МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования «САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

КАФЕДРА КОМПЬЮТЕНЫЙ ТЕХНОЛОГИЙ И ПРОГРАММНОЙ ИНЖЕНЕРИИ

КУРСОВОЙ ПРОЕКТ  
ЗАЩИЩЕН С ОЦЕНКОЙ

РУКОВОДИТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Ст. преподаватель |  |  |  | Е.О. Шумова |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ПОЯСНИТЕЛЬНАЯ ЗАПИСКА К КУРСОВОМУ ПРОЕКТУ |
| Разработка приложения для организации объектов при заданных критериях |
| по дисциплине: ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | 4932 |  | 04.10.2021 |  | Белов А.И. |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2021

Оглавление

[1. Постановка задачи 1](#_Toc86920269)

[1.1. Анализ предметной области 1](#_Toc86920270)

[1.2. Формулировка технического задания 1](#_Toc86920271)

[2. Проектирование классов 1](#_Toc86920272)

[2.1. Классы сущностей 1](#_Toc86920273)

[2.2. Управляющие классы 1](#_Toc86920274)

[2.3. Интерфейсные классы 1](#_Toc86920275)

[2.4. Используемые паттерны проектирования 1](#_Toc86920276)

[3. Разработка приложения 1](#_Toc86920277)

[3.1. Разработка интерфейса приложения 1](#_Toc86920278)

[3.2. Реализация классов 1](#_Toc86920279)

[3.3. Разработка тестового приложения 1](#_Toc86920280)

[4. Тестирование 1](#_Toc86920281)

# Постановка задачи

Вариант 20: Разработка системы классов для обеспечения работы аптеки

## Анализ предметной области

Предметная область: Аптека

Описание: Рассмотрим небольшую частную аптеку. Планируется разработать для данного предприятия информационную систему для управления работой аптеки. В аптеке работают четыре категории сотрудников: администраторы, менеджеры, кассиры и работники склада. Доступ к полному функционалу систему должны иметь только администраторы, остальные категории должны иметь доступ только к функционалу имеющему прямое отношение к их зоне ответственности.

На данном предприятии менеджеры заняты заполнением и ведением каталога продукции: созданием позиций в каталоге, удалением вышедшей из продажы продукции, созданием классификации продуктов. На сегодняшний день принята следующая классификация: все товары делятся на типы товаров, далее делятся по категориям и по форме выпуска. Так же товары классифицируются по брендам независимо от типа.

Работники склада должны иметь доступ к управлению складом. Оно производится не напрямую, а только через создание накладных при завозе или списании определенных товаров со склада. После завозе/списании новой продукции работник должен создать накладную, выбрать необходимый товар со склада или из каталога, указать количество товара и списать/записать товары на склад. Накладные должны сохраняться в системе без возможности редактирования и удаления.

Кассиры должны иметь доступ к управлению продажами, так же как и в случае с работниками склада, оно производится только путем создания продаж. При совершении покупки кассиру необходимо выбрать все необходимые товары со склада, указать их количество, указать дату покупки и создать продажу. Продажи должны сохраняться в системе без возможности редактирования и удаления.

Также необходим функционал просмотра статистики по продажам – количество продаж и их общая стоимость в среднем по месяцам и по временам года в зависимости от выбранного типа или категории товаров.

Объекты предметной области:

|  |  |  |
| --- | --- | --- |
| № | Наименование | Информация |
| 1 | Товар | Любые виды продукции |
| 2 | Тип товаров | Классификация продукции по типам |
| 3 | Категория товаров | Классификация продукции по категориям |
| 4 | Форма выпуска | Классификация продукции по форме выпуска |
| 5 | Бренд | Бренды продукции |
| 6 | Склад | Содержит информацию о продукции в магазине и ее количестве. |
| 7 | Накладная по складу | Содержит информацию о завозах/списании товаров |
| 8 | Продажа | Содержит информацию о всех произведенных продажах аптеки. |
| 9 | Каталог | Содержит информацию о всех видах продукции |

Субъекты предметной области:

|  |  |  |
| --- | --- | --- |
| № | Наименование | Информация |
| 1 | Администратор | Пользователь, имеющий доступ ко всему функционалу системы. |
| 2 | Менеджер | Пользователь, имеющий доступ к статистике, управлению каталогом, брендам, типам, категориям и товарам. |
| 3 | Работник склада | Пользователь, имеющий доступ только к управлению складом и накладными по складу |
| 4 | Кассир | Пользователь, имеющий доступ только к управлению продажами |

## Формулировка технического задания

**Цель проекта:** Необходимо создать информационную систему для управления аптекой.

**Архитектура**: в качестве архитектуры выбрана двухуровеная система, членами которой являются: Реляционная база данных (далее – «БД») и Desktop клиент (далее – «клиент»). Целевая операционная система – Windows 10.



**Применяемые технологии:**

**БД**:

Сервер: SQL Server 2019

Система управления: MS SQL 2019 Express

**Клиент**:

Язык программирования: C# 9.0.

Платформа разработки: .NET Framework 4.7.2, Windows Forms

IDE: Visual Studio 2019

*Дополнительные библиотеки*:

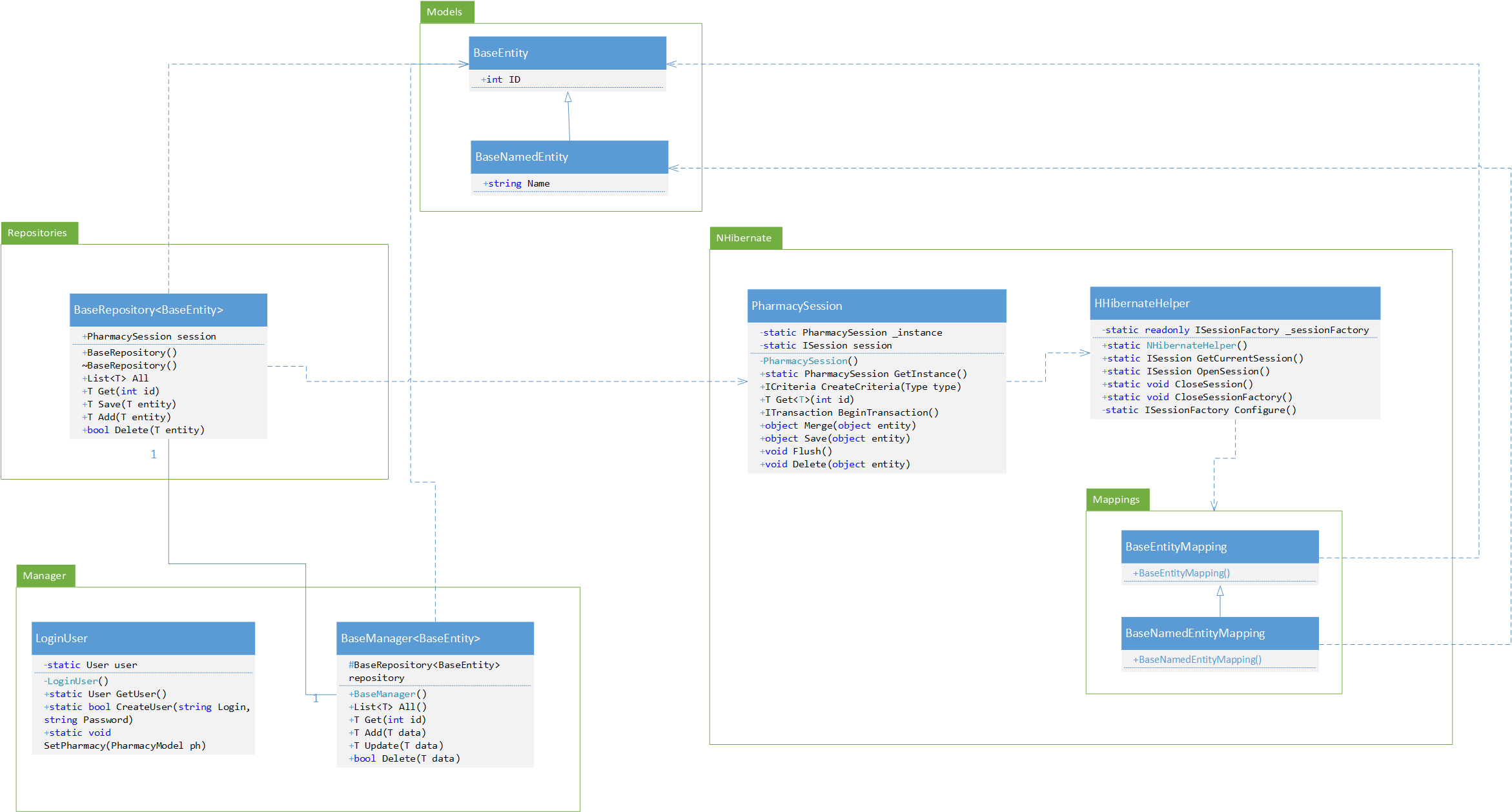
*ORM – библиотека: Nhibernate*

*UI - Набор элементов управления: DevExpress WinForms*

**Функицональные требования**:

|  |  |  |
| --- | --- | --- |
| № | Наименование | Описание |
| 1. | **Общее** |  |
| 1.1 | Авторизация | Реализовать вход в систему посредством ввода логина и пароля пользователя |
| 1.2 | Права доступа | Реализовать выдачу доступа к функционалу системы в зависимости от роли вошедшего в систему пользователя. |
| 2 | **Администрирование** | Доступно только пользователям с ролью «Администратор» |
| 2.1 | Управление пользователями | Просмотр, создание, редактирование, удаление пользователей из системы и выдача им необходимой роли. |
| 3 | **Каталог** | Доступно пользователям с ролью «Администратор» или «Менеджер» |
| 3.1 | Товары | Просмотр, создание, редактирование, удаление товаров из каталога, указание типа, категории, формы выпуска и бренда товара, запрет на удаление товаров находящихся на складе. |
| 3.2 | Типы товаров | Просмотр, создание, редактирование, удаление типов товаров из каталога, запрет на удаление типов при наличии товаров данного типа в каталоге. |
| 3.3 | Категории товаров | Просмотр, создание, редактирование, удаление категорий из каталога, указание типа товара, запрет на удаление категорий при наличии товаров данного типа в каталоге. |
| 3.4 | Формы выпуска товаров | Просмотр, создание, редактирование, удаление форм выпуска товров из каталога, указание типа товара, запрет на удаление форм выпуска при наличии товаров данного типа в каталоге. |
| 3.5 | Бренды | Просмотр, создание, редактирование, удаление брендов из каталога, запрет на удаление брендов при наличии товаров данного типа в каталоге. |
| 4 | **Статистика** | Доступно пользователям с ролью «Администратор» или «Менеджер» |
| 4.1 | Общая статистика | Просмотр средней статистики по продажам в виде стобчатого графика, возможность настроить статистику на отображение средних значений по временам года или по месяцам, на отображение значений: количества продаж, количества проданных товаров, общей цене проданных товаров, отфильтровать значения по типам и категориям. |
| 5 | **Склад** | Доступно пользователям с ролью «Администратор» или «Работник склада» |
| 5.1 | Посмотреть склад | Просмотр всех товаров на складе, их количества. Возможность сортировать позиции по всем полям. Возможность группировать позиции по категориям, брендам, формам выпуска. |
| 5.2 | Зачисление/списание товара | Создание накладной на зачисление/списание товара. Возможность выбрать товар со склада или из каталога, указать количество товара, дату и создать зачисление или списание товара. Реализовать проверку на списание большего количества товара, чем имеется на складе. |
| 5.3 | Отчетность | Просмотр всех зачислений и списаний товаров, возможность просмотра только зачислений или только списаний. Возможность группировать позиции по категориям, брендам, формам выпуска. Возможность сортировать позиции по всем полям. |
| 6 | **Касса** | Доступно пользователям с ролью «Администратор» или «Кассир» |
| 6.1 | Создать продажу | Возможность создать продажу, возможность заполнить товарами список продажи, выбрав их со склада и указав их количество. После заполнения списка указать дату и создать продажу. Реализовать проверку на продажу большего количества товара чем имеется на складе. |
| 6.2 | Отчетность | Просмотр всех продаж. Реализовать два списка, в левом отображаются все продажи с общей стоимостью чека и датой, имеется возможность сортировать продажи по всем полям и группировать по дате. При щелчке на продажу из левого списка в правом отображать список всех товаров из данной продажи с наименованием продукта, количеством проданных единиц и ценой, возможность сортировать по всем полям. |

# Проектирование классов



## Классы сущностей

Сущности представлены классами с названиями \*Model, где \*-наименование сущности. Модели не несут в себе никакой логики кроме гетеров и сетеров для полей и предназначены только для хранения данных.

Все классы сущностей будут наследоваться от базовой модели BaseEntity, которая будет использоваться в шаблонных классах в роли «любой сущности»

## Управляющие классы

Управляющие классы представлены маппингами, репозиториями и менеджерами.

Маппинг – класс с названием \*Mapping, где \*-наименование конкретной сущности. Маппинг представляет собой класс содержащий информацию о том, как Nhibernate должен переконвертировать данные из БД в необходимые системе объекты классов.

Все классы маппинга будут наследоваться от базового маппинга BaseEntityMapping<T> where T:\*

Репозиторий – класс с названием \*Repository, где \*-наименование конкретной сущности. Репозиторий представляет собой класс реализующий логику работы с БД и содержит операции по сохранению, удалению, редактированию объектов и т.д.

Все классы репозитории будут наследоваться от базового репозитория BaseRepository<T> where T:\*

Менеджер – класс с названием \*Manager, где \*-наименование конкретной сущности.

Менеджер представляет собой класс реализующий бизнес-логику работы с сущностью и содержит как классические операции добавления, редактирования и удаления объектов, так и операции имеющие отношение к функционалу предметной области.

Все классы менеджеры будут наследоваться от базового менеджера BaseManager<T> where T:\*

Так же планируется классы:

NhibernateHelper – класс выделяющий необходимый для работы в данной системе функционал Nhibernate.

PharmacySession – класс хранящий в себе открытую сессию работы с бд и функции управления ею.

LoginUser – класс реализующий логику авторизации в системе.

## Интерфейсные классы

Условно, интерфейсные классы в данной системе можно разделить на два вида: UserControls и Forms

UserControls – классы с названием uc\*, где \*-название конкретной сущности, представляют собой таблицу или несколько таблиц, содержаших в себе различные списки объектов, данные формы практически не реализуют никакого функционала, кроме фильтрации, поиска и группировки данных. Особенность данных классов в том, что они не являются отдельным окном, а отрисовываются прямо на основной форме.

Forms – классы с названием \*Form, где \*-название конкретной сущности, представляют собой небольшую форму с полями данных, необходимую для работы с одним определенным объектом системы. Отрисовываются отдельно от основной формы.

Отдельно реализуются классы:

LoginForm – представляет собой небольшую форму для авторизации в системе, первая форма которую видит пользователь.

MainForm – представляет собой крупную форму, предоставляющую пользователю быстрый доступ ко всему основному функицоналу системы с помощью выдвигающейся панели инструментов и ссылок. На данной форме пользователь проводит большую часть всего времени и именно на ней отрисовываются UserControls

## Используемые паттерны проектирования

NhibernateHelper – Фасад для работы с библиотекой Nhibernate

PharmacySession – Синглтон для сессии Nhibernate, необходим для избежания случайного создания нескольких паралелльных сессий работы с БД.

BaseRepository<T>, BaseManager<T>, BaseEntityMapping<T> - Шаблонные классы для логики сущностей.

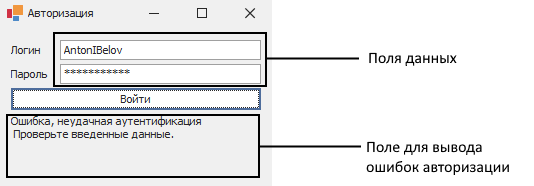
В большинстве сущностей используется паттерн Мост для сохранения структуры БД и избежания дополнительной нагрузки на ПК.

PharmacyModel – компоновщик для всех данных имеющихся в системе.

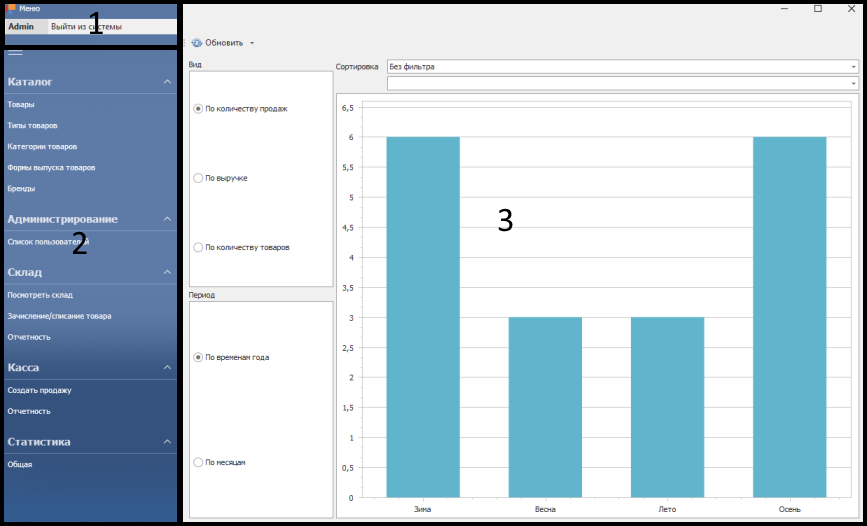
На форме MainForm реализован паттерн Стратегия, позволяющий не открывать для каждого функционала новые формы, а заменять его прямо на главной форме.

# Разработка приложения

## Разработка интерфейса приложения

Первое что видит пользователь при запуске системы – окно авторизации, оно достаточно легкое и не нагружено информацией, содержит два поля – логин и пароль, кнопку авторизации и поле для вывода ошибок в случае их появления. 

При удачной авторизации пользователя переключает на главное меню:



Главное меню можно условно разделить на три части:

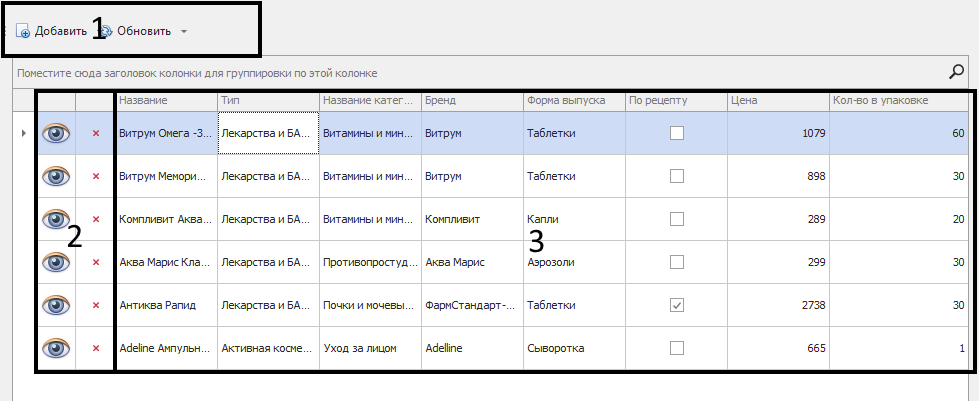
1 – Информация о пользователе использующем систему и кнопка выхода из системы, при нажатии на которую главное меню будет закрыто, а пользователю нужно будет заново пройти авторизацию.

2 – Боковое меню предоставляющее доступ к функционалу системы. Весь функционал разбит на пять раскрывающихся списков – «Каталог», «Администрирование», «Склад», «Касса», «Статистика». В зависимости от ограничений роли вошедшего пользователя списки с недоступным функционалом будут заблокированы.

3- Поле для UserControl, в случае если пользователь выбрал в боковом меню функционал, который реализован посредством UsercControl, этот элемент отобразится в данной части экрана.

Далее рассмотрим типовые UserControls и Forms используемые в системе.

UserControl со списком элементов – типовая форма используемая в функционале – Товары, Типы товаров, Категории товаров, Формы выпуска товаров, Список пользователей, Посмотреть склад, Отченость. Для примера возьмем пункт «Товары».



Типовой элемент можно разделить на три области:

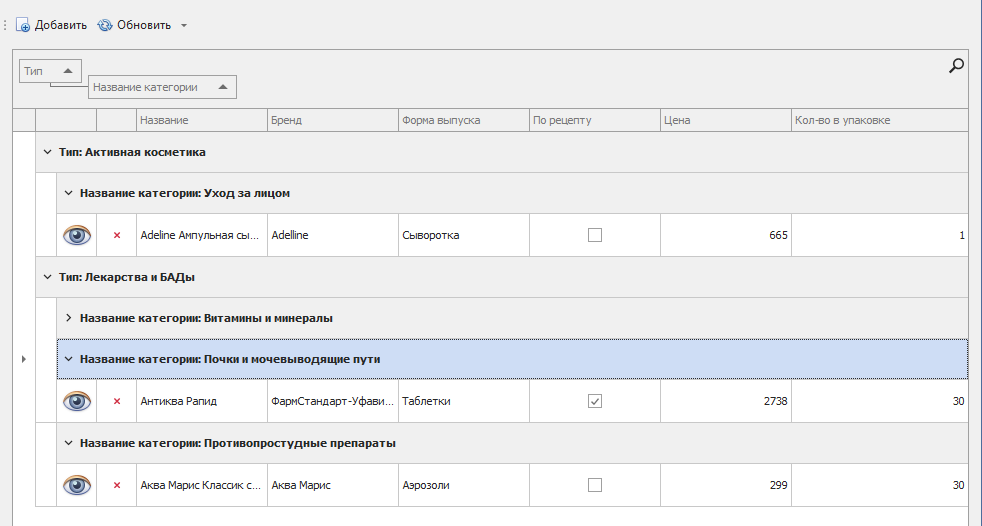
1 – Панель инструментов, содержит кнопку создания нового элемента и обновления списка

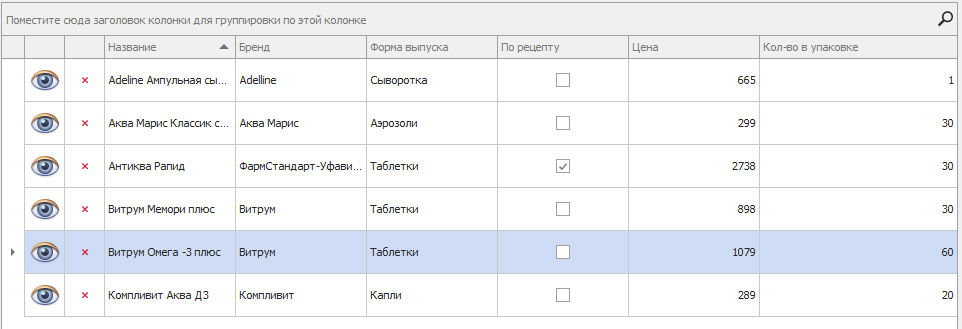
2 – Командные столбцы, содержат кнопки просмотра и удаления конкретного элемента

3- Сама таблица с данными

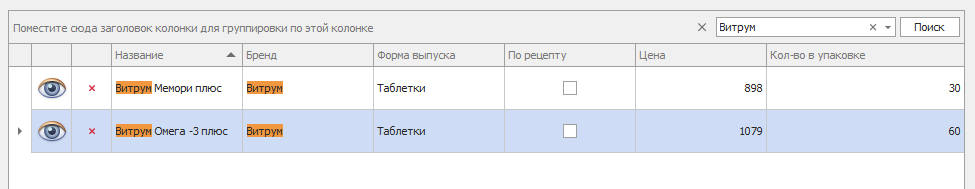
При необходимости все типовые элементы можно:

Группировать:



Сортировать: 

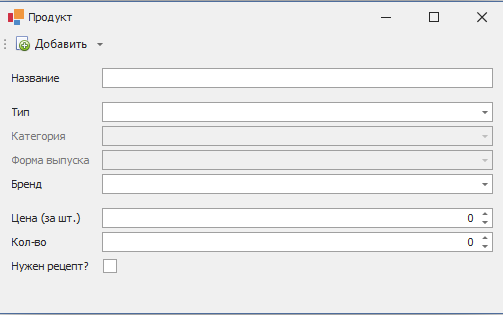
Искать информацию:



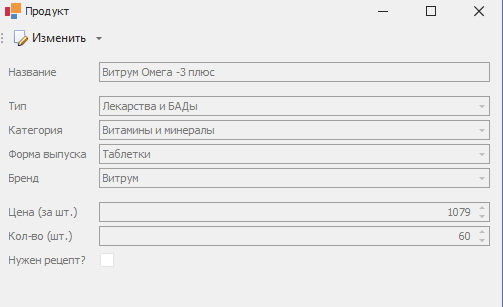
Для создания, просмотра и редактирования объектов используются отдельные формы. Типовая форма работы с объектом имеет три состояния:

Создание объекта

Все поля пустые или заполнены значениями по умолчанию, в панели инструментов только кнопка добавить

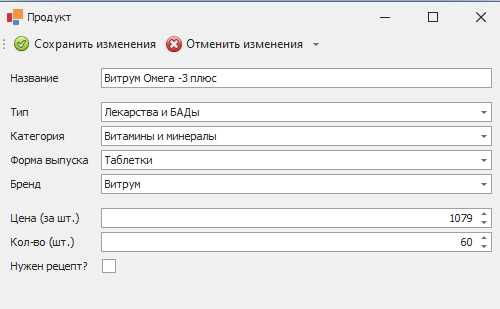


Просмотр объекта:

Все поля заполнены информацией об объекте, форма недоступна для изменения, единственный активный элемент – кнопка изменить, при нажатии на которую форма переключается в состояние редактирования. 

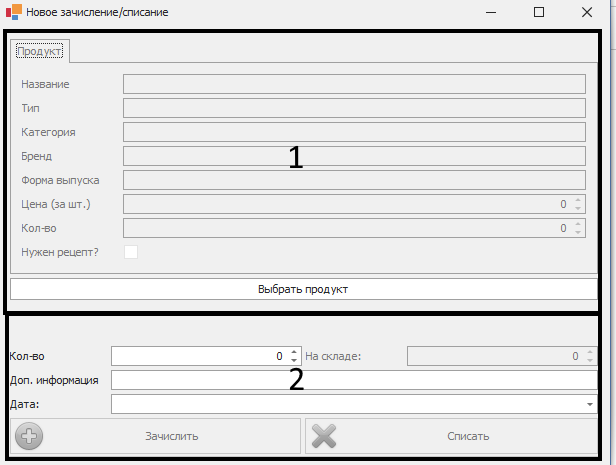
Редактирование объекта:

Так же как и при просмотре, все поля заполнены информацией об объекте, однако становятся доступны для редактирования. В панели инструментов кнопки – сохранить изменения, по нажатию на которую данные будут изменены и сохранены в системе, а форма переключится в просмотр объекта с уже новыми данными и кнопка отменить изменения – форма вернется в просмотр объекта отменив все внесенные изменения.



Отдельно рассмотрим отличные от остальных формы:

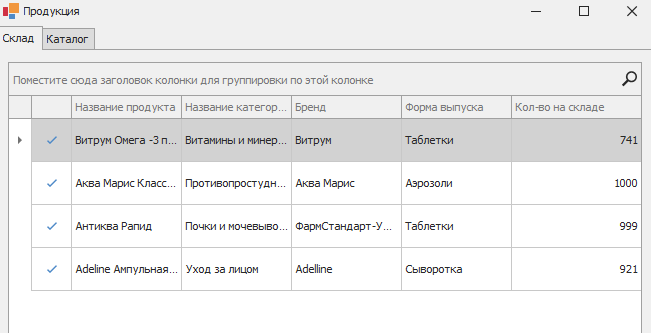
Форма зачисления/списания

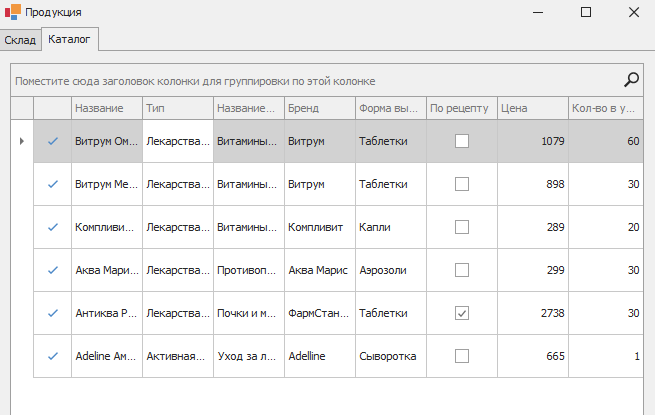


1 – Поле для выбора необходимой продукции, по нажатию выбрать продукт открывается форма просмотра каталога(см. далее), из нее выбирается необходимый продукт и данные о нем загружаются в поля. Область недоступна для редактирования.

2 – Поле для создания накладной, в данной форме содержатся поля информации по проводимому списанию или зачислению, так же недоступное поле «На складе» которое демонстрирует кол-во товаров данного вида продукции в данным момент находится на складе.

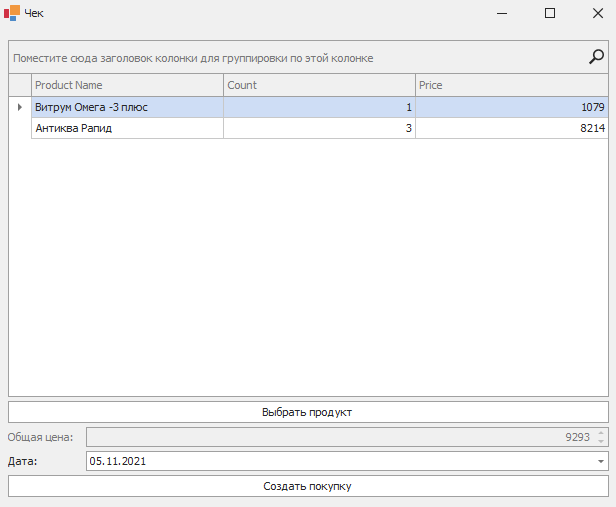
Форма склад/каталог



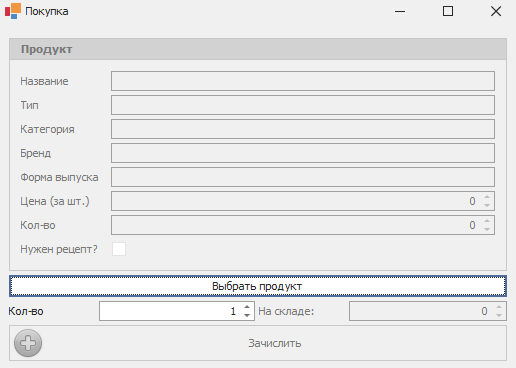


Представляет собой по сути копию UserControls для функционала «Товары» и «Посмотреть склад» с урезанным функционалом, при нажатии на кнопку в столбце управления форма возвращает выбранный продукт.

Форма продажи



Представляет собой таблицу в которую внесены выбранные на продажу товары, нередактируемое поле – общая цена и дата создания продажи, по нажатию кнопки выбрать продукт открывается форма выбора продукта. Которая ыявляется почти полной копией зачисление/списание продукта и с тем-же функционалом.



## Реализация классов

**Классы сущностей**

|  |  |
| --- | --- |
| Наименование | Наследуется от |
| BaseEntity | - |
| Описание: | Базовый класс для сущностей |
| Поля: | Int ID; |

|  |  |
| --- | --- |
| Наименование | Наследуется от |
| BaseNamedEntity | BaseEntity |
| Описание: | Базовый класс для сущностей с наименованием |
| Поля: | String Name; |

|  |  |
| --- | --- |
| Наименование | Наследуется от |
| PharmacyModel | BaseEntity |
| Описание: | Аптека |
| Поля: | String Address  WareHouse WareHouse  Ilist<Sale> Sales |

|  |  |
| --- | --- |
| Наименование | Наследуется от |
| Sale | BaseEntity |
| Описание: | Продажа |
| Поля: | Double Price;  DateTime Date;  Ilist<SaleItem> Items; |

|  |  |
| --- | --- |
| Наименование | Наследуется от |
| SaleItem | BaseEntity |
| Описание: | Позиция продажи |
| Поля: | Product Product;  Sale Sale;  Int Count;  Double Price; |

|  |  |
| --- | --- |
| Наименование | Наследуется от |
| Brand | BaseNamedEntity |
| Описание: | Бренд товара |
| Поля: | Int ID; |

|  |  |
| --- | --- |
| Наименование | Наследуется от |
| CategoryProduct | BaseNamedEntity |
| Описание: | Категория товара |
| Поля: | TypeProduct Type; |

|  |  |
| --- | --- |
| Наименование | Наследуется от |
| FormProduct | BaseNamedEntity |
| Описание: | Форма товара |
| Поля: | TypeProduct Type;  String Unit; |

|  |  |
| --- | --- |
| Наименование | Наследуется от |
| TypeProduct | BaseNamedEntity |
| Описание: | Тип товара |
| Поля: | IList<CategoryProduct> Categories;  IList<FormProduct> Forms; |

|  |  |
| --- | --- |
| Наименование | Наследуется от |
| Product | BaseNamedEntity |
| Описание: | Товар |
| Поля: | CategoryProduct Category;  Brand Brand;  FormProduct Form;  bool IsDeleted;  double Price;  bool IsNeedRecipe; |

|  |  |
| --- | --- |
| Наименование | Наследуется от |
| WareHouseReport | BaseEntity |
| Описание: | Накладная на списание/зачисление товара на склад |
| Поля: | WareHouse WareHouse;  Product Product;  Int Count;  String Type;  String Description;  DateTime Date; |

|  |  |
| --- | --- |
| Наименование | Наследуется от |
| WareHouse | BaseEntity |
| Описание: | Склад |
| Поля: | PharmacyModel Pharmacy;  IList<WareHouseItem> Items;  IList<WareHouseItem> Changes; |

|  |  |
| --- | --- |
| Наименование | Наследуется от |
| WareHouseItem | - |
| Описание: | Товар на складе |
| Поля: | WareHouse WareHouse;  Product Product;  Int Count; |

**Управляющие классы**

|  |  |
| --- | --- |
| Наименование | Наследуется от |
| NHibernateHelper | - |
| Описание: | Статический класс-фасад для работы с сессиями NHibernate |
| Поля: | ISessionFactory \_sessionFactory |
| Функции: | Void Configure()  Void CloseSessionFactory()  Void CloseSession()  ISession OpenSession()  ISession GetCurrentSession() |

|  |  |
| --- | --- |
| Наименование | Наследуется от |
| PharmacySession | - |
| Описание: | Синглтон хранящий текущую сессию |
| Поля: | PharmacySession \_instance;  ISession session; |
| Функции: | PharmacySession GetInstance()  List<T> GetList<T>()  Void Clear()  ICriteria CreateCriteria(Type type)  T Get<T>(int id)  Object Merge(object enity)  Object Save(object entity)  Void Flush()  Void Delete(object entity) |

|  |  |
| --- | --- |
| Наименование | Наследуется от |
| BaseRepository<T> where T:BaseEntity | - |
| Описание: | Базовый класс репозитория, реализует логику взаимодействия сущности с БД  Является обобщением и наследуется для остальных классов репозиториев при необходиости их переимплементации. |
| Поля: | PharmacySession session; |
| Функции: | List<T> All()  T Get(int id)  T Save(T entity)  T Add(T entity)  Bool Delete(T entity) |

|  |  |
| --- | --- |
| Наименование | Наследуется от |
| LoginUser | - |
| Описание: | Реализует авторизацию пользователя в системе. |
| Поля: | User User; |
| Функции: | User GetUser()  Bool CreateUser(string login, string password) |

|  |  |
| --- | --- |
| Наименование | Наследуется от |
| BaseManager<T> where T:BaseEntity | - |
| Описание: | Базовый класс репозитория, реализует бизнес-логику сущности  Является обобщением и наследуется для остальных классов менеджеров при необходиости их переимплементации. |
| Поля: | BaseRepository<T> repository; |
| Функции: | List<T> All()  T Get(int id)  T Update(T entity)  T Add(T entity)  Bool Delete(T entity) |

**Интерфейсные классы**

|  |  |
| --- | --- |
| Наименование | Наследуется от |
| LoginForm | XtraForm |
| Описание: | Форма авторизации |
| Поля: |  |
| Функции: | btnLogin\_Click  LoginForm\_VisibleChanged |

|  |  |
| --- | --- |
| Наименование | Наследуется от |
| MainForm | FluentDesignForm |
| Описание: | Основная форма |
| Поля: | User User |
| Функции: | MainForm\_Load  async Task LoadModuleAsync(ModuleInfo module)  async void ControlCatalogProducts\_Click  async void ControlCatalogTypes\_Click  async void ControlCatalogCategories\_Click  async void ControlCatalogForms\_Click  async void ControlCatalogBrands\_Click  async void ControlUserList\_Click  async void ControlPharmacyList\_Click  async Task Loaduc(string uc, string path) |

|  |  |
| --- | --- |
| Наименование | Наследуется от |
| ucBrand | TutorialControlBase |
| Описание: | Список брендов |
| Поля: |  |
| Функции: | Void loadData()  void ucBrand\_Load  void btnViewElement\_ButtonClick  void barBtnAdd\_ItemClick  void btnDeleteElement\_Click  void barButtonRefresh\_ItemClick |

|  |  |
| --- | --- |
| Наименование | Наследуется от |
| ucCategory | TutorialControlBase |
| Описание: | Список категорий |
| Поля: |  |
| Функции: | Void loadData()  void ucCategory \_Load  void btnViewElement\_ButtonClick  void barBtnAdd\_ItemClick  void btnDeleteElement\_Click  void barButtonRefresh\_ItemClick |

|  |  |
| --- | --- |
| Наименование | Наследуется от |
| ucForm | TutorialControlBase |
| Описание: | Список форм |
| Поля: |  |
| Функции: | Void loadData()  void ucForm \_Load  void btnViewElement\_ButtonClick  void barBtnAdd\_ItemClick  void btnDeleteElement\_Click  void barButtonRefresh\_ItemClick |

|  |  |
| --- | --- |
| Наименование | Наследуется от |
| ucProduct | TutorialControlBase |
| Описание: | Список товаров |
| Поля: |  |
| Функции: | Void loadData()  void ucProduct \_Load  void btnViewElement\_ButtonClick  void barBtnAdd\_ItemClick  void btnDeleteElement\_Click  void barButtonRefresh\_ItemClick |

|  |  |
| --- | --- |
| Наименование | Наследуется от |
| ucSales | TutorialControlBase |
| Описание: | Список продаж |
| Поля: |  |
| Функции: | Void loadData()  void ucSales \_Load  void btnViewElement\_ButtonClick  void barBtnAdd\_ItemClick  void btnDeleteElement\_Click  void barButtonRefresh\_ItemClick |

|  |  |
| --- | --- |
| Наименование | Наследуется от |
| ucStatByTime | TutorialControlBase |
| Описание: | Экран статистики |
| Поля: |  |
| Функции: | Void loadData()  void ucStatByTime \_Load  void btnViewElement\_ButtonClick  void barBtnAdd\_ItemClick  void btnDeleteElement\_Click  void barButtonRefresh\_ItemClick |

|  |  |
| --- | --- |
| Наименование | Наследуется от |
| ucType | TutorialControlBase |
| Описание: | Список типов |
| Поля: |  |
| Функции: | Void loadData()  void ucType \_Load  void btnViewElement\_ButtonClick  void barBtnAdd\_ItemClick  void btnDeleteElement\_Click  void barButtonRefresh\_ItemClick |

|  |  |
| --- | --- |
| Наименование | Наследуется от |
| ucUsers | TutorialControlBase |
| Описание: | Список пользователей |
| Поля: |  |
| Функции: | Void loadData()  void ucUsers \_Load  void btnViewElement\_ButtonClick  void barBtnAdd\_ItemClick  void btnDeleteElement\_Click  void barButtonRefresh\_ItemClick |

|  |  |
| --- | --- |
| Наименование | Наследуется от |
| ucWareHouse | TutorialControlBase |
| Описание: | Склад |
| Поля: |  |
| Функции: | Void loadData()  void ucWareHouse\_Load  void btnViewElement\_ButtonClick  void barBtnAdd\_ItemClick  void btnDeleteElement\_Click  void barButtonRefresh\_ItemClick |

|  |  |
| --- | --- |
| Наименование | Наследуется от |
| BrandForm | XtraForm |
| Описание: | Форма товара |
| Поля: |  |
| Функции: | BrandForm(int id)  void BrandForm\_Load  void barBtnCancelEdit\_ItemClick  void barBtnStartEdit\_ItemClick  void barBtnCompleteEdit\_ItemClick  void barBtnAdd\_ItemClick  Brand saveData() |

|  |  |
| --- | --- |
| Наименование | Наследуется от |
| CatalogForm | XtraForm |
| Описание: | Форма каталога |
| Поля: |  |
| Функции: | CatalogForm(int id)  void CatalogForm\_Load  void barBtnCancelEdit\_ItemClick  void barBtnStartEdit\_ItemClick  void barBtnCompleteEdit\_ItemClick  void barBtnAdd\_ItemClick  Catalog saveData() |

|  |  |
| --- | --- |
| Наименование | Наследуется от |
| CategoryForm | XtraForm |
| Описание: | Форма категории |
| Поля: |  |
| Функции: | CategoryForm(int id)  void CategoryForm\_Load  void barBtnCancelEdit\_ItemClick  void barBtnStartEdit\_ItemClick  void barBtnCompleteEdit\_ItemClick  void barBtnAdd\_ItemClick  CategorysaveData() |

|  |  |
| --- | --- |
| Наименование | Наследуется от |
| FormForm | XtraForm |
| Описание: | Формы формы товара |
| Поля: |  |
| Функции: | FormForm(int id)  void FormForm\_Load  void barBtnCancelEdit\_ItemClick  void barBtnStartEdit\_ItemClick  void barBtnCompleteEdit\_ItemClick  void barBtnAdd\_ItemClick  Form saveData() |

|  |  |
| --- | --- |
| Наименование | Наследуется от |
| ProductForm | XtraForm |
| Описание: | Форма товара |
| Поля: |  |
| Функции: | ProductForm(int id)  void ProductForm\_Load  void barBtnCancelEdit\_ItemClick  void barBtnStartEdit\_ItemClick  void barBtnCompleteEdit\_ItemClick  void barBtnAdd\_ItemClick  Product saveData() |

|  |  |
| --- | --- |
| Наименование | Наследуется от |
| ReportForm | XtraForm |
| Описание: | Форма накладной о складе |
| Поля: |  |
| Функции: | ReportForm(int id)  void ReportForm\_Load  void barBtnCancelEdit\_ItemClick  void barBtnStartEdit\_ItemClick  void barBtnCompleteEdit\_ItemClick  void barBtnAdd\_ItemClick  Report saveData() |

|  |  |
| --- | --- |
| Наименование | Наследуется от |
| SaleForm | XtraForm |
| Описание: | Форма продажи |
| Поля: |  |
| Функции: | SaleForm(int id)  void SaleForm\_Load  void barBtnCancelEdit\_ItemClick  void barBtnStartEdit\_ItemClick  void barBtnCompleteEdit\_ItemClick  void barBtnAdd\_ItemClick  Sale saveData() |

|  |  |
| --- | --- |
| Наименование | Наследуется от |
| SaleItemForm | XtraForm |
| Описание: | Форма добавления позиции в продажу |
| Поля: |  |
| Функции: | SaleItemForm(int id)  void SaleItemForm\_Load  void barBtnCancelEdit\_ItemClick  void barBtnStartEdit\_ItemClick  void barBtnCompleteEdit\_ItemClick  void barBtnAdd\_ItemClick  SaleItemsaveData() |

|  |  |
| --- | --- |
| Наименование | Наследуется от |
| UserForm | XtraForm |
| Описание: | Форма пользователя |
| Поля: |  |
| Функции: | UserForm(int id)  void UserForm\_Load  void barBtnCancelEdit\_ItemClick  void barBtnStartEdit\_ItemClick  void barBtnCompleteEdit\_ItemClick  void barBtnAdd\_ItemClick  User saveData() |

## Архитектура



# Тестирование

Для тестирования итогового приложения поочередно произведем проверки всего основого функционала системы:

|  |  |
| --- | --- |
| 1.1. | Авторизация с правильными входными данными. |
| Ожидаемый  Результат: | Система авторизирует пользователя, открывает главное меню и дает доступ к функционалу в зависимости от роли. |

|  |  |
| --- | --- |
| 1.2. | Авторизация с неправильными входными данными. |
| Ожидаемый  Результат: | Система не авторизирует пользователя и уведомляет его об ошибке. |

|  |  |
| --- | --- |
| 2. | Создание типа продукта |
| Ожидаемый  Результат: | Тип добавлен в БД, тип можно выбрать при создании продукта |

|  |  |
| --- | --- |
| 3. | Создание категории продукта |
| Ожидаемый  Результат: | Категория добавлена в БД, категорию можно выбрать при создании продукта |

|  |  |
| --- | --- |
| 4. | Создание формы выпуска продукта |
| Ожидаемый  Результат: | Форма выпуска добавлена в БД, форму выпуска можно выбрать при создании продукта |

|  |  |
| --- | --- |
| 5.1. | Создание продукта |
| Ожидаемый  Результат: | Продукт добавлен в БД и доступен для выбора из каталога. |

|  |  |
| --- | --- |
| 5.2. | Создание продукта с некорректными данными |
| Ожидаемый  Результат: | Система выводит сообщение об ошибке и не производит сохранение в БД. |

|  |  |
| --- | --- |
| 6. | Изменение продукта |
| Ожидаемый  Результат: | Продукт в бд обновлен и доступен для выбора из каталога с обновленными данными. |

|  |  |
| --- | --- |
| 7. | Удаление продукта |
| Ожидаемый  Результат: | Поле Deleted у продукта в БД выставлено в TRUE, продукт не отображается в каталоге. Все отчетности с данным продуктом корректно открываются и отображают данные. |

|  |  |
| --- | --- |
| 8. | Зачисление продукта на склад |
| Ожидаемый  Результат: | Обновление склада в БД, продукт доступен для выбора со склада. |

|  |  |
| --- | --- |
| 9.1. | Списание товара со склада в большем объеме, чем имеется на складе |
| Ожидаемый  Результат: | Система уведомляет пользователя об ошибке и не производит операцию. Накладная не записывается в БД. |

|  |  |
| --- | --- |
| 9.2. | Списание товара со склада. |
| Ожидаемый  Результат: | Количество товара на складе уменьшается, накладная записывается в БД. |

|  |  |
| --- | --- |
| 10.1. | Создание продажи без указания позиций продажи. |
| Ожидаемый  Результат: | Система уведомляет пользователя об ошибке и не производит операцию. Продажа не записывается в БД. |

|  |  |
| --- | --- |
| 10.2. | При создании продажи выбрать большее кол-во товара чем имеется на складе. |
| Ожидаемый  Результат: | Система уведомляет пользователя об ошибке и не производит операцию. Товар не записывается в продажу. |

|  |  |
| --- | --- |
| 10.3. | Корректное создание продажи. |
| Ожидаемый  Результат: | Продажа записана в БД, количество товара на складе уменьшено. |

|  |  |
| --- | --- |
| 11. | Создание пользователя |
| Ожидаемый  Результат: | После выхода из системы в нее можно зайти под данными созданного пользователя. |

|  |  |
| --- | --- |
| 12. | Удаление пользователя |
| Ожидаемый  Результат: | После удаления и выхода из системы в нее нельзя войти под данными удаленного пользователя. |

|  |  |
| --- | --- |
| 13. | Проверка корректности статистики |
| Алгоритм: | Сохранить значения статистики при различных вариациях настроек  Создать продукт с неестественно высокой стоимостью  Зачислить созданный продукт на склад  Создать продажу созданного продукта  Проверить изменения статистики относительно сохраненных данных  Удалить добавленные объекты  Проверить изменения статистики относительно сохраненных данных |
| Ожидаемый  Результат: | При первом сравнении данные за определенный месяц/время года должны резко возрасти. При выборе фильтра по категории отличной от категории проданного товара, статистика должна остаться прежней.  При втором сравнении данные должны быть идентичны первоначальным |

Результаты тестирования:

|  |  |  |
| --- | --- | --- |
| Номер и название теста | Успех | Провал |
| * 1. Авторизация с правильными входными данными. | + |  |
| 1.2. Авторизация с неправильными входными данными. | + |  |
| 2. Создание типа продукта | + |  |
| 3. Создание категории продукта | + |  |
| 4. Создание формы выпуска продукта | + |  |
| 5.1. Создание продукта | + |  |
| Ыы5.2. Создание продукта с некорректными данными | + |  |
| 6. Изменение продукта | + |  |
| 7. Удаление продукта | + |  |
| 8. Зачисление продукта на склад | + |  |
| 9.1. Списание товара со склада в большем объеме, чем имеется на складе | + |  |
| 9.2. Списание товара со склада. | + |  |
| 10.1. Создание продажи без указания позиций продажи. | + |  |
| 10.2. При создании продажи выбрать большее кол-во товара чем имеется на складе | + |  |
| 10.3. Корректное создание продажи. | + |  |
| 11. Создание пользователя | + |  |
| 12. Удаление пользователя | + |  |
| 13. Проверка корректности статистики | + |  |