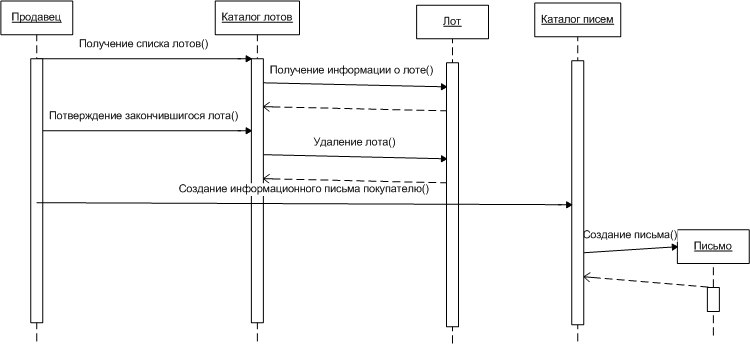
**Установление новой ставки:**



**Администрирование лотов:**



**Подтверждение окончания аукциона для определенного лота:**



**Просмотр писем:**



## Проектирование слоя бизнес-логики

При проектирование слоя бизнес-логики был выбран шаблон Domain Model(модель области определения), потому что он образует сеть взаимосвязанных объектов, в которой каждый объект представляет собой отдельно значащую сущность, поэтому он хорошо подходит для отображения объектно-ориентированной модели приложения. Но все же паттерн обладает несколькими недостатками, а именно сложностью взаимодействия с БД и сложностью реализации, которые необходимо учитывать при разработке.

## Реализация слоя бизнес-логики

Классы, отвечающие за бизнес логику, находятся в пакете business:

1. User – класс, хранящий в себе информацию о пользователях.
2. Administrator – класс, осуществляющий сущность администратора.
3. Seller – класс, осуществляющий сущность продавца.
4. Customer – класс, осуществляющий сущность покупателя.
5. SelectObject – класс, хранящий информацию о выбираемых элементах.
6. Delivery – класс, хранящий способ доставки.
7. Payment – класс, хранящий способ оплаты.
8. Category – класс, хранящий категорию товаров.
9. Lot – класс, хранящий в себе информацию о лоте.
10. Letter – класс, хранящий информацию о письме.

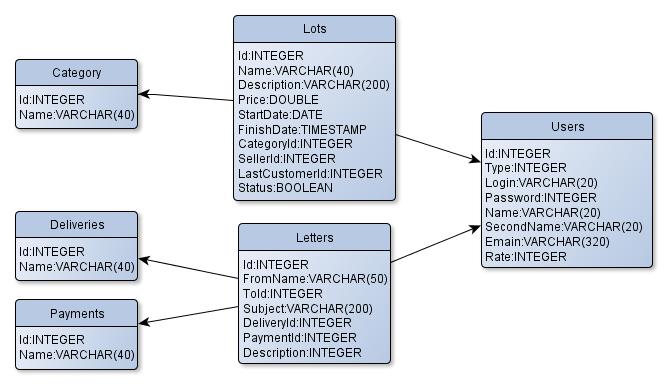
## Проектирование слоя источников данных

В качестве шаблона для реализации слоя источников данных был выбран шаблон Data Mapper по следующим причинам:

1. Он хорошо сочетается с шаблоном Domain Model
2. Поля объектов бизнес логики совпадают с колонками в таблице БД
3. Обмен с БД достаточно прост.
4. Шаблон позволяет отделить слой доступа к данным от слоя бизнес-логики.

Для реализации наследования использовался шаблон Single Table Inheritance. В данном случае этот шаблон подходит лучше всего, так как иерархия наследования присутствует в классах Customer, Seller и Administrator, но их поля полностью совпадают. Для реализации данного шаблона было введено дополнительное поле Type.

**Схема разработанной базы данных:**



## Реализация слоя источников данных

В качестве БД использовался Apache Derby версии 10.9.1.0.

Разработанные классы можно найти в пакете db.

* ConnectionManager - интерфейс, описывающий создание соединения с БД.
* DerbyConnectionManager - конкретный класс для доступа к рабочей БД.
* Mapper - базовый класс для всех мапперов.
* UserMapper - маппер для класса User.
* CategoryMapper – маппер для класса Category.
* DeliveryMapper – маппер для класса Delivery.
* LetterMapper – маппер для класса Letter.
* LotMapper – маппер для класса Lot.
* PaymentMapper – маппер для класса Payment.

## Проектирование и реализация сервисного слоя

Сервисный слой инкапсулирует бизнес-логику приложения, образует набор доступных операций и управляет ответом приложения в каждой операции. Разработанные классы можно найти в пакетах service и httpservice.

* CategoryCatalog – для получения информации о категориях товара.
* DeliveryCatalog – для получения информации о способах доставки.
* LetterCatalog – для получения информации о письмах.
* LotCatalog – для получения информации о лотах.
* PaymentCatalog – для получения информации о способах платежа.
* UserCatalog – для получения информации о пользователях.
* HttpClientManager – для получения xml файла от сервиса yandex.xml
* YandexXMLParser – для парсинга полученного xml файла.

Внешним сервисом был выбран yandex.xml, потому что все сервисы, которые дают доступ к данным о товаре открыты только для больших корпораций, а yandex.xml открыт для всех пользователей.

## Проектирование и реализация слоя представления

Реализация графического интерфейса с использованием библиотеки Swing находится в пакете gui.

* AuthFrame - окно авторизации.
* RegistrationFrame - окно регистрации.
* AddBidFrame – окно установки ставки.
* EditPersonDataFrame – окно изменения личных данных пользователя.
* LetterListFrame – окно списка писем.
* LetterFrame – окно написания письма.
* LetterViewFrame – окно просмотра письма.
* MainAdminFrame – главное администраторское окно.
* MainCustomerFrame – главное окно покупателя.
* MainSellerFrame – главное окно продавца.
* ManageLotFrame – окно администрирования лотов.
* PrivateOfficeFrame – окно личной информации.
* RateFrame – окно выставления рейтинга.

# Комплексное тестирование системы

Тестирование проводилось как автоматически, при помощи Unit-тестов, так и вручную.

## Запуск тестов:

-------------------------------------------------------

T E S T S

-------------------------------------------------------

Running db.CategoryMapperTest

Tests run: 2, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 7.24 sec

Running db.DeliveryMapperTest

Tests run: 2, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.304 sec

Running db.LetterMapperTest

Tests run: 2, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.272 sec

Running db.LotMapperTest

Tests run: 4, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.329 sec

Running db.PaymentMapperTest

Tests run: 2, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.165 sec

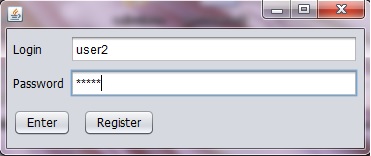
Running db.UserMapperTest

Tests run: 3, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.176 sec

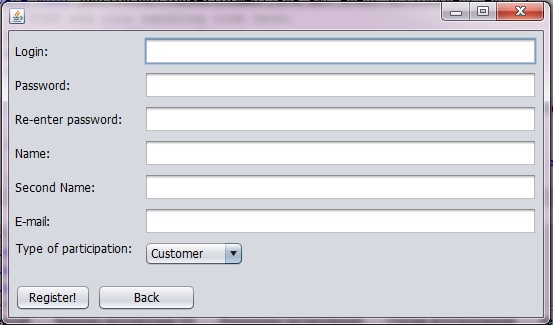
Results :

Tests run: 15, Failures: 0, Errors: 0, Skipped: 0

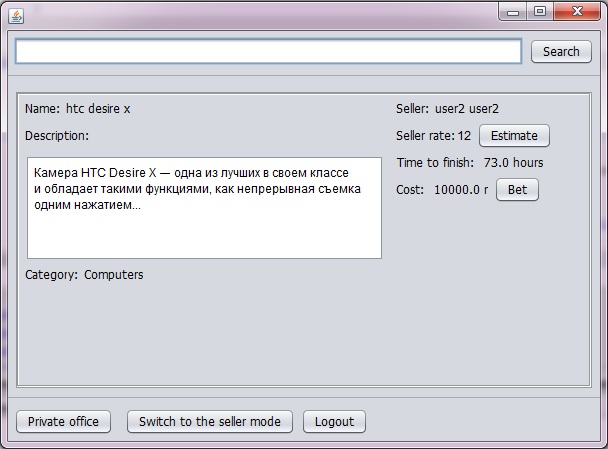
**Окно входа в приложение:**



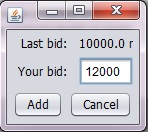
**Окно регистрации:**



**Окно покупки лотов:**

о

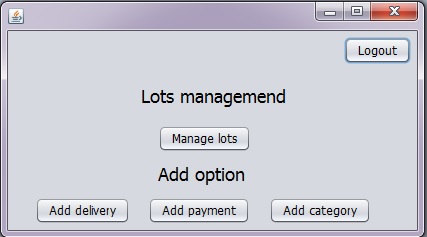
**Окно выставления ставок:**



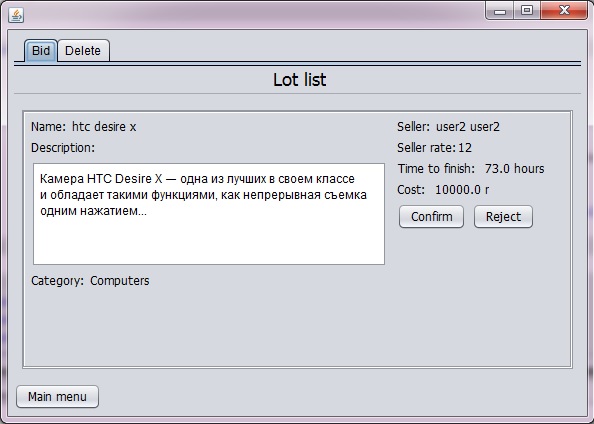
**Окно выставления рейтинга:**



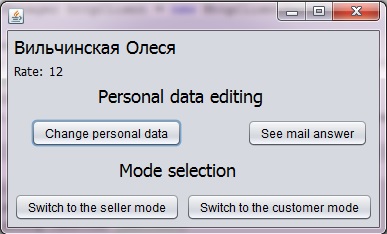
**Главное окно администратора:**



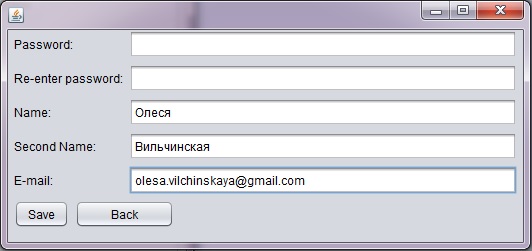
**Окно администрирование лотов:**



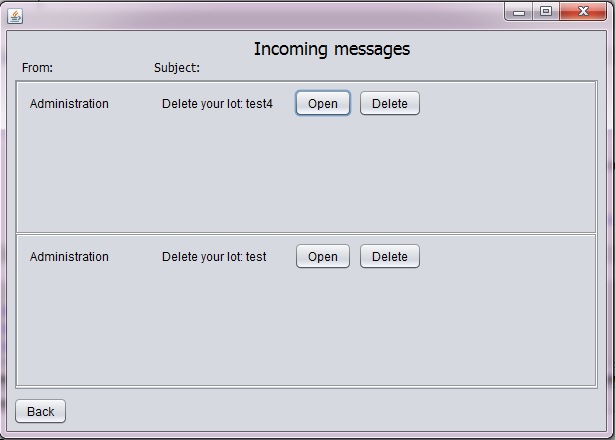
**Окно личных данных:**



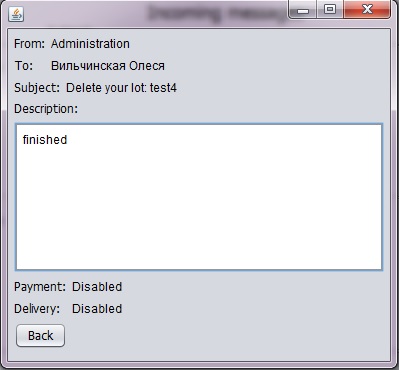
**Окно редактирования личных данных:**



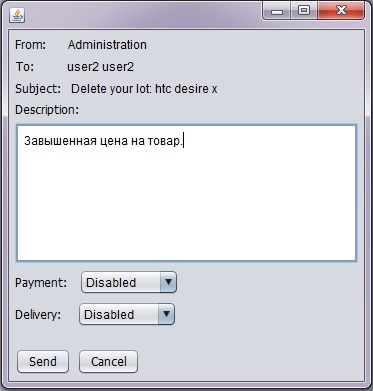
**Окно просмотра писем:**



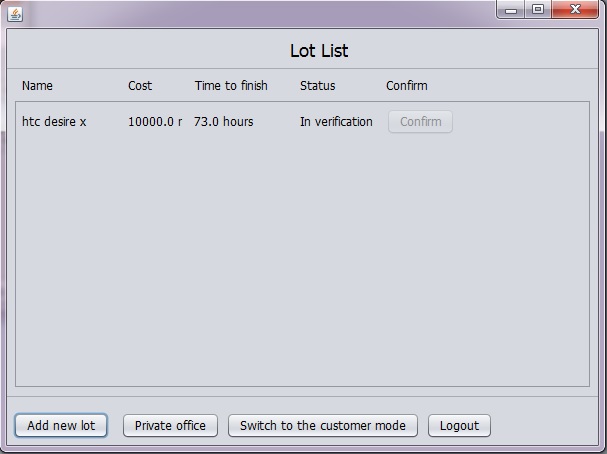
**Окно просмотра письма:**



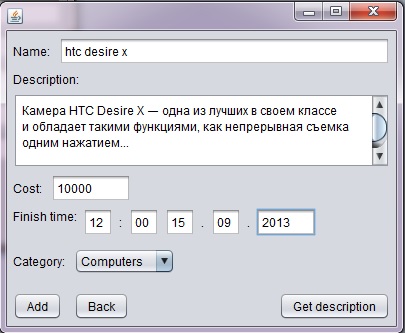
**Окно создания письма:**



**Главное окно продавца:**



**Окно добавления нового лота:**



## Выводы

В результате выполнения курсовой работы был реализован проект “Аукцион”. Были изучены основные архитектурные шаблоны проектирования, получены навыки по их использованию. Также были рассмотрены типовые решения организации бизнес-логики, слоя доступа к данным, сервисного слоя.