Министерство образования Республики Беларусь

Учреждение образования

БелорусскиЙ государственный университет

информатики и радиоэлектроники

Факультет информационных технологий и управления

Кафедра информационных технологий автоматизированных систем

Дисциплина: Объектно-ориентированное программирование

|  |
| --- |
|  |
|  |
|  |

Пояснительная записка

к курсовой работе

на тему

**ИНФОРМАЦИОННАЯ СИСТЕМА СКЛАДА ПРЕДПРИЯТИЯ**

БГУИР КР 1-53 01 02 109 ПЗ

|  |  |  |
| --- | --- | --- |
| Студент |  | Д. С. Ермак |
| Руководитель |  | А. А. Навроцкий |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Минск 2022

СОДЕРЖАНИЕ

[Введение 5](#_Toc144758627)

[1 Анализ технического задания на курсовое проектирование 6](#_Toc144758628)

[1.1 Анализ предметной области 6](#_Toc144758629)

[1.2 Обзор систем-аналогов 7](#_Toc144758630)

[1.3 Постановка задачи 10](#_Toc144758631)

[2 Проектирование иерархии классов склада предприятия 12](#_Toc144758632)

[2.1 Разработка функционала и построение блок-схемы 12](#_Toc144758633)

[2.2 Разработка структуры классов 14](#_Toc144758634)

[3 Программная реализация иерархии классов 18](#_Toc144758635)

[3.1 Описание процесса реализации 18](#_Toc144758636)

[3.2 Описание работы программы 19](#_Toc144758637)

[Заключение 30](#_Toc144758638)

[Список использованных источников 32](#_Toc144758639)

[Приложение А 33](#_Toc144758640)

# 

# ВВЕДЕНИЕ

Современный уровень развития различных программных и аппаратно-программных систем, и в первую очередь, систем управления, связан со значительным повышением их сложности из-за присущих им требований к гибкости, доступности и поддержке.

На данный момент наиболее перспективным можно считать объектно-ориентированный подход проектирования. К существенным отличиям объектно-ориентированного подхода от процедурного проектирования относят возможность использования итеративно-поступательного цикла создания программного обеспечения и перенос акцента проектирования с разработки алгоритмов функционирования системы на построения системы абстракций и их взаимодействия. Существенным минусом объектно-ориентрованного подхода можно назвать скорость разработки, однако, благодаря структурированности и достаточной простоте в понимании, значительно снижается риск различного рода критических ошибок.

Актуальность выбранного метода разработки информационной системы – упрощение структурирования данных, гибкость применения созданного алгоритма, доступность, а также возможность поддержки, позволяющая подстраивать созданные алгоритмы под требования пользователя.

Данная курсовая работа нацелена на создание консольного приложения, позволяющего автоматизировать процесс работы склада предприятия.

Темой курсового проекта является создание информационной системы склада предприятия. Данная тема актуальна в связи с тем, что в любой фирме, занимающейся реализацией того или иного товара необходимы постоянный контроль поступления и отпуска товара. Хорошо структурированная и организованная система учета зачастую избавляет от невынужденных ошибок человеческого характера.

Основная цель информационной системы заключается в том, чтобы обеспечить сбор, обработку, хранение, изменение и удаление информации. Также целью проекта является развитие навыков самостоятельной и творческой работы и закрепление навыков работы на языке *C****#****,* совместно с использованием системы управления базами данных *MySQL* и фреймворком *Entity Framework*.

# Анализ технического задания на курсовое проектирование

## Анализ предметной области

Любое предприятие, так или иначе занимающееся распространением той или иной продукции, сталкивается с необходимостью реализации складской системы, которая включает в себя прием продукции, ее учет, хранение, комплектацию, сортировку и поставке в сторонние организации.

Так как сохраняется тенденция развития информационных технологий, любая подобная система требует программной реализации. Подобные информационные системы используются многими предприятиями по всему миру. Многие складские информационные системы реализуются при помощи 1С, однако они могут быть реализованы на любом другом языке программирования. Наиболее для этого подходят объектно-ориентированные языки, одним из которых является *C****#***.

Для создания информационной системы склада предприятия есть несколько весомых причин:

Во-первых, актуальность. Огромное количество предприятий различных масштабов занимается реализацией самых разных категорий продукции. В следствие чего, на рынке постоянно требуется программное обеспечение для подобных систем, а уже существующие нуждаются в постоянном усовершенствовании и поддержке.

Во-вторых, дистанционное взаимодействие. Предприятие имеет возможность сэкономить на найме лишних сотрудников, поскольку информационная система позволит одному либо нескольким сотрудникам контролировать весь процесс приема и поставки продукции, а также данная система будет удобна поставщикам и заказчикам, так как заявку можно оставить онлайн.

Третья причина – минимизации риска ошибки из-за человеческого фактора. Информационная система контролирует процесс работы и, благодаря грамотной реализации структуры информационной системы, риск ошибки из-за невнимательности или халатности сотрудника будет минимизирован.

Исходя из написанного выше, можно смело утверждать, что создание данной информационной системы актуально и сейчас. А в связи с постоянным ростом количества и размеров подобных предприятий, разработка данной системы востребована в наше время.

## Обзор систем-аналогов

В данном курсовом проекте необходимо разработать иерархию классов для реализации информационной системы склада предприятия. Для создания данной системы были изучены веб-сайты, которые предоставляют возможность ведения складского учета на своей платформе, для получения информации о необходимом функционале, а также об особенностях подобных систем. Также нужно понимать, что ресурс должен иметь интуитивно-понятный интерфейс и обладать высокой функциональностью. Наиболее популярными сайтами являются *btbpm.com*, *toplogwms.ru* и *moysklad.ru.*

Возможности, предоставляемые сайтом *btbpm.com* можно увидеть на рисунке 1.1.

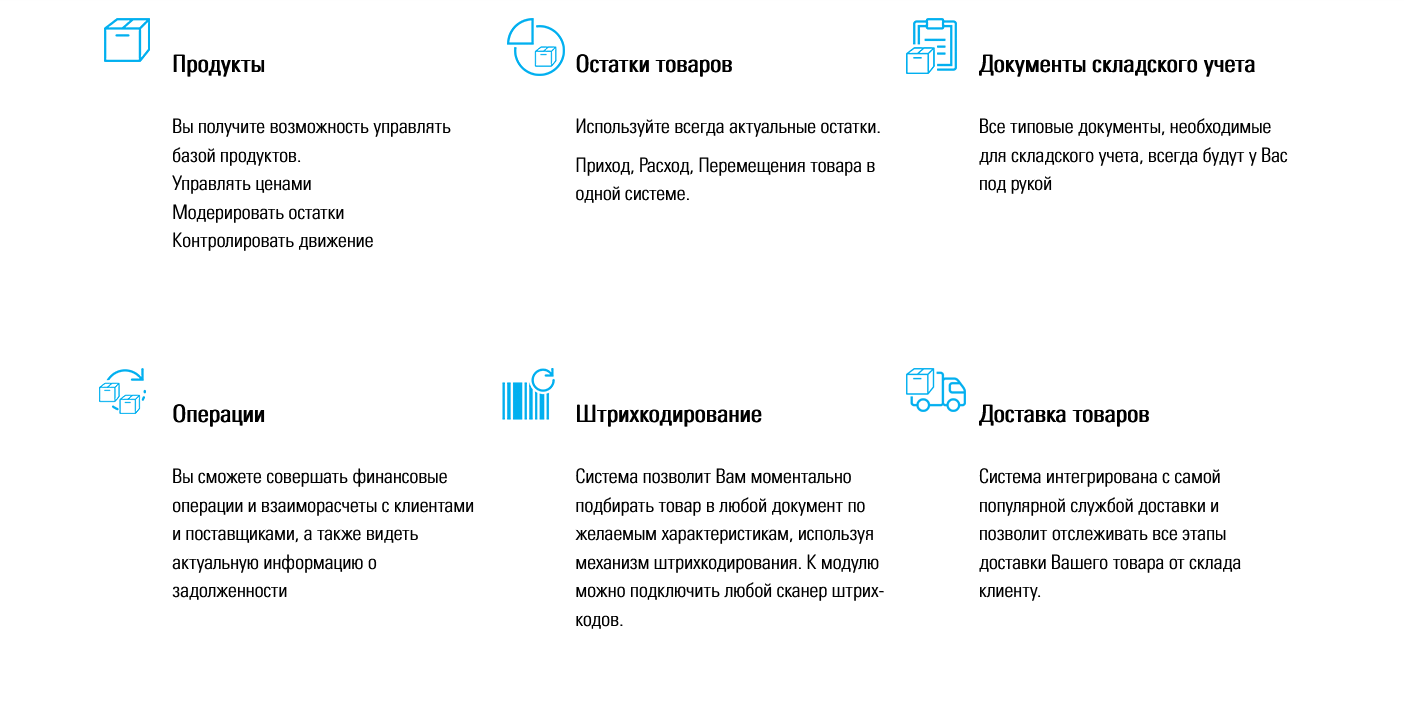


Рисунок 1.1‑ Функциональные возможности сайта *btbpm.com*

Данный сайт обладает достаточно неплохим функционалом, интуитивно понятным дизайном, однако на нем отсутствует возможность сбора всевозможной статистики, в следствие чего данный вариант не удобен для больших предприятий, которым необходимо анализировать статистику о своих клиентах.

Сайт *toplogwms.ru* предоставляет больший функционал, в отличие от предыдущего примера. Функциональные возможности данного сайта представлены на рисунке 1.2.



Рисунок 1.2 – Функциональные возможности сайта  
*toplogwms.ru*

Ярким представителем информационных систем складского учета является продукт «1С: Предприятие 8. *WMS* Логистика. Управление складом». Функционал системы "1С: *WMS* Логистика. Управление складом" позволяет оптимизировать процессы и решить основные проблемы, актуальные для складских комплексов:

* оптимизация использования складских площадей при размещении и хранении товара;
* сокращение затрат на складское хранение;
* сокращение времени и количества ошибок на обработку складских операций;
* повышение точности и оперативности учета товара;
* исключение потерь, связанных с критичностью сроков реализации товаров;
* уменьшение затрат на заработную плату складских работников.

"1С: *WMS* Логистика. Управление складом" предназначена для управления процессами складской грузообработки в режиме реального времени, а именно:

* получения актуальной информации об остатках товара на складе в "онлайн" режиме;
* оптимизации товарных потоков на складе;
* управления и оптимизации приемки, размещения, перемещения, отбора, отгрузки и прочих складских операций;
* контроля работы складского персонала.

В свою очередь, сайт *moysklad.ru* имеет приятный и понятный интерфейс, достаточный функционал, включающий в себя и сбор статистических данных. Возможности сервиса представлены на рисунке 1.3.

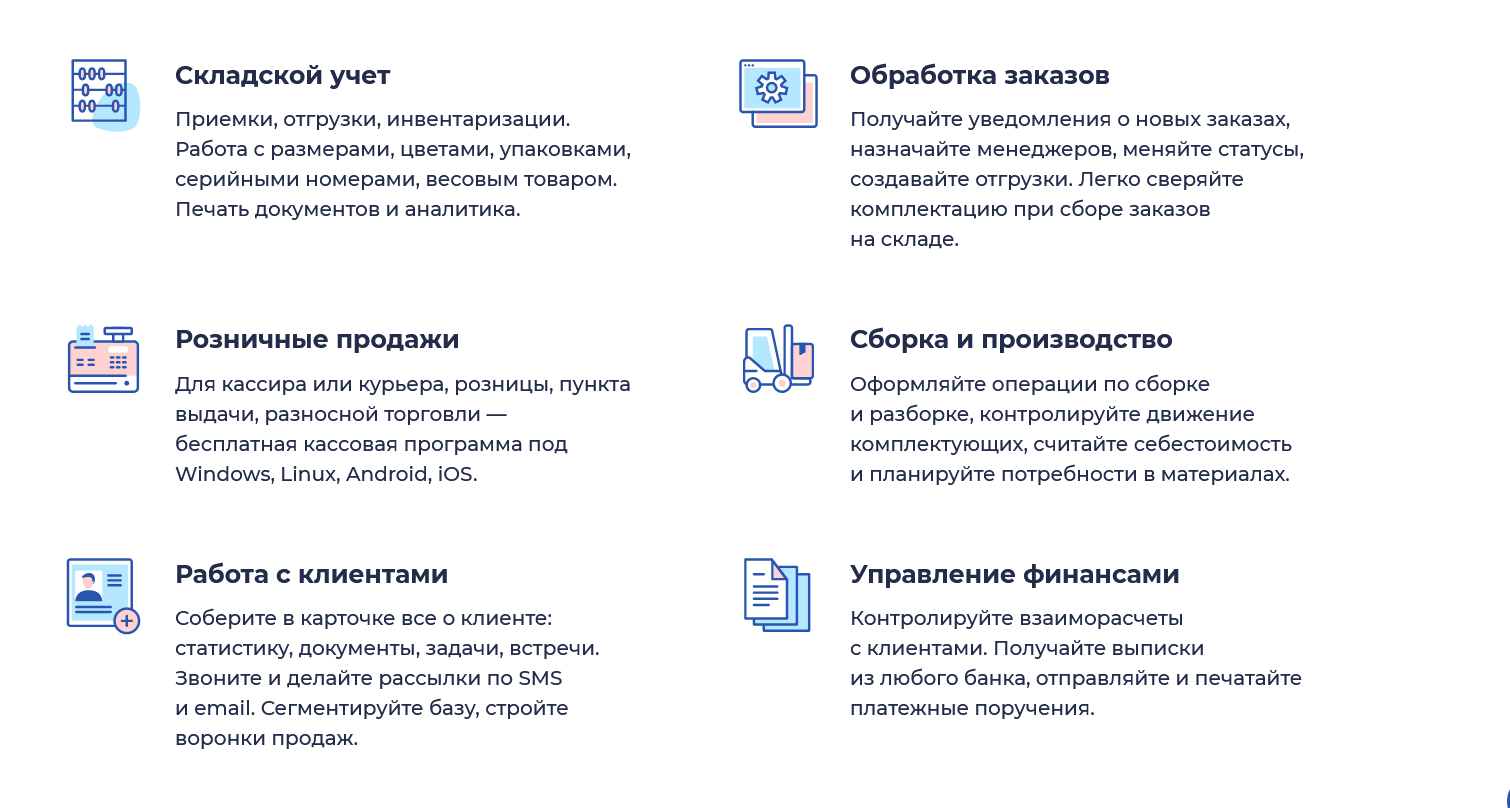


Рисунок 1.3 – Функциональные возможности сайта *moysklad.ru*

Также популярным решением на рынке информационных систем автоматизации работы склада предприятия является *leadwms.ru.* Данная информационная система используется такими компаниями, как Балтика, Аманат, Рельеф Центр, ГК ЭФКО, *Fix Price*. *LogistiX* (компания-разработчик данной информационной системы) – российский разработчик *IT*-решений для логистики, инновационный лидер рынка *WMS* в России. На рисунке 1.4 представлены решения данной информационной системы для различных ситуаций.

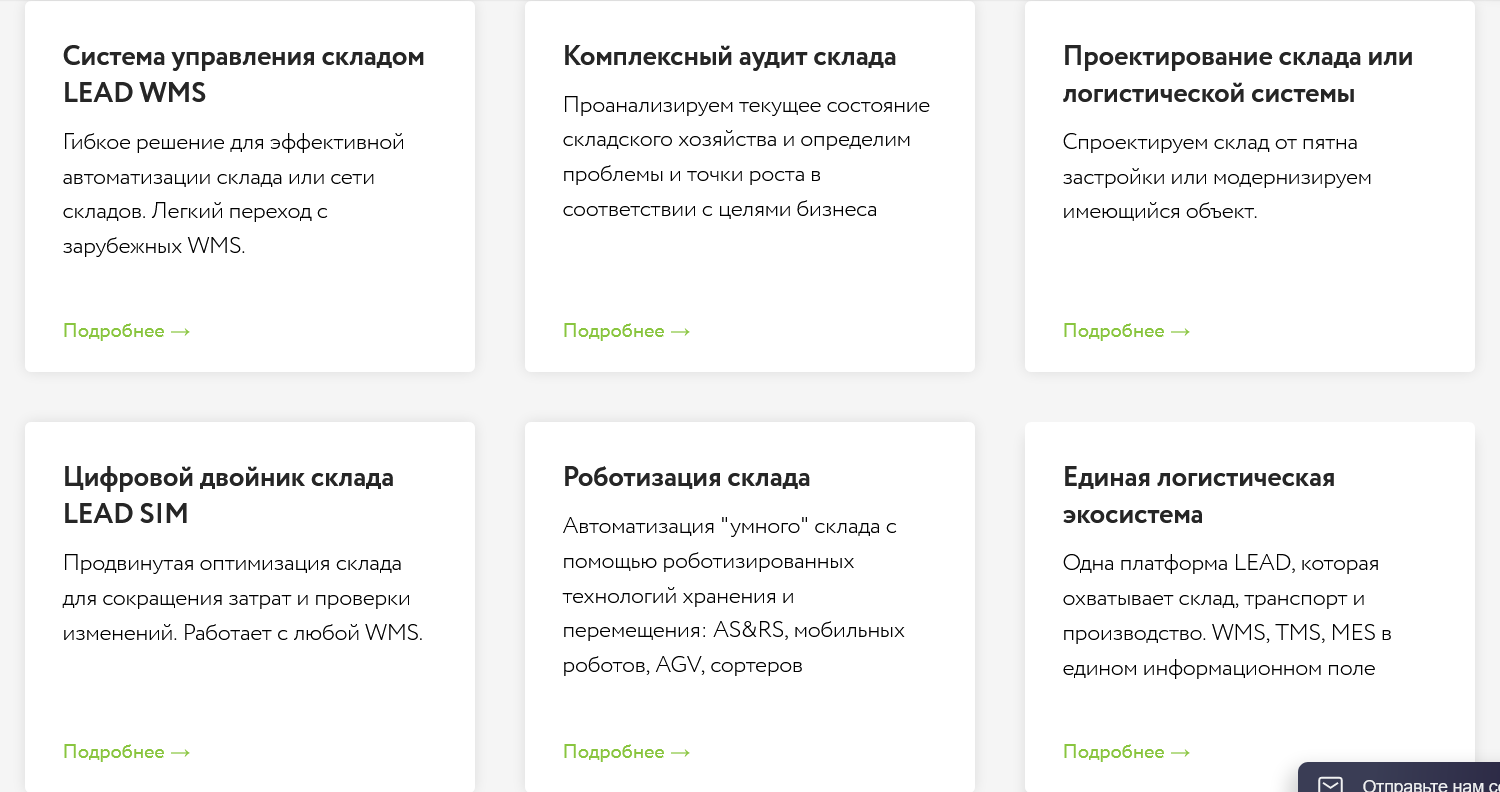


Рисунок 1.4 – Решения, предлагаемые *LogistiX*

Проанализировав системы-аналоги, можно сделать вывод. При разработке иерархии классов для реализации информационной системы склада предприятия необходимо сделать упор на реализацию максимального количества функциональных возможностей по тематике: от основополагающих процессов складского учета до подведения статистики в том или ином виде.

## Постановка задачи

В настоящей курсовой работе необходимо разработать структуру иерархии классов информационной системы склада предприятия. На основе этой структуры необходимо создать приложение для тестирования иерархии.

Для реализации поставленных задач необходимо:

* разработать иерархию классов, определить базовый и наследуемые классы;
* разработать и описать структуру каждого класса в отдельности, объявить поля и методы класса;
* освоить современные технологии для работы с базами данных *MySQL* используя при этом фреймворк *Entity Framework Core* и применить их при создании информационной системы;
* создать базу данных, которая будет хранить всю информацию, и связать её с нашей программой.

При реализации приложения необходимо использование объектно-ориентированного программирования. Оно предусматривает наличие базового и производных классов, что делает приложение более универсальным. По возможности можно добавлять, изменять или удалять в приложении различные методы и поля, иначе говоря, подстраивать приложение под конкретную ситуацию. Однако, как отмечалось ранее, объектно-ориентированный подход замедляет разработку решений, но за счет структурированности позволяет избежать большого количества ошибок, а в следствии упрощает процесс устранения неисправностей и поддержки.

Система склада предприятия должна обеспечивать операции над данными о товарах (добавление, изменение, удаление). Система должна поддерживать хранение данных, которые в нашем случае хранятся в базе данных.

В текущей курсовой работе в качестве базы данных будет использоваться *MySQL* и фреймворк *Entity Framework* для работы с ней, который позволяет абстрагироваться от самой базы данных и ее таблиц и работать с данными как с объектами классом независимо от типа хранилища. Стоит отметить, что *Entity Framework* предоставляет универсальный *API* (*Application programming interface*) для работы с данными. И если, к примеру, мы решим сменить целевую систему управления базами данных, то основные изменения в проекте будут касаться прежде всего конфигурации и настройки подключения к соответствующим провайдерам. А код, который непосредственно работает с данными, получает данные, добавляет их в базу данных, останется прежним.

Система обеспечивает добавление, изменение и удаление данных в базе данных, а также их поиск и сортировку в базе данных на основании той или иной информации. Система складского учета по команде пользователя формирует и выводит на экран различную справочную информацию.

Кроме этого, в системе будем использовать асинхронное программирование. Асинхронная программа не выполняет операции в иерархическом или последовательном порядке. Получающееся в результате распараллеливание означает, что асинхронная программа может обрабатывать несколько запросов одновременно и независимо. Асинхронное программирование используется для оптимизации высоконагруженных приложений с частым ожиданием системы, поэтому использование данного подхода ускоряет работу нашего приложения.

# ПРОЕКТИРОВАНИЕ ИЕРАРХИИ КЛАССОВ СКЛАДА ПРЕДПРИЯТИЯ

### **2.1 Разработка функционала и построение блок-схемы**

Действующие лица:

* Администратор – сотрудник, управляющий складом;
* Система(программа) – система, которая обрабатывает команды и выдает запрошенную информацию;
* База данных – отдельная часть системы, являющаяся базой хранения всей необходимой информации.

Можно выделить следующие группы функций информационной системы:

* добавление сотрудника / товара / склада / расположения на складе /отдела / должности / категории товара;
* изменение сотрудника / товара / склада / расположения на складе /отдела / должности / категории товара;
* удаление сотрудника / товара / склада /отдела / должности / категории товара;
* просмотр информации о сотрудниках / товарах / складах /отделах / должностях / категориях товаров;
* просмотр отсортированной по тому или иному параметру информации о сотрудниках / складах / товарах / отделах.

Манипуляции над данными производится администратором путем запроса в систему. Далее выполняются соответсвующие запросу действия и результат сохраняется в базе данных. Администратор может производить любые манипуляции над данными которые включают: добавление, изменение, удаление и сортировку. Система является посредником между пользователем и хранимой в базе информацией.

Опишем общий алгоритм работы программы.

При запуске программы мы попадает в главное меню, в котором описаны следующие пункты:

1 Вывести информацию о информационной системе.

2 Работы с информацией.

3 Завершение работы программы.

При выборе пункта «Вывести информацию о информационной системе», открывается меню вывода, которое включает в себя пункты:

1 Вывести информацию об отделах компании.

– выводится каждый отдел компании, если он не включает в себя сотрудников, то указывается что отдел пуст;

– выводятся сотрудники этого отдела и их справочные данные.

2 Вывести информацию о сотрудниках компании.

– выводятся отсортированный список сотрудников, каждый сотрудник включает в себя поля: имя, фамилия, отдел, должность, зарплата, телефон и возраст.

3 Вывести информацию о складах компании и их товарах.

– выводится каждый склад, если складов не существует, то выведется сообщение об этом;

– выводятся отсортированные категории товаров, а затем товары, соответствующие этой категории, если на складе отсутствуют товары, то выведется сообщение об этом.

4 Выход

При выборе пункта «Работа с информацией», открывается меню для работы, которое включает в себя пункты, которые содержат меню для работы с конкретными данными:

1 Работа с сотрудниками.

– добавить сотрудника;

– изменить сотрудника;

– уволить сотрудника;

– просмотреть сотрудников;

– выход.

2 Работа с товарами.

– добавить товар;

– изменить товар;

– удалить товар;

– просмотреть характеристики товара;

– просмотреть список товаров;

– выход.

3 Работа с отделами.

– добавить / изменить / удалить / просмотреть отдел;

– выход.

4 Работа с должностями.

– добавить / изменить / удалить / просмотреть должность;

– выход.

5 Работа со складами.

– добавить / изменить / удалить / просмотреть склад;

– выход.

6 Работа с категориями товаров.

– добавить / изменить / удалить / просмотреть категорию товара;

– выход.

7 Выход.

Описание функции «Добавить товар»:

Данный функция начинает выполняться, когда необходимо добавить в базу данных новый товар.

1. Работник вводит необходимые данные о новом товаре.
2. Система получает вводимые данные.
3. Данные отправляются в базу данных.
4. Сохранение базы данных.
5. В консоль выводится сообщение об успешном добавлении товара.

### **2.2 Разработка структуры классов**

При разработке структуры классов было создано 23 класса.

1. *DefaultUser* (базовый абстрактный класс, хранит базовые данные пользователя);
2. *User* (класс, производный от *DefaultUser*, содержит более полную информацию о пользователе, чем базовый);
3. *Employee* (класс, производный от *User*, содержащий информацию о сотрудниках);
4. *Department* (класс, содержащий информацию об отделах);
5. *Post* (класс, содержащий информацию о должностях);
6. *Product* (класс, содержащий информацию о товарах);
7. *Address* (класс, хранит данные о складах);
8. *CategoryOfProduct* (класс, хранит данные о категориях);
9. *Location* (класс, хранит данные расположения товаров на складе);
10. *ApplicationDbContext* (класс, для подключения к базе данных и её настройки);
11. *InsertDatabase* (класс, для базового заполнения базы данных);
12. *Menu* (класс, содержит функционал главного меню);
13. *MenuOutputOfCompanyInformation* (класс, содержит функционал для вывода отсортированных данных информационной система);
14. *MenuWorkWithInformation* (класс, содержит функционал для работы с базой данных);
15. *Output* (класс, для вывода информации);
16. *MenuWorkWithProduct* (класс, для работы с товарами);
17. *MenuWorkWithErmployee* (класс, для работы с сотрудниками);
18. *MenuWorkWithDepartment* (класс, для работы с отделами);
19. *MenuWorkWithPost* (класс, для работы с должностями);
20. *MenuWorkWithCategoryOfProduct* (класс, для работы с категориями товаров);
21. *MenuWorkWithAddress* (класс, для работы со складами);
22. *InitialMigration* (класс, наследуемый от класса *Migration*, для миграции базы данных);
23. *Program* (класс, хранит метод *main*).

Схема взаимодействия основных классов программы представлена на рисунке 2.1.

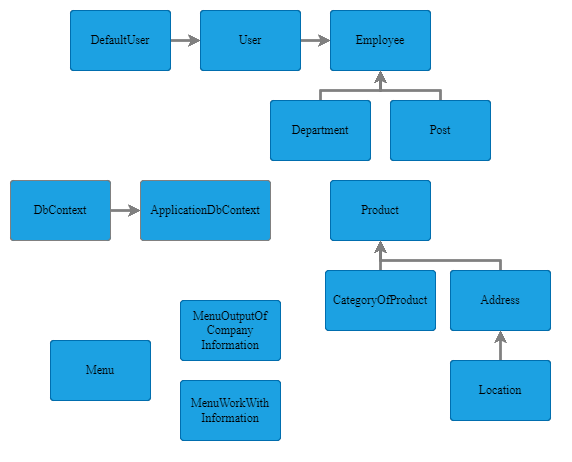


Рисунок 2.1 – Схема взаимодействия основных классов программы

Класс *ApplicationDbContext*, наследуется от класса *DbContext*, который мы добавляем путем подключения фреймворка *Entity Framework Core*, который позволяет абстрагироваться от самой базы данных и ее таблиц и работать с данными как с объектами классом независимо от типа хранилища. Данная технология предполагает работу не с таблицами, а с объектами и их коллекциями. Данный класс добавляет наши классы, такие как *Address*, *CategoryOfProduct*, *Product*, *Location*, *User*, *Employee*, *Department*, *Post*, которые имеют связь как показано на рисунке 2.1, в базу данных, также в данном классе происходит полная настройка, как и самой базы данных, так и подключения и работы с ней.

Стоит отметить, что *Entity Framework* предоставляет универсальный *API* для работы с данными. И если, к примеру, мы решим сменить целевую систему управления базами данных, то основные изменения в проекте будут касаться прежде всего конфигурации и настройки подключения к соответствующим провайдерам. А код, который непосредственно работает с данными, получает данные, добавляет их в базу данных, останется прежним.

В курсовой работе класс *DefaultUser,* он хранит информацию о имени пользователя, является базовым для класса *User*, который хранит информацию о фамилии, возрасте, телефоне, который далее наследуется классом *Employee*, где добавляется новое поле зарплата, а также связь с таблицами *Department* и *Post,* как показано на рисунке 2.1, которые непосредственно хранят информацию об отделе и должности сотрудника; все эти классы содержат конструкторы, проверку вводимой информации и свойства (*get* и *set*).

Также существуют классы *Product*, CategoryOfProduct, *Address, Location*; классы, для взаимодействия с данными: *Menu*, *MenuOutputOfCompany-Information*, *MenuWorkWithInformation*, *MenuWork-WithEmployee*, *MenuWork-WithDepartment*, *MenuWorkWithAddress*, *MenuWorkWithPost*, *MenuWorkWith-Product*, *MenuWorkWithCategoryOfProduct*; класс для миграции базы данных *InitialMigration*; класс *ReceiveBdInformation* для получения информации из базы данных; класс для вывода информации *Output*; класс хранящий метод *main*, *Program*.

Классы *Address*, *CategoryOfProduct* и *Location* содержат информацию о складах, категориях товаров и расположению товара соответственно. Класс *Address* связан с классом *Location*, а класс *Product* имеет связь с классами *Address* и *CategoryOfProduct* соответственно, как показано на рисунке 2.1, и хранит информацию о название товара, его характеристиках, цене закупки и продажи. Данные классы содержат конструктор и ряд методов, позволяющих установить новые значения полей класса и получить текущие значения этих полей, методы вывода и проверки информации.

Класс *InitialMigration*, наследуемый от класса *Migration*, необходим для миграции базы данных. Миграция, по сути, представляет план перехода базы данных от старой схемы к новой. Для использования миграций в проект был добавлен пакет *Microsoft.EntityFrameworkCore.Tools* через менеджер *Nuget*. Также для работы с миграцией были проделаны команды в Консоли диспетчера пакетов.

Класс *InsertDatabase* необходим для заполнения базы данных стандартными значениями, он используется при самом первом запуске программы, затем данные уже имеются и обновляются самим пользователем.

Класс *ReceiveBdInformation* используется для получения данных из базы данных, а также их сортировки.

Класс *Menu* представляет собой главное меню и имеют ссылки на классы *MenuOutputOfCompanyInformation* и *MenuWorkWithInformation*, первый их них представляет собой меню вывода данных и использует класс *Output*, который служит для вывода информации программы, второй представляет собой меню для работы с данными использует классы *MenuWorkWithEmployee*, *MenuWorkWithDepartment*, *MenuWorkWithAddress*, *MenuWorkWithPost*, *MenuWorkWithProduct*, *MenuWorkWithCategoryOfProduct*, которые используются для работы с сотрудниками, товарами, отделами, должностями, складами и категорией товаров соответственно. Данные классы представляют собой меню для работы с определенными данными, а также включают ряд методов, таких как добавление, изменение, удаление, вывод и сортировка.

# 3 **ПРОГРАММНАЯ РЕАЛИЗАЦИЯ ИЕРАРХИИ КЛАССОВ**

### **3.1 Описание процесса реализации**

При запуске программы выполняется подключение к базе данных, затем вызывается класс *Menu* и его метод *MainMenuAsync*(), где выводится главное меню и пользователю предлагается выбрать необходимый пункт для дальнейших действий.

При выборе пункта «Вывести информацию о информационной системе», вызывается класс *MenuOutputOfCompanyInformation*,который представляет собой меню вывода данных и использует класс *Output*, который служит для вывода информации программы. Данное меню включает в себя пункты: «Вывести информацию об отделах компании», «Вывести информацию о сотрудниках компании», «Вывести информацию о складах компании и их товарах», «Выход». Которые соответственно используют методы класса *Output:* *OutputDepartments*, *OutputEmployees*, *OutputWarehouse*. При выборе пункта «Выход», пользователя возвращает в главное меню.

При выборе пункта «Работа с информацией», вызывается класс *MenuWorkWithInformation*,который представляет собой меню для работы с данными. Данное меню включает в себя пункты:

1. Работа с сотрудниками.
2. Работа с товарами.
3. Работа с отделами.
4. Работа с должностями.
5. Работа со складами.
6. Работа с категориями товаров.
7. Выход.

При выборе определенного пункта используются соответствующие классы: *MenuWorkWithEmployee*, *MenuWorkWithProduct*, *MenuWorkWith-Department*, *MenuWorkWithPost*, *MenuWorkWithAddress*, *MenuWorkWith-CategoryOfProduct*. Данные классы представляют собой меню для работы с определенными данными, а также включают ряд методов, таких как добавление, изменение, удаление, вывод и сортировка. Далее выполняются соответсвующие запросу действия и работа с информацией, после чего база данных сохраняется.

При выборе пункта «Завершить работу программы» в главном меню программа завершает свою работу.

### **3.2 Описание работы программы**

При запуске программы пользователь видит главное меню  
(рисунок 3.1)

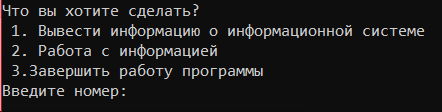


Рисунок 3.1 – Главное меню

Далее пользователю предлагается выбрать необходимый пункт.

В случае, если пользователь введёт номер числа, который не указан в меню, либо любой другой символ, программа выдаст ошибку, что данного номера не существует и предложит ввести номер заново. Данная проверка работает везде, где необходим ввод (рисунок 3.2).

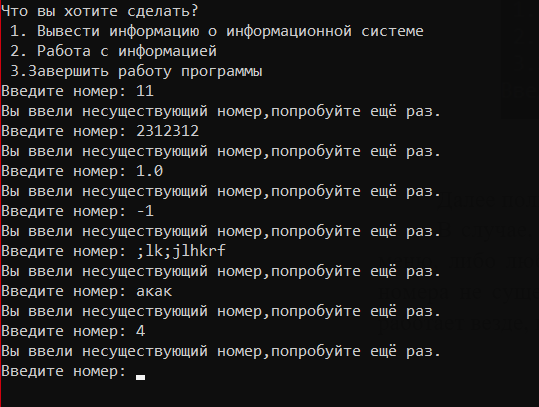


Рисунок 3.2 – Неудачная попытка

После выбора пункта «Вывести информацию о информационной системе» пользователю выводится меню вывода (рисунок 3.3).

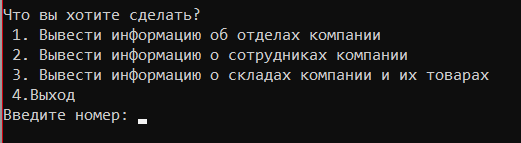


Рисунок 3.3 – Меню вывода

После выбора пункта «Вывести информацию об отделах компании», то пользователю выводится отсортированная информация об отделах компании (рисунок 3.4).

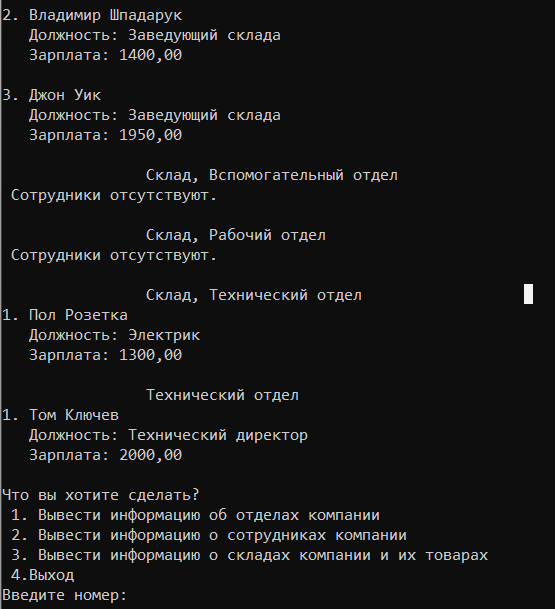
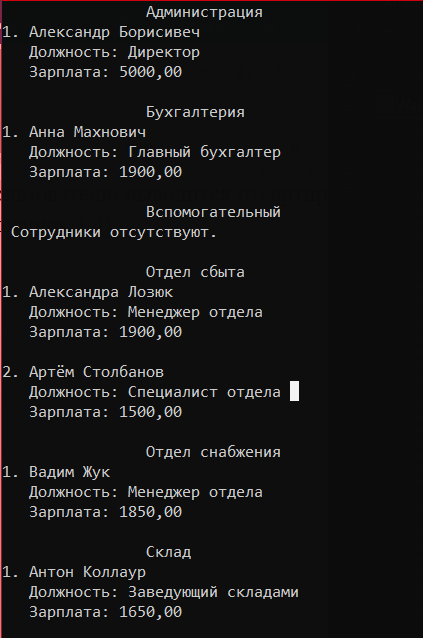


Рисунок 3.4 – Отделы компании

Если пользователь выберет пункт «Вывести информацию о сотрудниках компании», тогда пользователю выведется отсортированная информация о сотрудниках компании (рисунок 3.5).

После выбора пользователем пункта «Вывести информацию о складах компании и их товарах», тогда пользователю выведется отсортированная информация о складах компании и их товарах (рисунок 3.6).

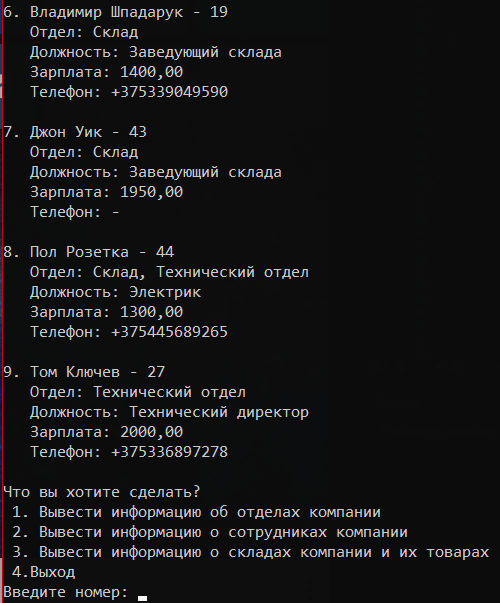
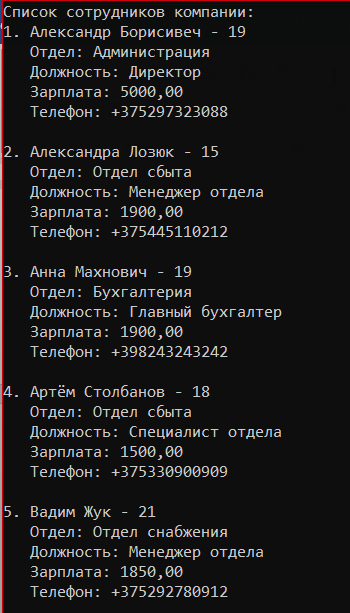


Рисунок 3.5 – Список сотрудников

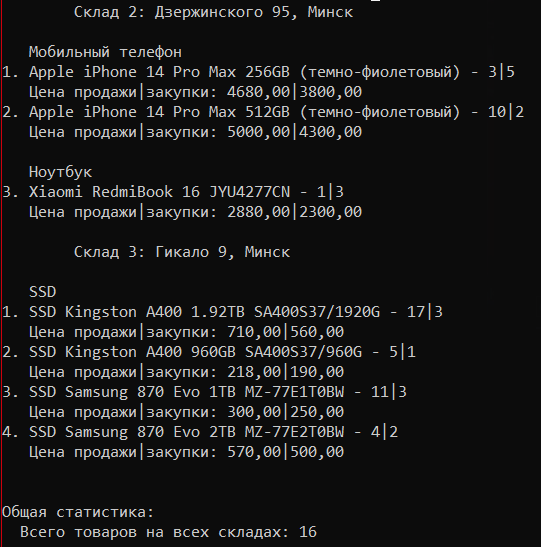
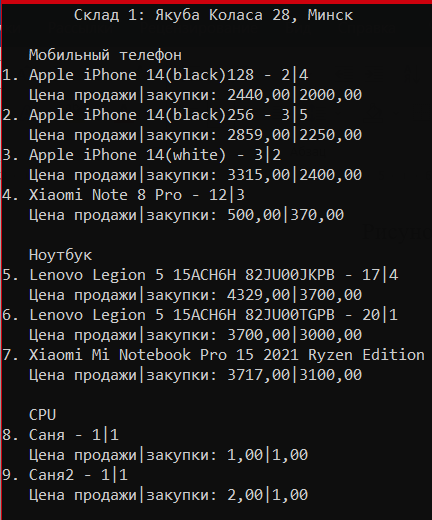


Рисунок 3.6 – Склады компании

В случае, если пользователь захочет вернутся назад в главное меню, то он может в любой момент выбрать пункт «Выход» (рисунок 3.5).

Если пользователь в главном меню выберет пункт «Работа с информацией» пользователю выведется меню для работы с данными системы, результат можно увидеть на рисунке 3.7.

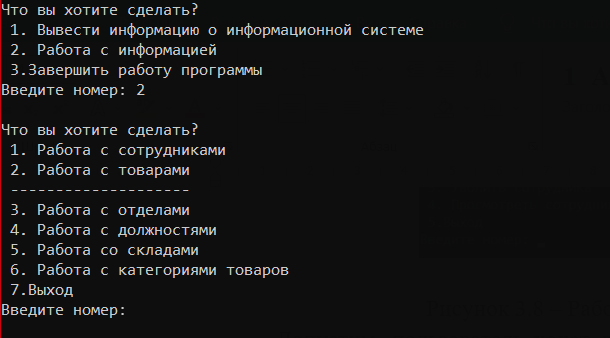


Рисунок 3.7 – Меню работы с информацией

Если пользователь далее выберет пункт «Работа с сотрудниками» пользователю выведется меню для работы сотрудниками, которое включает в себя добавление, изменение, увольнение, просмотр списка сотрудников и соответственно выход в меню на уровень ниже, результат можно увидеть на рисунке 3.8.

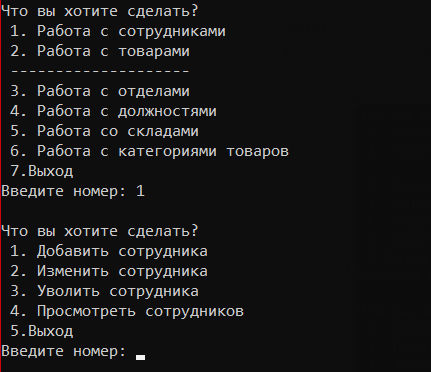


Рисунок 3.8 – Работа с сотрудником

Допустим, пользователь хочет добавить нового сотрудника, выберем пункт «Добавить сотрудника», результат представлен на рисунках 3.9 – 3.11.

Для того чтобы проверить, добавился наш сотрудник или нет, выберем пункт «Просмотреть сотрудников» и убедимся в этом (рисунок 3.12).

Теперь попробуем изменить нашего нового сотрудника, для этого выберем пункт в меню «Изменить сотрудника» и выберем необходимого, к примеру, повысим его в должности и увеличим ему зарплату, результат представлен на рисунках 3.13 – 3.15.

Стоит отметить, что мы можем изменять не только зарплату и должность, а также ещё имя, фамилию, возраст, отдел и телефон. И сразу же, для удобства, в этом меню присутствует пункт «Просмотреть результат» для быстрого просмотра необходимого результата.

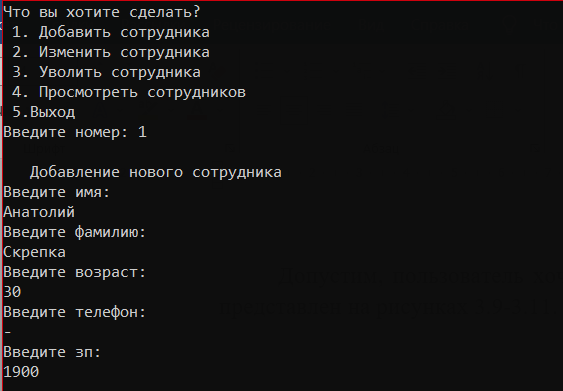


Рисунок 3.9 – Ввод данных сотрудника

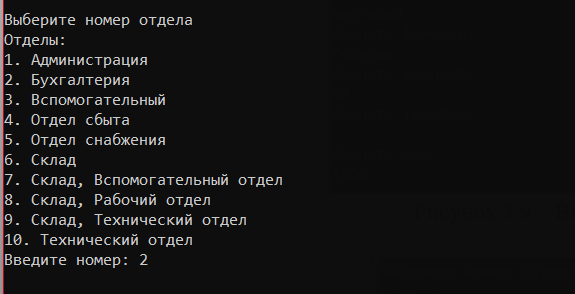


Рисунок 3.10 – Выбор отдела сотрудника

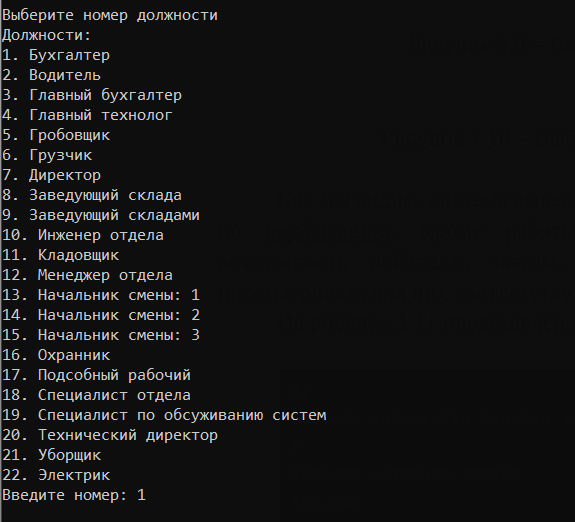


Рисунок 3.11 – Выбор должности сотрудника

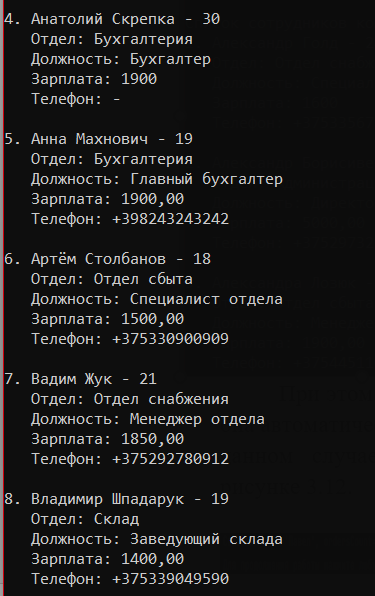
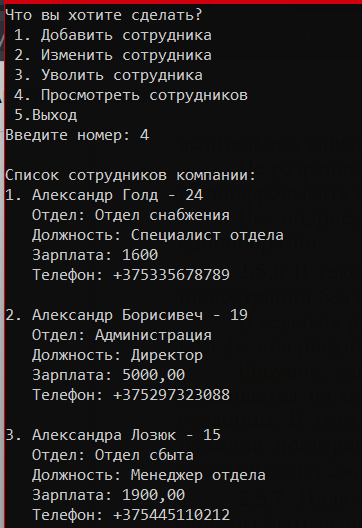


Рисунок 3.12 – Проверка добавленного сотрудника

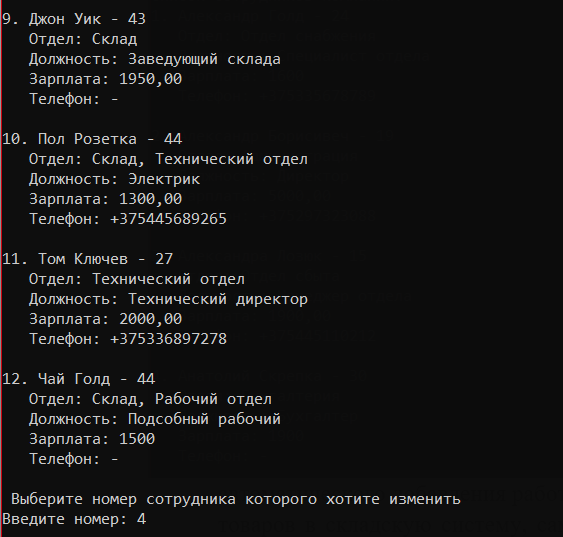
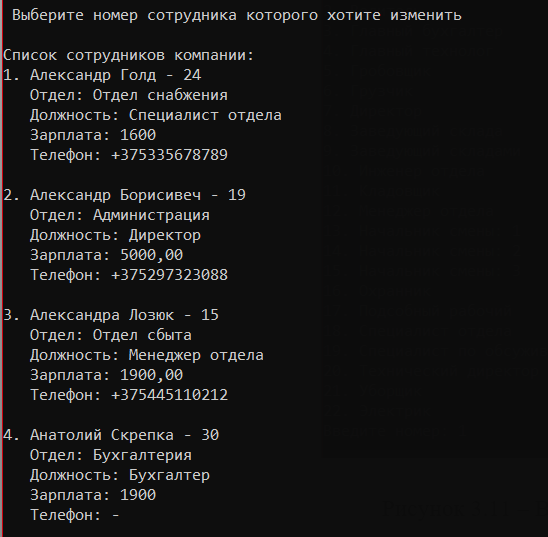


Рисунок 3.13 – Выбор сотрудника

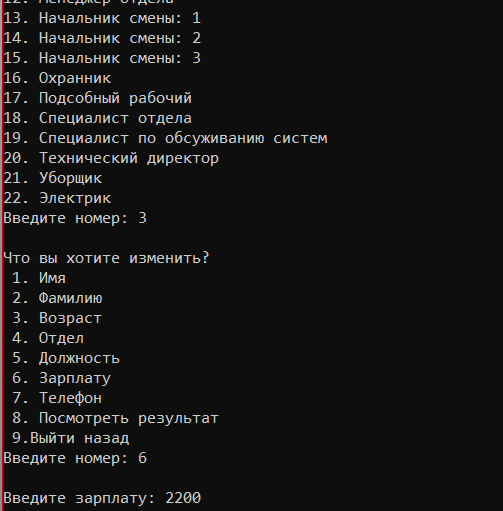
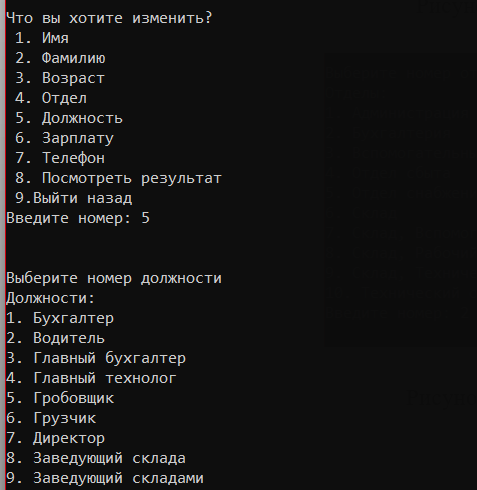


Рисунок 3.14 – Изменение данных сотрудника

Также, мы можем увольнять сотрудников, для этого необходимо выбрать пункт «Уволить сотрудника». Далее действуем, как и в алгоритме выше, выбираем сотрудника (рисунок 3.13), а затем просмотрим результат. К примеру, уволим сотрудника «Александр Голд», результат представлен на рисунке 3.16.

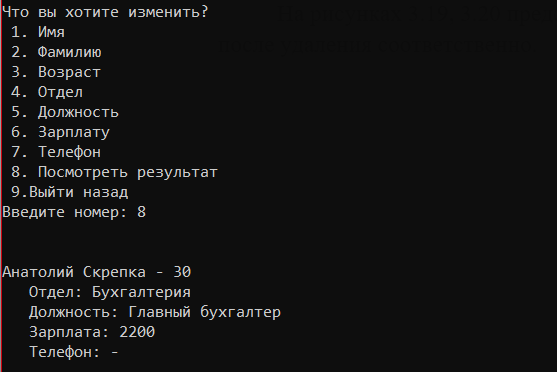


Рисунок 3.15 – Результат после изменения

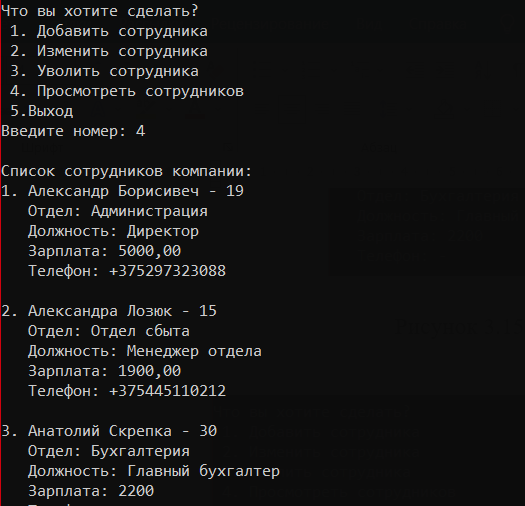
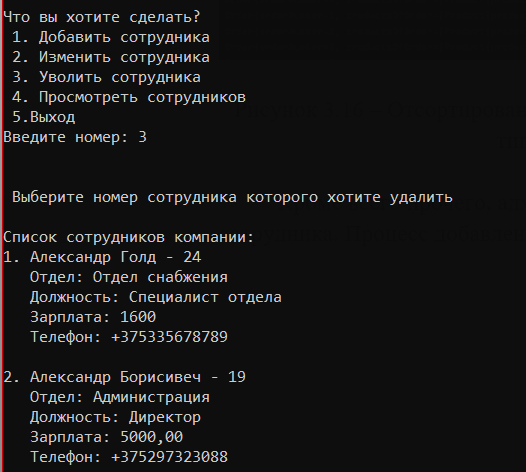


Рисунок 3.16 – Увольнение сотрудника

Также, для удобства в этом меню был добавлен пункт «Просмотреть сотрудников», он аналогичен тому, который был показан на рисунке 3.5, список сотрудников также отсортирован в удобном для пользователя виде.

Если пользователь в меню работы с информацией выберет пункт «Работа с товарами» пользователю выведется меню для работы с товарами на складах, результат можно увидеть на рисунке 3.17.

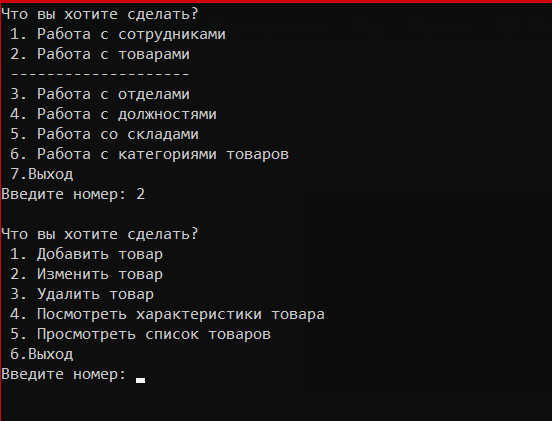


Рисунок 3.17 – Меню работы с товарами

Добавим новый товар, для этого нам необходимо будет ввести название товара, цену, характеристики, выбрать категорию товара и склад, а затем номер полки и стеллажа на этом складе (рисунок 3.18).

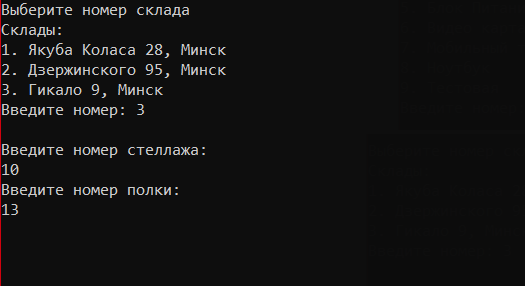
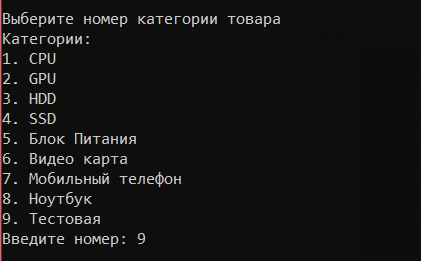
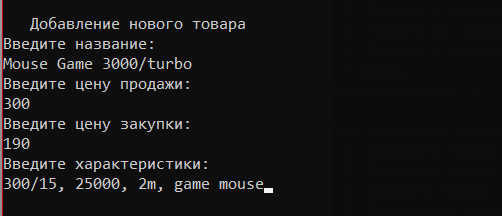


Рисунок 3.18 – Добавление нового товара

Здесь алгоритм схож с алгоритмом работы с сотрудниками, здесь мы также можем изменять товар (рисунок 3.19), удалять, просмотреть подробные характеристики (рисунок 3.20) и увидеть весь список товаров на складах, как и на рисунке 3.6.

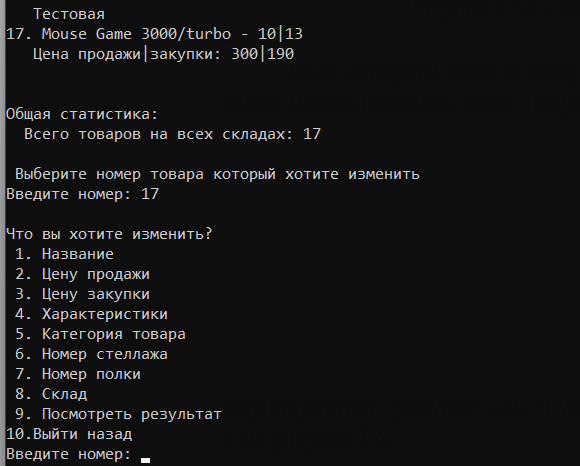


Рисунок 3.19 – Изменение товара

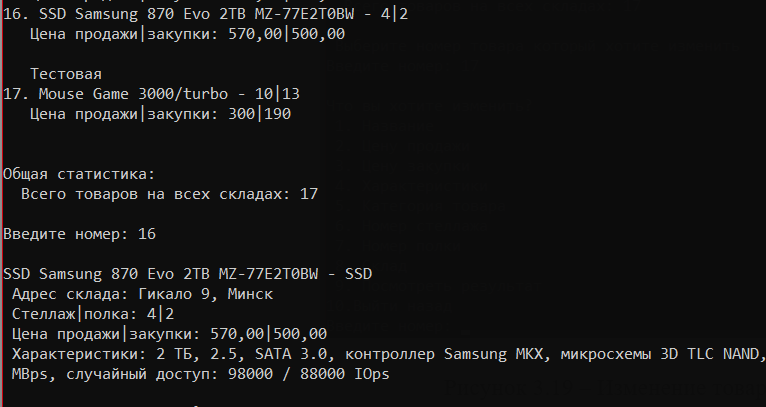


Рисунок 3.20 – Подробные характеристики товара

Процесс работы в меню «Работа с информацией» с отделами, должностями, складами и категорией товара, аналогичен работе с сотрудниками и товарами. Стоит отметить только, что главной их особенностью является то, что в данных меню можно добавить новый отдел, должность, склад и категорию товара, а до этого мы выбирали из уже существующих в базе данных. Это сделано для того, чтобы каждый раз пользователь не добавлял одинаковую информацию с одним и тем же наименованием. Также в этих меню мы можем добавлять, изменять, удалять и просматривать отдельно наши данные, что очень удобно и делают нашу систему более автоматизированной.

После выполнения каждого из запросов система возвращает пользователя в соответствующее меню. При необходимости пользователь выбирает следующее действие, которое желает выполнить, пока он не выполнит все требующиеся мероприятия. После любого взаимодействия с информацией, которая хранится в базе данных, происходит её сохранение. Для завершения работы программы необходимо вернуться в главное меню и выбрать пункт «Завершить работу программы».

# ЗАКЛЮЧЕНИЕ

В результате выполнения курсовой работы были изучены одни из популярных информационных систем склада:

* «*CRM* Склад *Creatio* для автоматизации складского учета».
* *«TopLog WMS* – система управления складом».
* «Складской учет товаров онлайн | МойСклад».
* «*LEAD WMS*.Система управления складом».

Был проведен анализ интерфейса и функционала работы этих информационных систем, в результате данного анализа за основу для построения информационной системы склада предприятия использовалась система «*TopLog WMS* – система управления складом».

В результате в разработанной информационной системе были реализованы все принципы объектно-ориентированного подхода. Данная система была разработана с целью облегчения и автоматизации работы склада. Таким образом, данная информационная система решает ряд следующих задач:

* работу со складами и товарами, а также категорией товара и его расположением;
* работу с сотрудниками, отделами, должностями;
* предоставлении полой информации о складах и сотрудниках;
* подсчет и предоставление различной статистики о товарах, продажах, закупках;
* хранение информации.

Основные преимущества данной информационной системы:

1. Удобный интерфейс для работы.
2. Изменение данных, которое можно осуществлять как в самой программе, так и в базе данных.
3. Программа может обрабатывать несколько запросов одновременно и независимо.
4. Возможность смены действующей системы управления базой данных, не изменив код приложения. Это достигается тем, что *Entity Framework* предоставляет универсальный *API* для работы с данными, поэтому основные изменения в проекте будут касаться прежде всего конфигурации и настройки подключения к соответствующим провайдерам.

Задачи, поставленные в курсовой работе, такие как разработка консольного приложения, основанного на спроектированной иерархии классов, создание и подключение базы данных, а также её настройка были выполнены.

Главная задача, которая включала в себя освоение новых современных технологий на языке *C***#** для работы с базами данных, а именно *MySQL* и *Entity Framework Core* фреймворк для работы с ней, и применение полученного опыта при выполнение поставленных задач, была успешна выполнена.

# Список использованных источников

[1] Шилдт, Г. *C*# 4.0: полное руководство /Г. Шилдт – Москва: Вильямс, 2011. – 1056 с.

[2] Рихтер, Дж. *CLR via C*#/Дж. Рихтер – СПб.: Питер, 2008. – 656 с.

[3] *CRM* Склад *Creatio* для автоматизации складского учета [Электронный ресурс]. – Режим доступа: *https://btbpm.com/sklad*

[4] Албахари, Дж. *С*# 8.0 Карманный справочник/Дж. Албахари – СПб: Питер, 2020. – 224 с.

[5] Фримев, А. *Entity Framework Core* 6 для профессионалов/А. Фримев – СПб.: Питер, 2019. - 624 с.

[6] Клири, С. Конкурентность в *C*#. Асинхронное, параллельное и многопоточное программирование/С. Клири – СПб.: Питер, 2020. – 272 с.

[7] Документация по .*NET* | *Microsoft Learn* [Электронный ресурс]. – Режим доступа: *https://learn.microsoft.com/ru-ru/dotnet/*

[8] Руководство по *Entity Framework Core* 7 [Электронный ресурс]. – Режим доступа: *https://metanit.com/sharp/efcore/*

[9] *Entity Framework Core* | *Microsoft* *Learn* [Электронный ресурс]. – Режим доступа: *https://learn.microsoft.com/en-us/ef/core/*

[10] СТП 01-2017. Стандарт предприятия. Дипломные проекты (работы). Общие требования. - Минск: БГУИР, 2017. - 169 с.

# ПРИЛОЖЕНИЕ A

**(обязательное)**

**Программный код**

*https://github.com/Dmitriy123654/Warehouse-accounting-application*