**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ ДОНЕЦКОЙ НАРОДНОЙ РЕСПУБЛИКИ Государственное образовательное учреждение высшего профессионального образования «ДОНЕЦКИЙ НАЦИОНАЛЬНЫЙ УНИВЕРСИТЕТ»**

Студент группы ИВТ-6

1 уск. курса

Сафин Егор Игоревич

ЛАБОРАТОРНАЯ РАБОТА №2

По «Технологии разработки программного обеспечения»

«Информационная система магазина Mr.Comp»

Донецк 2020

ЛАБОРАТОРНАЯ РАБОТА №2

Тема: Проектирование и макетирование программного продукта.

Цель: Научиться проектировать простейшие системы и составлять документацию по проектированию программного продукта.

В свою очередь магазин компьютерной техники осуществляет реализацию продукции, т.е. занимается доведением товара до потребителя. Данный магазин —– юридически самостоятельная организация, деятельность которой направлена на продажу товаров компьютерной сферы различных производителей, с которыми был заключён договор на право представлять данную марку той или иной компьютерной продукции.

Для компьютерного магазина основными бизнес-процессами являются:

· закупка и поставка продукции в магазин;

· представление имеющейся продукции покупателю;

· продажа компьютерной продукции.

Закупка и поставка продукции в магазин является основным начальным процессом для, непосредственно, самой продажи продукции.

Представление имеющейся продукции покупателю включает: выставление товара на витрины магазина, предоставление характеристик соответствующего товара, а также предоставление возможности покупателю протестировать товар.

Продажа компьютерной продукции включает: заключение договоров купли-продажи, заявок на предоставление дополнительных услуг, и соответственно подготовку товара к выдаче.

Вспомогательными бизнес-процессами в магазине компьютерной техники являются:

· поддержка сайта данного магазина;

· распространение рекламы;

· создание тестовых стендов.

Поддержка сайта включает в себя: обеспечение бесперебойной работы сайта, улучшение пользовательского интерфейса, разработка функционала, обновление каталога продукции.

Распространение рекламы состоит из: привлечения потенциальных клиентов различными способами (в сети Интернет, на билбордах, с помощью листовок, визиток и тп.).

Создание тестовых стендов —– оборудование стенда для возможности протестировать товар клиентом.

Продажа компьютерной продукции. Данный процесс делится на несколько подпроцессов:

1) выбор компьютерной техники;

2) оформление покупки;

3) оплата товаров;

4) выдача товаров.

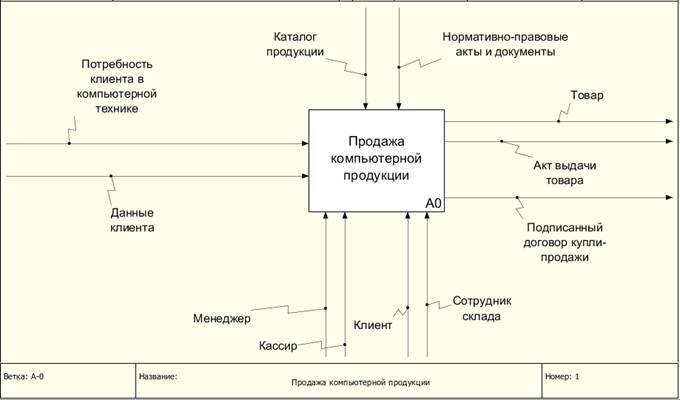
Подпроцесс «Выбор компьютерной техники» состоит из: просмотра каталога, консультации с менеджером, осмотра презентационных товаров, принятия решения о совершении покупки, проверки наличия товаров на складе. Основными действующими лицами являются менеджер и клиент.

Подпроцесс «Оформление покупки» включает в себя: заполнение данных клиента, печать заказа на продажу, печать заявки на оказание услуг, подписание заявки и заказа. Основными действующими лицами являются клиент и менеджер.

Подпроцесс «Оплата товаров» состоит из: подачи документов на кассу, оплаты товаров на кассе и подписания договора купли-продажи. Основными действующими лицами являются кассир и клиент.

Подпроцесс «Выдача товара» представляет собой составление и заключение акта выдачи товара и передача его покупателю. Действующие лица —– покупатель, сотрудник склада.

На рисунке 1.1 представлена контекстная диаграмма бизнес-процесса «Продажа компьютерной продукции». Данная диаграмма представляет набор входной, управляющей, выходной информации, а также механизмов управления.



**Рисунок 1.1 – Контекстная диаграмма модели “AS-IS” «Продажа компьютерной продукции»**

**Структура проекта**

1) Логика приложения (классы и соответствующие файлы .h и .cpp):

1.Любой пользователь

2. Сотрудники

3.Администратор

4.Директор

5.Чек

6 .Настройка

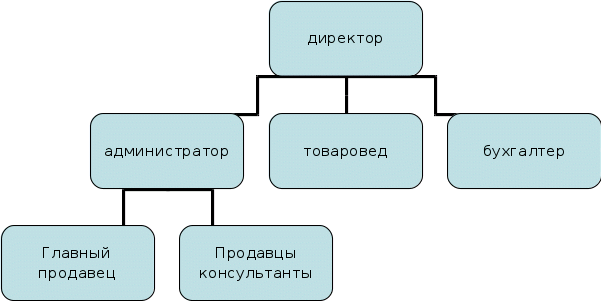
2) Окна и GUI-элементы (классы и соответствующие файлы .h и .cpp):

Главная форма Доступ к основным функциям

BD Основное окно, из него можно произвести все доступные действия с БД

Settings Доступ к различным инструментам, для изменения интерфейса, работы БД.

PrintCheck Окно печати чека

Рис.1.2 

Общая схема классов уровня логики приложения приведена на Рис.1.2

**Макет интерфейса программного продукта**

Макет – это статическое визуальное более-менее реалистичное представление концепции пользовательского интерфейса.

Условно можно разделить все макеты на две группы: **wireframes** и **mockups**. К сожалению, и те, и другие в русском языке называются одинаково, что приводит к путанице. Следует, однако, отметить, что иногда и английские названия используют в качестве синонимов, что также ясности не добавляет.

**Для чего можно использовать макеты**

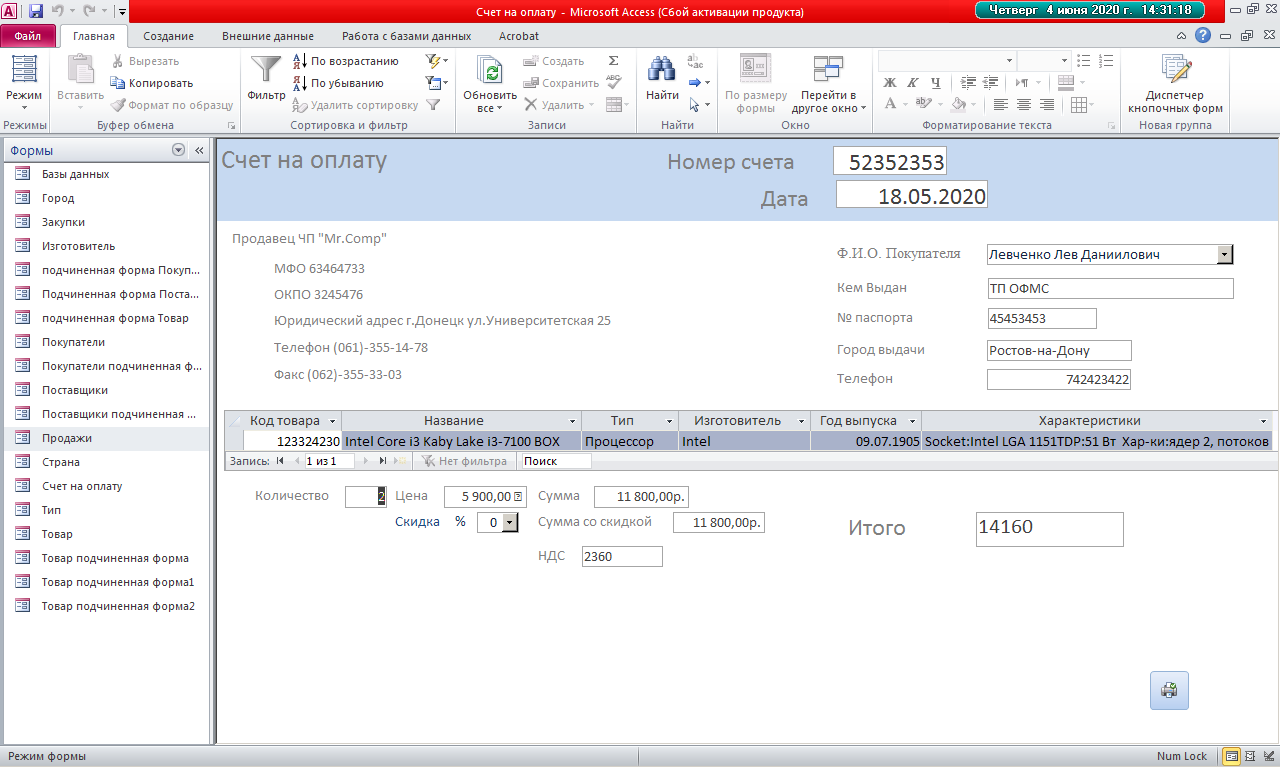
1) **Обсуждение функциональности с заказчиком**. Заказчики, как правило, люди очень занятые и зачастую физически не способные выделить время на чтение многостраничных документов (даже если это в их интересах). В этом смысле картинка стоит тысячи слов: она гарантирует, что у заказчика появится четкое представление о том, что именно будет сделано.

2)**Оценка юзабилити**. Макеты – это фактически первый артефакт в проекте, юзабилити которого уже можно и нужно оценивать. На этом этапе проще и дешевле всего устранить проблемы, если они обнаружатся.

3) **Постановка задачи разработчикам**. Попробуйте меня переубедить, но я все же считаю, что очень маленький процент разработчиков вчитывается в спецификации. Наличие наглядной иллюстрации того, что должно быть сделано, гораздо полезнее, чем длинное текстовое описание (хотя, разумеется, оно также должно присутствовать, ибо не все можно отобразить на макете).

4) **Постановка задачи дизайнерам**. Вы значительно облегчите дизайнерам работу, если вместо сухого сбивчивого описания страницы предоставите ему макет и свободу действий. В то же время в некоторых случаях это позволит оградиться от чрезмерного дизайнерского креатива: вы будете уверены в том, что результат будет хотя бы отдаленно похож на то, чего хотел заказчик.

5) **Тестирование**. С этим, я думаю, все понятно: сверяя макеты и созданный интерфейс, можно будет обнаружить дефекты довольно быстро.



**Рисунок 1.3 –** макет интерфейса программного продукта

**Особенности реализации и использования классов**

**User**

Абстрактный класс для создания объектов принадлежащий любому пользователю. Содержит поля: Surname, Name, Middle\_name, Age, Phone.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace MyNameSpace

{

abstract class Class1

{

string Surname { get; set; }

string Name { get; set; }

string Middle\_name { get; set; }

int Age { get; set; }

int Phone { get; set; }

}

}

**Employee**

Родительский класс, для всех должностей имеющихся в магазине имеет поля: Number – спец. номер сотрудника. Реализует методы: AddClassN, AddCheck, EditTheCheck, EditTheItem, EditTheClassBD, AccessLevel и Warning – предупреждение от администратора. Большинство методом наследуются классами потомков.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace MyNameSpace

{

class Employee: human

{

int Workplace;

int Number;

void AccessLevel() { }

void Warning() { }

void AddClassN() { }

void DeleteCheck() { }

void EditTheCheck() { }

void EditTheEitem() { }

void EditTheClassBD() {}

}

}

**Administrator**

Класс администраторов. Реализует методы: AddClassN, AddCheck, EditTheCheck, EditTheItem. Класс Администратора позволяет самому администратор выполнять большинство функций приложения, которые недоступны в реализации более низкого уровня доступа.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace MyNameSpace

{

class Administrator:Employee

{

void AddClassN() { }

void DeleteCheck() { }

void EditTheCheck() { }

void EditTheEitem() { }

}

}

**Director**

Классу Директор доступны все имеющиеся приложения, Полностью включая функционал Администратора. Реализует методы: AddClassN, AddCheck, EditTheCheck, EditTheItem, EditTheClassBD.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace MyNameSpace

{

class Director:Employee

{

void AddClassN() { }

void DeleteCheck() { }

void EditTheCheck() { }

void EditTheEitem() { }

void EditTheClassBD() {}

}

}

**Barten**

Сотруднику доступны следующие методы: AddClassN, AddCheck, EditTheCheck. Функционал сотрудника ниже, чем администратора и директора.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace MyNameSpace

{

class Barten:Employee

{

void AddClassN() { }

void DeleteCheck() { }

void EditTheCheck() { }

}

}

**Check**

Чек имеет небольшое количество функций, которые так же наследуются от Employer.

using System.Text;

namespace MyNameSpace

{

class Waiter:Employee

{

Void printCheck() {}

}

}

**Описание control flow основных частей проекта**

**Авторизация**

При открытии приложения, появляется главное окно предоставляющее следующие кнопки использования: Создать чек, Меню, а так же База данных.

При создании чека, появляются перечни данных, которые следует заполнить, после добавления новой строки в BD заносятся новые данных. Создание чека доступно для всех пользователей приложения.

Во вкладке меню находится перечень доступных для покупки товара . Которые так же находятся в базе данных, их можно добавить, редактировать, а так же удалить