Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Петрозаводский государственный университет»

Физико-технический институт

Кафедра информационно-измерительных систем и физической электроники

Командный проект «Приложение для пекарни Bekker 2.0»

Авторы работы:

студенты группы 21312

Д.Пушко

Т.Шестаков

А.Мирон

Научный руководитель:

канд. физ.-мат. наук, доцент

А.В.Бульба

Петрозаводск 2022

Цель: Разработать программу, включающую в себя интерфейс для работы в Пекарне. Включает в себя: меню владельца (босс), меню касира, меню администратора. Каждое меню должно обладать уникальными возможностями, а боссу должна выводиться информация о работе других пользователей.

Программная реализация: среда разработки QT Creator версии 5.4.2, язык программирования C++. Заголовочные файлы и их назначение:

- class.h – хранит в себе описание классов, а так же подключение всех используемых библиотек;
- function.h – содержит прототипы методов, используемых в программе.

Единицы компиляции:

- bekkerapp.cpp – основной файл, запускающий интерфейс;
- class.cpp – файл, в котором содержится реализация методов классов;
- function.cpp файл описывающий работу функций, которые вызываются во время работы программы.

Пошаговое описание процесса разработки:

- Поступила заявка от заказчика: разработать простой интерфейс для работы в пекарне. Заказчик хочет увидеть авторизацию под разный персонал, особо отметил кассира, владельца, администратора. Для каждого из персонала необходимо разработать своё меню, у кассира должны быть опции, добавления товара, удаления всего чека (в случае, если человек передумал покупать), просмотр общего чека ,а так же возможность согласовать оплату, в случае оплаты заказа, чек должен отправляться в файл с общими чеками для просмотра их

владельцем. Владелец, должен иметь опцию просматривать общие чеки, просматривать заявку на закупку продукции, которую отправил администратор, а также уметь просматривать какой процент от продажи уйдет на налог. Администратор должен составлять заказ покупок для пекарни, то есть если необходимо докупить какой-то товар, он мог внести его в заявку, а также просматривать эту заявку и сохранять её, с функцией добавления нового продукта.

- Заказчик предоставил список и цены продукции, которая будет продаваться у него в пекарни, вида Название продукта/цена/валюта
- Действующие актеры: Продавец, Администратор, Босс, Покупатель
- Варианты использования: Добавить продукт, удалить продукт, просмотр всего чека, оплата, просмотр покупок за сегодня, просмотр общих покупок, Доходы и налоги, Оформить заявку, просмотреть заявку, Сохранить заявку, Выход в меню, Выход с профиля, Зайти в профиль.

Диаграммы использования:

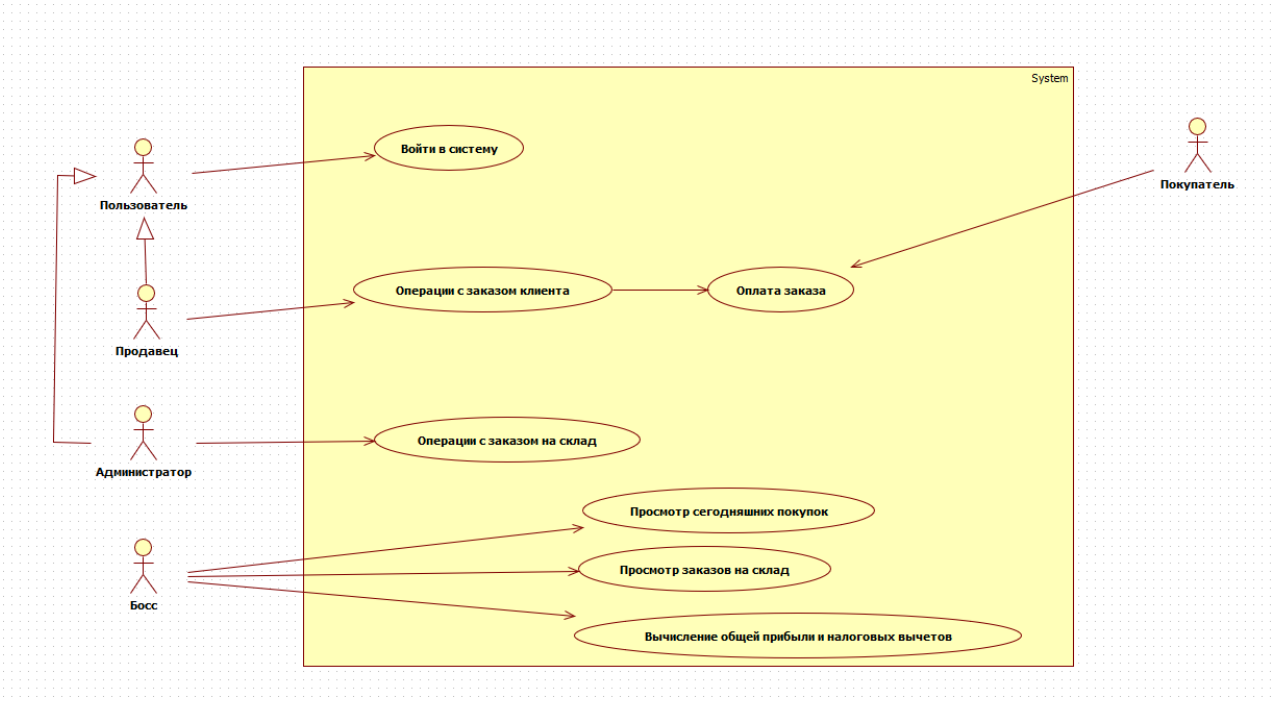

Рис.1 «Диаграмма вариантов использования «Система предприятия»»

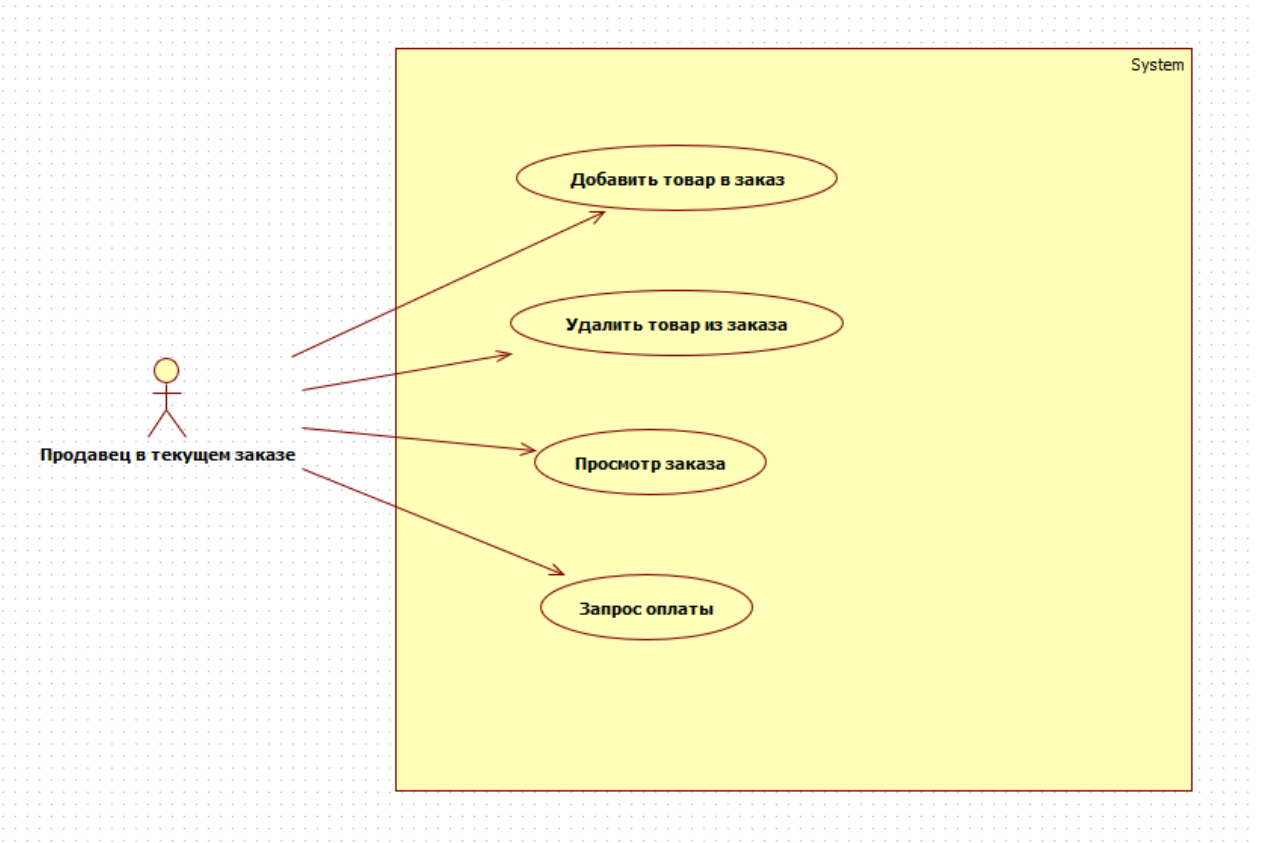

Рис.2 «Диаграмма вариантов использования «Операции с заказом клиента»»

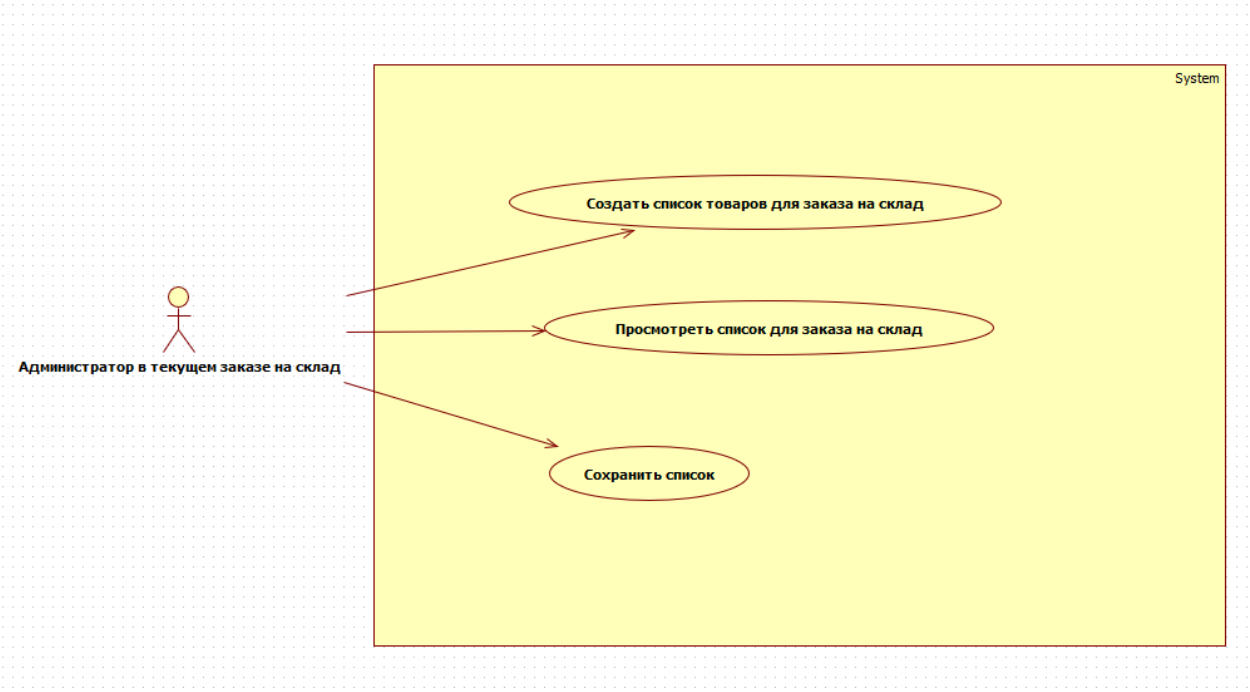

Рис.3 «Диаграмма вариантов использования «Операции с заказом на склад»»

Диаграммы активности:

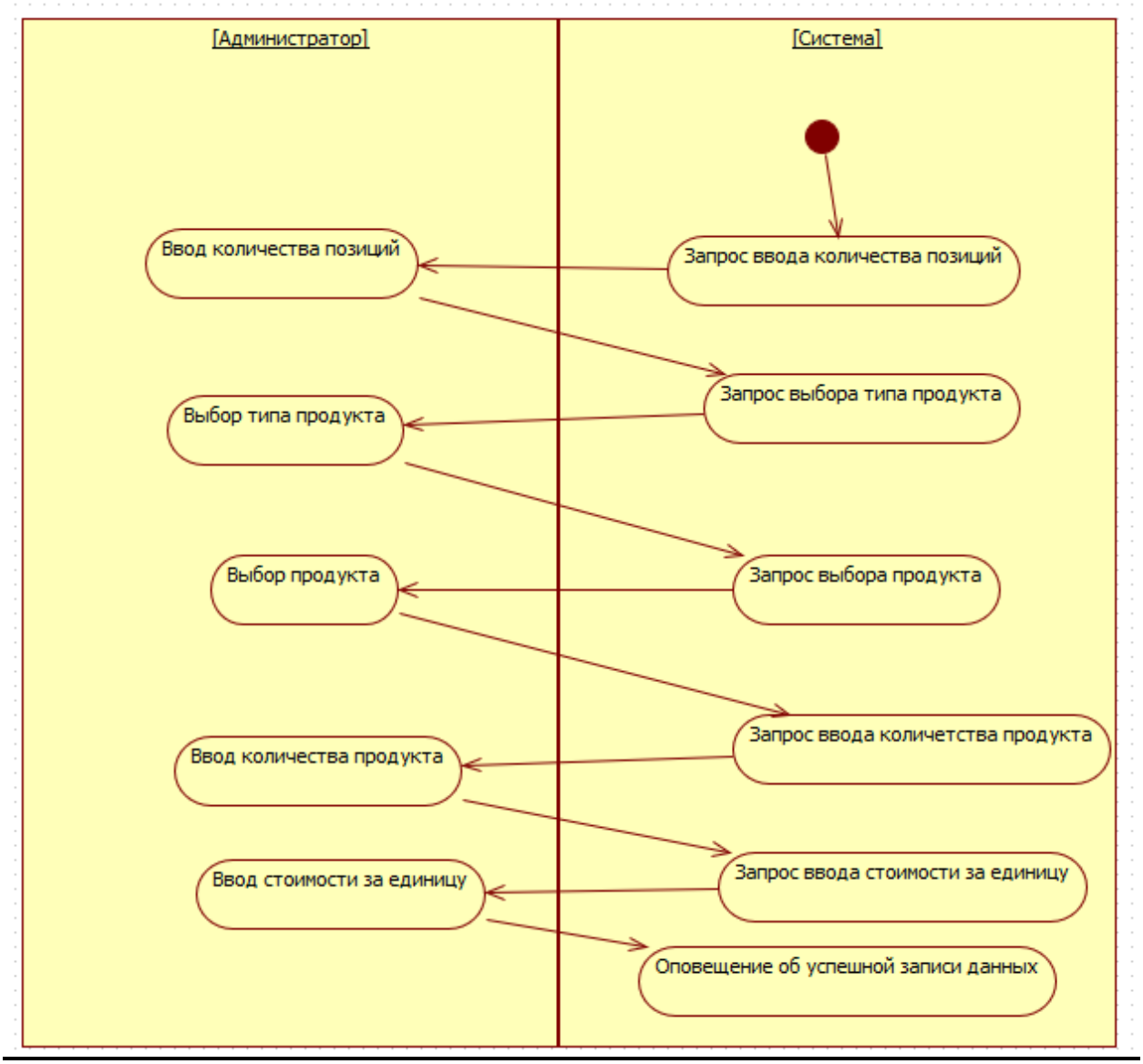

Рис.4 «Диаграмма активности для создания списка заказа на склад»

Диаграммы классов:

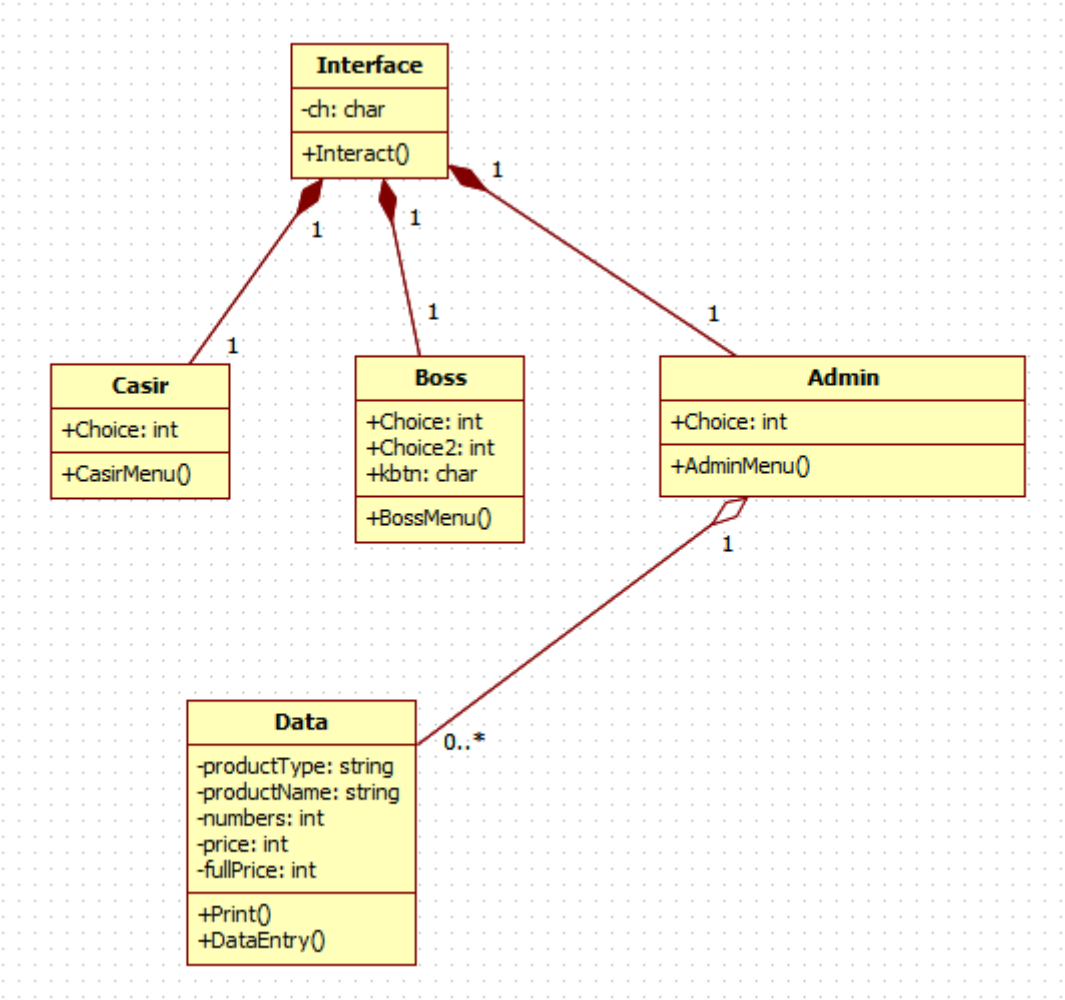

Рис.5 «Диаграмма классов, использующихся в программе»

Диаграммы последовательности:

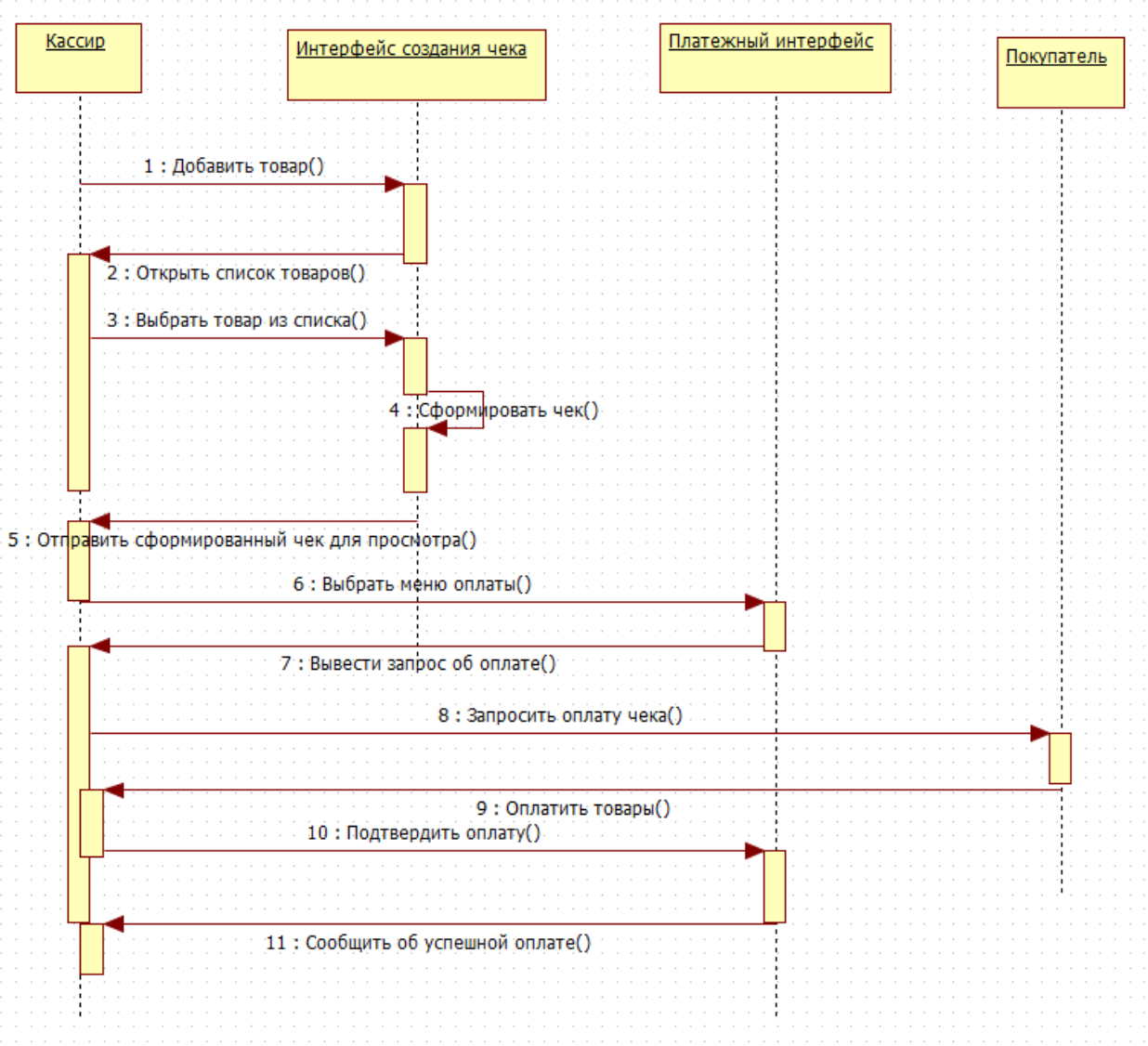

Рис.6 «Диаграмма последовательности операции купли-продажи»

- Существительные: Кассир, Админ, Босс, Интерфейс, Дата, Чек, продукт, пекарня, проверка, сложение, выход, меню, сумма, итого, цена, доход, налог
- Кассир – отвечает, за действия кассира. Админ- отвечает, за действия админа. Босс-отвечает, за действия босса. Интерфейс-отвечает за вывод меню. Дата- отвечает, за хранения данных при заполнении заявки
- Список атрибутов: Кассир/Босс/Админ(Choice – выбор в меню).Кассир(line – для работы с файлами, oplata – для выбора успешно/безуспешной оплаты)Босс(line – для работы с

файлами,s2/sum/tax/sum2 – для выборки 2 элемента в файле и подсчета суммы, налога, дохода соответственно)

- Файлы, которым обмениваются классы, это файлы: Storage Orders – хранение чеков от оплаты, Orders List – заявка от Администратора)

Классы располагаются в отдельном заголовочном файле:

Код class,h

```
 #ifndef CLASS
#define CLASS

#include<iostream>
#include<string>
#include<fstream>
#include<windows.h>
#include <sstream>
#include <stdio.h>

using namespace std;

class Casir{
public:
    Casir();
    ~Casir();
    int Choice;
    void CasirMenu();
};

class Boss{
public:
    Boss();
    ~Boss();
    char kbtn;
    void BossMenu();
private:
    int Choice;
    int Choice2;
};

class Admin{
public:
    Admin();
    ~Admin();
    int Choice;
    void AdminMenu();
};


class UserInterface
{
private:
char ch;
public:
UserInterface();
```

```
~UserInterface();
void interact();
};
struct productOrder
{
    string productType, productName;
};

struct Info
{
    int numbers, price, fullPrice;
};

class Data
{
 private:
    productOrder productorder;
    Info info;

 public:
    Data();
    Data(productOrder productorder_, Info info_);
    ~Data();

    void Print();
    void DataEntry(productOrder productorder_, Info info_);

    productOrder GetProductOrder() {return productorder;}
    Info GetInfo() {return info;}
};

#endif // CLASS
```

Код class.cpp

```
#include "class.h"
#include "function.h"

//using namespace std;

Casir::Casir(){

}
Casir::~Casir(){

}
Boss::Boss(){

}
Boss::~Boss(){

}
Admin::Admin(){

}
Admin::~Admin(){

}

void Casir::CasirMenu(){
    string line;
```

```
 int a;
 char oplata;
 tochka:
system("cls");
cout << "Sales Menu" << endl;
cout << "1. Add a product" << endl;
cout << "2. Delete product" << endl;
cout << "3. Final receipt" << endl;
cout << "4. Payment" << endl;
cout << "5. Logout" << endl;
cout << "Choice: ";
cin >> Choice;
if(Choice > 5){
    cout << "Please choose from 1 to 5" << endl;
    system("pause");
    goto tochka;
}
switch (Choice){
case 1:
    tochka2:
    system("cls");
    cout << "Choose the type of product" << endl;
    cout << "1. Cake" << endl;
    cout << "2. Bun" << endl;
    cout << "3. Drink" << endl;
    cout << "4. Exit" << endl;
    cout << "Choice: ";
    cin >> Choice;
    if(Choice > 4){
        cout << "Please choose from 1 to 4" << endl;
        system("pause");
        goto tochka2;
    }
    switch(Choice){
        case 1:
        system("cls");
    {
        ifstream in("Cake.txt"); // окрываем файл для чтения
        if (in.is_open())
        {
        while (getline(in, line))
        {
        cout << line << endl;
        }
        }
        else {
        cout << "Error! Press any button to get back to menu. " << endl;
        system ("pause");
        goto tochka2;
        }
        in.close(); // закрываем файл
        cout << "Choice: ";
        cin >> (a);
        if (a == 1){
ofstream rec;
rec.open("Order.txt", ios::app);
if (rec.is_open())
{
   rec << "Snickers_Cake" << " 212 Rub"<< endl;
}
else
    cout << "Error! Press any button to get back to menu. " << endl;
    system ("pause");
    goto tochka2;
```

```
            }

            if (a == 2){
    ofstream rec;
    rec.open("Order.txt", ios::app);
    if (rec.is_open())
    {
       rec << "Napoleon_Cake" << " 240 Rub"<< endl;
    }
    else
        cout << "Error! Press any button to get back to menu. " << endl;
        system ("pause");
        goto tochka2;
            }

            if (a == 3){
    ofstream rec;
    rec.open("Order.txt", ios::app);
    if (rec.is_open())
    {
       rec << "Cheese_Cake" << " 570 Rub"<< endl;
    }
    else
        cout << "Error! Press any button to get back to menu. " << endl;
        system ("pause");
        goto tochka2;
            }

            if (a == 4){
    ofstream rec;
    rec.open("Order.txt", ios::app);
    if (rec.is_open())
    {
       rec << "Sour_Cream_Cake" << " 164 Rub"<< endl;
    }
    else
        cout << "Error! Press any button to get back to menu. " << endl;
        system ("pause");
        goto tochka2;
            }
            goto tochka2;
            break;
        }

            case 2:
            system("cls");
        {
            ifstream in("Bun.txt"); // окрываем файл для чтения
            if (in.is_open())
            {
            while (getline(in, line))
            {
            cout << line << endl;
            }
            }
            else {
                cout << "Error! Press any button to get back to menu. " <<
endl;
                system ("pause");
                goto tochka2;
            }
            in.close(); // закрываем файл
            cout << "Choice: ";
            cin >> (a);
```

```
        if (a == 1){
ofstream rec;
rec.open("Order.txt", ios::app);
if (rec.is_open())
{
   rec << "Sausage_In_The_Dough" << " 40 Rub"<< endl;
}
else
    cout << "Error! Press any button to get back to menu. " << endl;
    system ("pause");
    goto tochka2;
        }

        if (a == 2){
ofstream rec;
rec.open("Order.txt", ios::app);
if (rec.is_open())
{
   rec << "Pizza_Application" << " 25 Rub"<< endl;
}
else
    cout << "Error! Press any button to get back to menu. " << endl;
    system ("pause");
    goto tochka2;
        }

        if (a == 3){
ofstream rec;
rec.open("Order.txt", ios::app);
if (rec.is_open())
{
   rec << "Cabbage_Pie" << " 15 Rub"<< endl;
}
else
    cout << "Error! Press any button to get back to menu. " << endl;
    system ("pause");
    goto tochka2;
        }

        if (a == 4){
ofstream rec;
rec.open("Order.txt", ios::app);
if (rec.is_open())
{
   rec << "Pie_With_Filling_Rice_Egg " << " 10 Rub"<< endl;
}
else
    cout << "Error! Press any button to get back to menu. " << endl;
    system ("pause");
    goto tochka2;
        }
        goto tochka2;
        break;
    }

        case 3:
        system("cls");
    {
        ifstream in("Drink.txt"); // окрываем файл для чтения
        if (in.is_open())
        {
        while (getline(in, line))
        {
        cout << line << endl;
```

```
            }
            }
            else {
                cout << "Error! Press any button to get back to menu. " <<
endl;
                system ("pause");
                goto tochka2;
            }
            in.close(); // закрываем файл
            cout << "Choice: ";
            cin >> (a);
            if (a == 1){
    ofstream rec;
    rec.open("Order.txt", ios::app);
    if (rec.is_open())
    {
       rec << "Tea" << " 15 Rub"<< endl;
    }
    else
        cout << "Error! Press any button to get back to menu. " << endl;
        system ("pause");
        goto tochka2;
            }
            if (a == 2){
    ofstream rec;
    rec.open("Order.txt", ios::app);
    if (rec.is_open())
    {
       rec << "Coffee" << " 35 Rub"<< endl;
    }
    else
        cout << "Error! Press any button to get back to menu. " << endl;
        system ("pause");
        goto tochka2;
            }
            if (a == 3){
    ofstream rec;
    rec.open("Order.txt", ios::app);
    if (rec.is_open())
    {
       rec << "Coca-Cola" << " 80 Rub"<< endl;
    }
    else
        cout << "Error! Press any button to get back to menu. " << endl;
        system ("pause");
        goto tochka2;
            }
            if (a == 4){
    ofstream rec;
    rec.open("Order.txt", ios::app);
    if (rec.is_open())
    {
       rec << "Fanta" << " 75 Rub"<< endl;
    }
    else
        cout << "Error! Press any button to get back to menu. " << endl;
        system ("pause");
        goto tochka2;
            }
            if (a == 5){
    ofstream rec;
    rec.open("Order.txt", ios::app);
    if (rec.is_open())
    {
```

```
   rec << "Sprite" << " 70 Rub"<< endl;
}
else
    cout << "Error! Press any button to get back to menu. " << endl;
    system ("pause");
    goto tochka2;
        }
        goto tochka2;
        break;
    }
    case 4:
        goto tochka;
    }
    system("pause");
    system("cls");
case 2:
{
    del:
    system("cls");
    cout << "Delete product:"<< endl;
    ifstream in("Order.txt"); // окрываем файл для чтения
    if (in.is_open())
    {
    while (getline(in, line))
    {
    cout << line << endl;
    }
    }
    cout << "" << endl;
    cout << "Choose main item:" << endl;
    cout << "1. Delete all order" << endl;
    cout << "2. Exit" << endl;

    cout << "Choice: ";
    cin >> Choice;
    if(Choice > 2){
        cout << "Please choose from 1 to 2" << endl;
        system("pause");
        goto del;
    }
    switch(Choice){
          case 1:{
        ofstream outfile;
        outfile.open("Order.txt", ofstream::out | ofstream::trunc);
        outfile.close();

    }

          case 2:{
            goto tochka;
    }
    }

    break;
}
case 3:
    system ("cls");
{
    ifstream out("Order.txt");
    if (out.is_open()){
    while (getline(out,line)){
    cout << line << endl;
    }
    }
```

```
    out.close();
    string s1,tail;
    int s2 = 0;
    int sum = 0;
    ifstream out1("Order.txt");
    if (out1.is_open()){
    while(out1>>s1>>s2)// в s1 - первое слово, в s2 - второе
    {
        s2 = s2;
        getline(out1,tail); // дочитываем остаток строки
        sum += s2;

    }
    cout <<"Result: " <<sum << endl;
    }
    else
        cout << "Error! Press any button to get back to menu. " << endl;
        system ("pause");
        goto tochka;
    out1.close();
    system("pause");
    goto tochka;
    break;
}
case 4:
{
    paid:
    system("cls");
    cout << "The buyer paid the receipt? (y/n)" << endl;
    cin >> oplata;
    if (oplata == 'y' || oplata == 'n'){
    if (oplata == 'y'){

        ifstream rf("Order.txt");
         if (!rf)
         {
           cout << "Cannot open source file." << endl;
         }
         ofstream wf("Storage Orders.txt",ios_base::app);
         if (!wf)
         {
           rf.close();
           cout << "Cannot open the files." << endl;
         }
         char sym;
          while (rf)// цикл посимвольного чтения
            {
           rf.get(sym); // считать из rf => sym
           if (rf)
             wf << sym; // записать sym => wf
            }
         rf.close();
         rf.open("Order.txt", ofstream::out | ofstream::trunc);
         rf.close();
         wf.close();
         goto tochka;
       }
    if (oplata == 'n'){
        ofstream outfile;
        outfile.open("Order.txt");
        outfile.close();
        goto tochka;
    }
    }else{
```

```
            cout << "Wrong parametr, repeat choice" << endl;
            system("pause");
            goto paid;
        }
        break;
}
    case 5:
    {
        system("cls");
        UserInterface theUserInterface;
        theUserInterface.interact();
    }
}
}

void Boss::BossMenu(){
    string line;
    string s1 = "",
           tail = "";
    int s2 = 0;
    double sum = 0;
    int tax = 0;
    int sum2 = 0;
    mainmenu:
    system("cls");
    cout << "Sales Menu" << endl;
    cout << "1. Today buyings" << endl;
    cout << "2. Storage orders" << endl;
    cout << "3. Income and taxes" << endl;
    cout << "4. Logout" << endl;
    cout << "Choice: ";
    cin >> Choice;
    if(Choice > 4){
        cout << "Please choose from 1 to 4" << endl;
        system("pause");
        goto mainmenu;
    }
    switch (Choice){
    case 1:
    {
        system("cls");
        cout << "Today buyings" << endl;
        ifstream out("Storage Orders.txt");
        if (out.is_open()){
        while (getline(out,line)){
        cout << line << endl;
        }
        }
        out.close();
        int s2 = 0;
        int sum = 0;
        ifstream out1("Storage Orders.txt");
        if (out1.is_open()){
        while(out1>>s1>>s2)// в s1 - первое слово, в s2 - второе
        {
            s2 = s2;
            getline(out1,tail); // дочитываем остаток строки
            sum += s2;

        }
        cout <<"Result: " <<sum << endl;
        }
        system("pause");
        system("cls");
```

```
    goto mainmenu;
    break;
}
case 2:
{
    prev:
    system("cls");
    cout << "Orders menu" << endl;
    cout << "1. Show orders" << endl;
    cout << "2. Exit to main menu" << endl;
    cout << "Choice: ";
    cin >> Choice2;
    if(Choice2 > 2){
        cout << "Please choose from 1 to 2" << endl;
        system("pause");
        goto prev;
    }
    switch(Choice2){
        case 1:
    {
            ifstream in("OrderList.txt"); // окрываем файл для чтения
            if (in.is_open())
            {
                system("cls");
                while (getline(in, line))
                {
                    cout << line << endl;
                }
            }
            else {
                cout << "error no such file";
            }
            system("pause");
            system("cls");
            goto prev;
            in.close();     // закрываем файл
            break;
    }

        case 2:
    {
        system("cls");
        Choice = 0;
        goto mainmenu;
        break;
        }
          default: cout << "Wrong parameter!!";
    }
    break;
}
case 3:
{
    system("cls");
    ifstream out1("Storage Orders.txt");
    if (out1.is_open()){
        sum = 0;
        s2 = 0;
    while(out1>>s1>>s2)// в s1 - первое слово, в s2 - второе
    {
        s2 = s2;
        getline(out1,tail); // дочитываем остаток строки
        sum += s2;
    }
    tax = (sum/100)*6;
```

```
            sum2 = sum - tax;
            }
            cout << "There's Income and Taxes of our company" << endl;
            cout <<"Income without taxes: " <<sum << endl;
            cout <<"Taxes(6%): " <<tax << endl;
            cout <<"Income: " <<sum2 << endl;

            system("pause");
            system("cls");
            goto mainmenu;
            break;
        }
        case 4:
        {
            system("cls");
            UserInterface theUserInterface;
            theUserInterface.interact();
        }
        default:
            cout << "Wrong parameter!";
                break;
        }
}
void Admin::AdminMenu(){

    int _size = 0;
    Data* d = new Data[_size];

    int choice;
    admmenu:
    system("cls");
    cout << "Administrator menu" << endl;
    cout << "1. Create a list to order" << endl;
    cout << "2. Check data" << endl;
    cout << "3. Save data" << endl;
    cout << "4. Logout" << endl;
    cout << "Choice: " << endl;
    cin >> choice;
    if(choice > 4){
        cout << "Please choose from 1 to 4" << endl;
        system("pause");
        goto admmenu;
    }
    switch(choice){
    case 1:
    {   system ("cls");
        DataEntry(d, _size);
        system("pause");
        system("cls");
        goto admmenu;
        break;
    }
    case 2:
    {
        system("cls");
        if (_size !=0)
            Print(d, _size);
        else
            cout << "Data is empty!" << endl;
        system("pause");
        system("cls");
        goto admmenu;
        break;
    }
```

```
   case 3:
   {
        system("cls");
        if (_size !=0 )
        SaveData(d, _size);
        else
            cout << "Data is empty!" << endl;
        system("pause");
        system("cls");
        goto admmenu;
        break;
   }
   case 4:
   {
        system("cls");
        UserInterface theUserInterface;
        theUserInterface.interact();
   }
   } // switch
}

UserInterface::UserInterface()
{

}
//-------------------------------------------------------------
UserInterface::~UserInterface()
{

}
//-------------------------------------------------------------

void UserInterface::interact()
{
start:
system("cls");
cout << "Good morning, welcome to Bekker 2.0 application \n";
cout << "for authorization press 'a' \n"
<< "for exit press 'q': \n";
cin >> ch;
switch (ch)
    {
    case 'a':
{
    system("cls");
    Auth();
    break;
}
    case 'q':
{
    pExit();
    cout << "press any key to exit";
    return;
    break;
}
    default: cout << "Unknown function! Press any button to repeat! " <<
endl;
    system("pause");
    goto start;
    }

}

Data::Data()
```

```
{}
Data::~Data()
{}

void Data::DataEntry(productOrder productorder_, Info info_)
{
    productorder.productType = productorder_.productType;
    productorder.productName   = productorder_.productName;

    info.numbers = info_.numbers;
    info.price = info_.price;
    info.fullPrice = info_.fullPrice;
}

void Data::Print()
{
    cout << "Product Type: "
         << productorder.productType << endl
         << "Product Name: "
         << productorder.productName << endl
         << "Amount: "
         << info.numbers << endl
         << "Price: "
         << info.price << endl
         << "FullPrice: "
         << info.fullPrice << endl
         <<"_____________________________________________" << endl;
}
```

- Если вы являетесь Кассиром, вам надо авторизоваться под ним, предоставляя его логин и пароль, после чего можете нажав клавишу «1» сформировать заказ покупателя, переходя по меню типа товара и наименования, нажав «2» вы сможете удалить весь заказ, в случае если покупатель отказался от покупки, «3» позволяет просмотреть текущий заказ, «4» позволяет провести проверку оплаты, если «да(оплата прошла успешно», он отправляется в файл, который может просмотреть Босс, если «нет», то заказ обнуляется. Если вы являетесь Боссом, вам также надо предоставить свой логин и пароль, после чего выбрать в меню интересующий вас пункт, аналогично с Админом.

Страница GitHub:

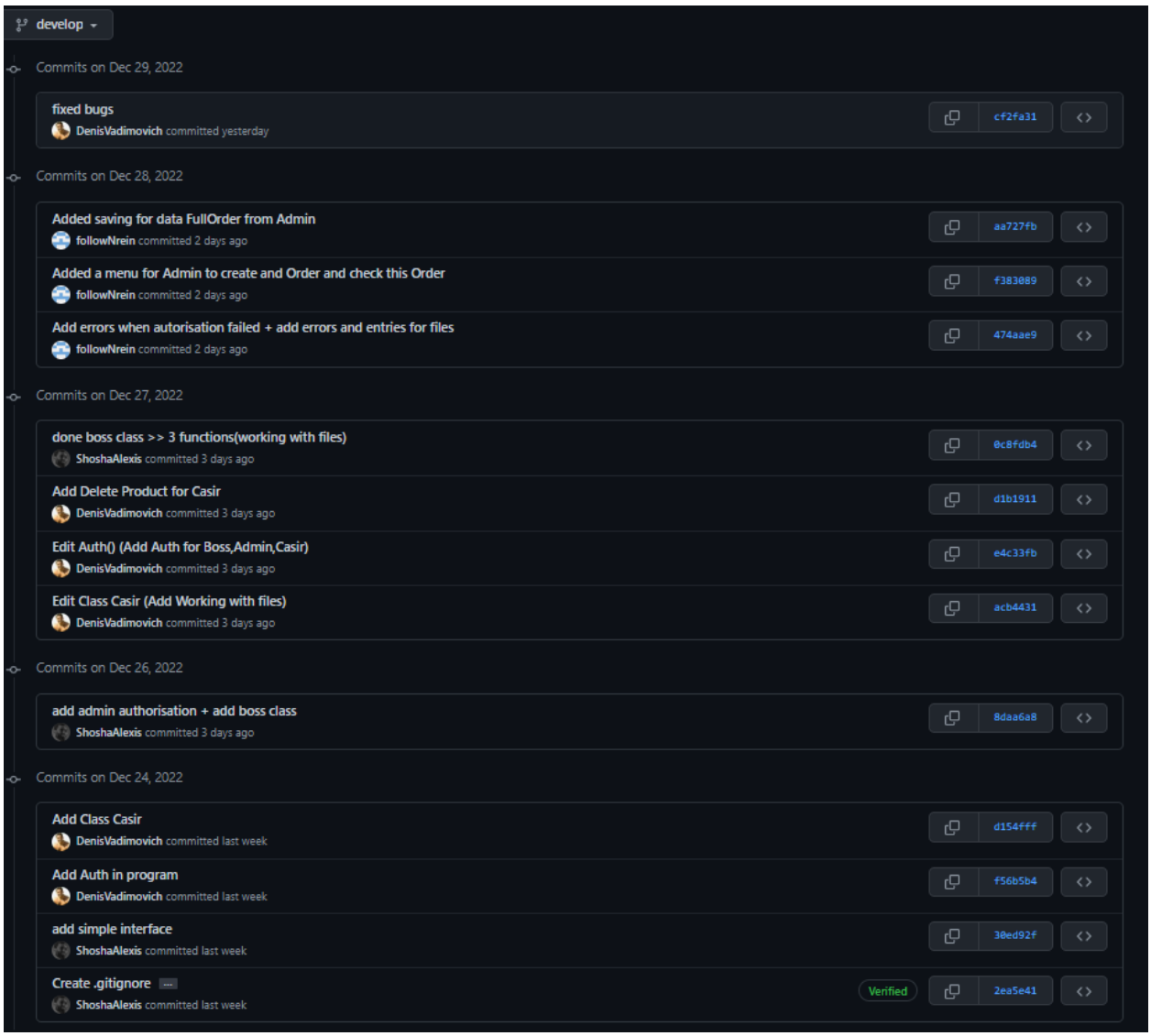

Рис.4 «Коммиты на GitHub»

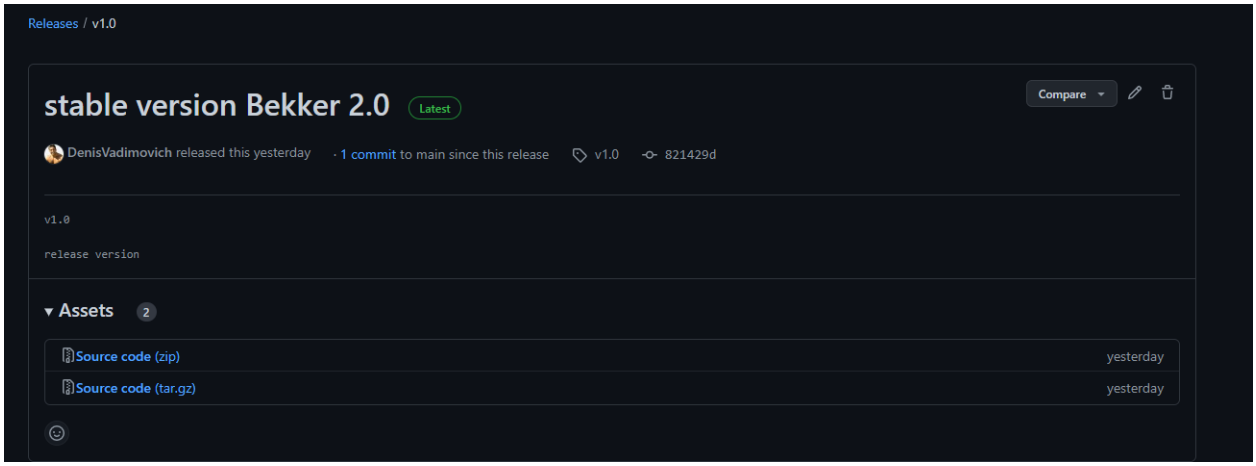

Рис.5 «Релиз стабильной (конечной) версии»

Gitk:

Рис.6 «История коммитов из gitk»

Заключение: В данной работе мы использовали QT Creator 5.4.2, написав код на C++, используя GIT, для обмена наработок между всеми участниками команды, все что собирались реализовать(прецеденты), были реализованы, на момент выхода версии интерфейса 1.0, сбоев и зависаний не наблюдается, были использованы принципы раздельной компиляции, каждая инициализация класса занесена в отдельный файл class.h. Очистка памяти – реализована не была, но планируется в дальнейшем, при выходе версии интерфейса 2.0, отсутствуют неиспользованные переменные и циклы/массивы/конструкций.

Приложив к программе диаграммы использования, активности, классов и последовательности, считаем что была достигнута цель и заказчик остался доволен проделанной работой.

Листинг программы

Начало файла class.h

```
#ifndef CLASS
#define CLASS

#include<iostream>
#include<string>
#include<fstream>
#include<windows.h>
#include <sstream>
#include <stdio.h>

using namespace std;
```

```
class Casir{
public:
    Casir();
    ~Casir();
    int Choice;
    void CasirMenu();
};

class Boss{
public:
    Boss();
    ~Boss();
    char kbtn;
    void BossMenu();
private:
    int Choice;
    int Choice2;
};

class Admin{
public:
    Admin();
    ~Admin();
    int Choice;
    void AdminMenu();
};


class UserInterface
{
private:
char ch;
public:
UserInterface();
~UserInterface();
void interact();
};
struct productOrder
{
    string productType, productName;
};

struct Info
{
    int numbers, price, fullPrice;
};

class Data
{
 private:
    productOrder productorder;
    Info info;

 public:
    Data();
    Data(productOrder productorder_, Info info_);
    ~Data();

    void Print();
    void DataEntry(productOrder productorder_, Info info_);

    productOrder GetProductOrder() {return productorder;}
    Info GetInfo() {return info;}
};
```

```
#endif // CLASS
```

Конец файла class.h

Начало файла function.h

```
#ifndef FUNC
#define FUNC
#include <class.h>

using namespace std;

int Auth();
void Check();
void DataEntry(Data* (&d), int& n);
void Print(Data* d, int n);
void SaveData(Data* d, int n);
double Check (std::string s);
int pExit();

#endif // FUNC
```

Конец файла function.h

Начало файла class.cpp

```
#include "class.h"
#include "function.h"

//using namespace std;

Casir::Casir(){

}
Casir::~Casir(){

}
Boss::Boss(){

}
Boss::~Boss(){

}
Admin::Admin(){

}
Admin::~Admin(){
```

```
}

void Casir::CasirMenu(){
    string line;
     int a;
     char oplata;
     tochka:
    system("cls");
    cout << "Sales Menu" << endl;
    cout << "1. Add a product" << endl;
    cout << "2. Delete product" << endl;
    cout << "3. Final receipt" << endl;
    cout << "4. Payment" << endl;
    cout << "5. Logout" << endl;
    cout << "Choice: ";
    cin >> Choice;
    if(Choice > 5){
        cout << "Please choose from 1 to 5" << endl;
        system("pause");
        goto tochka;
    }
    switch (Choice){
    case 1:
        tochka2:
        system("cls");
        cout << "Choose the type of product" << endl;
        cout << "1. Cake" << endl;
        cout << "2. Bun" << endl;
        cout << "3. Drink" << endl;
        cout << "4. Exit" << endl;
        cout << "Choice: ";
        cin >> Choice;
        if(Choice > 4){
            cout << "Please choose from 1 to 4" << endl;
            system("pause");
            goto tochka2;
        }
        switch(Choice){
            case 1:
            system("cls");
        {
            ifstream in("Cake.txt"); // окрываем файл для чтения
            if (in.is_open())
            {
            while (getline(in, line))
            {
            cout << line << endl;
            }
            }
            else {
            cout << "Error! Press any button to get back to menu. " << endl;
            system ("pause");
            goto tochka2;
            }
            in.close(); // закрываем файл
            cout << "Choice: ";
            cin >> (a);
            if (a == 1){
    ofstream rec;
    rec.open("Order.txt", ios::app);
    if (rec.is_open())
    {
       rec << "Snickers_Cake" << " 212 Rub"<< endl;
    }
```

```
    else
        cout << "Error! Press any button to get back to menu. " << endl;
        system ("pause");
        goto tochka2;
            }

            if (a == 2){
    ofstream rec;
    rec.open("Order.txt", ios::app);
    if (rec.is_open())
    {
       rec << "Napoleon_Cake" << " 240 Rub"<< endl;
    }
    else
        cout << "Error! Press any button to get back to menu. " << endl;
        system ("pause");
        goto tochka2;
            }

            if (a == 3){
    ofstream rec;
    rec.open("Order.txt", ios::app);
    if (rec.is_open())
    {
       rec << "Cheese_Cake" << " 570 Rub"<< endl;
    }
    else
        cout << "Error! Press any button to get back to menu. " << endl;
        system ("pause");
        goto tochka2;
            }

            if (a == 4){
    ofstream rec;
    rec.open("Order.txt", ios::app);
    if (rec.is_open())
    {
       rec << "Sour_Cream_Cake" << " 164 Rub"<< endl;
    }
    else
        cout << "Error! Press any button to get back to menu. " << endl;
        system ("pause");
        goto tochka2;
            }
            goto tochka2;
            break;
        }

            case 2:
            system("cls");
        {
            ifstream in("Bun.txt"); // окрываем файл для чтения
            if (in.is_open())
            {
            while (getline(in, line))
            {
            cout << line << endl;
            }
            }
            else {
                cout << "Error! Press any button to get back to menu. " <<
endl;
                system ("pause");
                goto tochka2;
```

```
        }
        in.close(); // закрываем файл
        cout << "Choice: ";
        cin >> (a);
        if (a == 1){
ofstream rec;
rec.open("Order.txt", ios::app);
if (rec.is_open())
{
   rec << "Sausage_In_The_Dough" << " 40 Rub"<< endl;
}
else
    cout << "Error! Press any button to get back to menu. " << endl;
    system ("pause");
    goto tochka2;
        }

        if (a == 2){
ofstream rec;
rec.open("Order.txt", ios::app);
if (rec.is_open())
{
   rec << "Pizza_Application" << " 25 Rub"<< endl;
}
else
    cout << "Error! Press any button to get back to menu. " << endl;
    system ("pause");
    goto tochka2;
        }

        if (a == 3){
ofstream rec;
rec.open("Order.txt", ios::app);
if (rec.is_open())
{
   rec << "Cabbage_Pie" << " 15 Rub"<< endl;
}
else
    cout << "Error! Press any button to get back to menu. " << endl;
    system ("pause");
    goto tochka2;
        }

        if (a == 4){
ofstream rec;
rec.open("Order.txt", ios::app);
if (rec.is_open())
{
   rec << "Pie_With_Filling_Rice_Egg " << " 10 Rub"<< endl;
}
else
    cout << "Error! Press any button to get back to menu. " << endl;
    system ("pause");
    goto tochka2;
        }
        goto tochka2;
        break;
    }

        case 3:
        system("cls");
    {
        ifstream in("Drink.txt"); // окрываем файл для чтения
        if (in.is_open())
```

```
            {
            while (getline(in, line))
            {
            cout << line << endl;
            }
            }
            else {
                cout << "Error! Press any button to get back to menu. " <<
endl;
                system ("pause");
                goto tochka2;
            }
            in.close(); // закрываем файл
            cout << "Choice: ";
            cin >> (a);
            if (a == 1){
    ofstream rec;
    rec.open("Order.txt", ios::app);
    if (rec.is_open())
    {
       rec << "Tea" << " 15 Rub"<< endl;
    }
    else
        cout << "Error! Press any button to get back to menu. " << endl;
        system ("pause");
        goto tochka2;
            }
            if (a == 2){
    ofstream rec;
    rec.open("Order.txt", ios::app);
    if (rec.is_open())
    {
       rec << "Coffee" << " 35 Rub"<< endl;
    }
    else
        cout << "Error! Press any button to get back to menu. " << endl;
        system ("pause");
        goto tochka2;
            }
            if (a == 3){
    ofstream rec;
    rec.open("Order.txt", ios::app);
    if (rec.is_open())
    {
       rec << "Coca-Cola" << " 80 Rub"<< endl;
    }
    else
        cout << "Error! Press any button to get back to menu. " << endl;
        system ("pause");
        goto tochka2;
            }
            if (a == 4){
    ofstream rec;
    rec.open("Order.txt", ios::app);
    if (rec.is_open())
    {
       rec << "Fanta" << " 75 Rub"<< endl;
    }
    else
        cout << "Error! Press any button to get back to menu. " << endl;
        system ("pause");
        goto tochka2;
            }
            if (a == 5){
```

```
ofstream rec;
rec.open("Order.txt", ios::app);
if (rec.is_open())
{
   rec << "Sprite" << " 70 Rub"<< endl;
}
else
    cout << "Error! Press any button to get back to menu. " << endl;
    system ("pause");
    goto tochka2;
        }
        goto tochka2;
        break;
    }
    case 4:
        goto tochka;
    }
    system("pause");
    system("cls");
case 2:
{
    del:
    system("cls");
    cout << "Delete product:"<< endl;
    ifstream in("Order.txt"); // окрываем файл для чтения
    if (in.is_open())
    {
    while (getline(in, line))
    {
    cout << line << endl;
    }
    }
    cout << "" << endl;
    cout << "Choose main item:" << endl;
    cout << "1. Delete all order" << endl;
    cout << "2. Exit" << endl;

    cout << "Choice: ";
    cin >> Choice;
    if(Choice > 2){
        cout << "Please choose from 1 to 2" << endl;
        system("pause");
        goto del;
    }
    switch(Choice){
          case 1:{
        ofstream outfile;
        outfile.open("Order.txt", ofstream::out | ofstream::trunc);
        outfile.close();

    }

          case 2:{
            goto tochka;
    }
    }

    break;
}
case 3:
    system ("cls");
{
    ifstream out("Order.txt");
    if (out.is_open()){
```

```
    while (getline(out,line)){
    cout << line << endl;
    }
    }
    out.close();
    string s1,tail;
    int s2 = 0;
    int sum = 0;
    ifstream out1("Order.txt");
    if (out1.is_open()){
    while(out1>>s1>>s2)// в s1 - первое слово, в s2 - второе
    {
        s2 = s2;
        getline(out1,tail); // дочитываем остаток строки
        sum += s2;

    }
    cout <<"Result: " <<sum << endl;
    }
    else
        cout << "Error! Press any button to get back to menu. " << endl;
        system ("pause");
        goto tochka;
    out1.close();
    system("pause");
    goto tochka;
    break;
}
case 4:
{
    paid:
    system("cls");
    cout << "The buyer paid the receipt? (y/n)" << endl;
    cin >> oplata;
    if (oplata == 'y' || oplata == 'n'){
    if (oplata == 'y'){

        ifstream rf("Order.txt");
         if (!rf)
         {
           cout << "Cannot open source file." << endl;
         }
         ofstream wf("Storage Orders.txt",ios_base::app);
         if (!wf)
         {
           rf.close();
           cout << "Cannot open the files." << endl;
         }
         char sym;
          while (rf)// цикл посимвольного чтения
            {
           rf.get(sym); // считать из rf => sym
           if (rf)
             wf << sym; // записать sym => wf
            }
         rf.close();
         rf.open("Order.txt", ofstream::out | ofstream::trunc);
         rf.close();
         wf.close();
         goto tochka;
       }
    if (oplata == 'n'){
        ofstream outfile;
        outfile.open("Order.txt");
```

```
            outfile.close();
            goto tochka;
        }
        }else{
            cout << "Wrong parametr, repeat choice" << endl;
            system("pause");
            goto paid;
        }
        break;
}
    case 5:
    {
        system("cls");
        UserInterface theUserInterface;
        theUserInterface.interact();
    }
}
}

void Boss::BossMenu(){
    string line;
    string s1 = "",
           tail = "";
    int s2 = 0;
    double sum = 0;
    int tax = 0;
    int sum2 = 0;
    mainmenu:
    system("cls");
    cout << "Sales Menu" << endl;
    cout << "1. Today buyings" << endl;
    cout << "2. Storage orders" << endl;
    cout << "3. Income and taxes" << endl;
    cout << "4. Logout" << endl;
    cout << "Choice: ";
    cin >> Choice;
    if(Choice > 4){
        cout << "Please choose from 1 to 4" << endl;
        system("pause");
        goto mainmenu;
    }
    switch (Choice){
    case 1:
    {
        system("cls");
        cout << "Today buyings" << endl;
        ifstream out("Storage Orders.txt");
        if (out.is_open()){
        while (getline(out,line)){
        cout << line << endl;
        }
        }
        out.close();
        int s2 = 0;
        int sum = 0;
        ifstream out1("Storage Orders.txt");
        if (out1.is_open()){
        while(out1>>s1>>s2)// в s1 - первое слово, в s2 - второе
        {
            s2 = s2;
            getline(out1,tail); // дочитываем остаток строки
            sum += s2;

        }
```

```
    cout <<"Result: " <<sum << endl;
    }
    system("pause");
    system("cls");
    goto mainmenu;
    break;
}
case 2:
{
    prev:
    system("cls");
    cout << "Orders menu" << endl;
    cout << "1. Show orders" << endl;
    cout << "2. Exit to main menu" << endl;
    cout << "Choice: ";
    cin >> Choice2;
    if(Choice2 > 2){
        cout << "Please choose from 1 to 2" << endl;
        system("pause");
        goto prev;
    }
    switch(Choice2){
        case 1:
    {
            ifstream in("OrderList.txt"); // окрываем файл для чтения
            if (in.is_open())
            {
                system("cls");
                while (getline(in, line))
                {
                    cout << line << endl;
                }
            }
            else {
                cout << "error no such file";
            }
            system("pause");
            system("cls");
            goto prev;
            in.close();     // закрываем файл
            break;
    }

        case 2:
    {
        system("cls");
        Choice = 0;
        goto mainmenu;
        break;
        }
          default: cout << "Wrong parameter!!";
    }
    break;
}
case 3:
{
    system("cls");
    ifstream out1("Storage Orders.txt");
    if (out1.is_open()){
        sum = 0;
        s2 = 0;
    while(out1>>s1>>s2)// в s1 - первое слово, в s2 - второе
    {
        s2 = s2;
```

```
            getline(out1,tail); // дочитываем остаток строки
            sum += s2;
        }
        tax = (sum/100)*6;
        sum2 = sum - tax;
        }
        cout << "There's Income and Taxes of our company" << endl;
        cout <<"Income without taxes: " <<sum << endl;
        cout <<"Taxes(6%): " <<tax << endl;
        cout <<"Income: " <<sum2 << endl;

        system("pause");
        system("cls");
        goto mainmenu;
        break;
    }
    case 4:
    {
        system("cls");
        UserInterface theUserInterface;
        theUserInterface.interact();
    }
    default:
        cout << "Wrong parameter!";
            break;
    }
}
void Admin::AdminMenu(){

    int _size = 0;
    Data* d = new Data[_size];

    int choice;
    admmenu:
    system("cls");
    cout << "Administrator menu" << endl;
    cout << "1. Create a list to order" << endl;
    cout << "2. Check data" << endl;
    cout << "3. Save data" << endl;
    cout << "4. Logout" << endl;
    cout << "Choice: " << endl;
    cin >> choice;
    if(choice > 4){
        cout << "Please choose from 1 to 4" << endl;
        system("pause");
        goto admmenu;
    }
    switch(choice){
    case 1:
    {   system ("cls");
        DataEntry(d, _size);
        system("pause");
        system("cls");
        goto admmenu;
        break;
    }
    case 2:
    {
        system("cls");
        if (_size !=0)
            Print(d, _size);
        else
            cout << "Data is empty!" << endl;
        system("pause");
```

```
        system("cls");
        goto admmenu;
        break;
     }
   case 3:
   {
        system("cls");
        if (_size !=0 )
        SaveData(d, _size);
        else
            cout << "Data is empty!" << endl;
        system("pause");
        system("cls");
        goto admmenu;
        break;
   }
   case 4:
   {
        system("cls");
        UserInterface theUserInterface;
        theUserInterface.interact();
   }
   } // switch
}

UserInterface::UserInterface()
{

}
//-------------------------------------------------------------
UserInterface::~UserInterface()
{

}
//-------------------------------------------------------------

void UserInterface::interact()
{
start:
system("cls");
cout << "Good morning, welcome to Bekker 2.0 application \n";
cout << "for authorization press 'a' \n"
<< "for exit press 'q': \n";
cin >> ch;
switch (ch)
    {
    case 'a':
{
    system("cls");
    Auth();
    break;
}
    case 'q':
{
    pExit();
    cout << "press any key to exit";
    return;
    break;
}
    default: cout << "Unknown function! Press any button to repeat! " <<
endl;
    system("pause");
    goto start;
    }
```

```
}

Data::Data()
{}
Data::~Data()
{}

void Data::DataEntry(productOrder productorder_, Info info_)
{
    productorder.productType = productorder_.productType;
    productorder.productName   = productorder_.productName;

    info.numbers = info_.numbers;
    info.price = info_.price;
    info.fullPrice = info_.fullPrice;
}

void Data::Print()
{
    cout << "Product Type: "
         << productorder.productType << endl
         << "Product Name: "
         << productorder.productName << endl
         << "Amount: "
         << info.numbers << endl
         << "Price: "
         << info.price << endl
         << "FullPrice: "
         << info.fullPrice << endl
         <<"_____________________________________________" << endl;
}
```

Конец файла class.cpp

Начало файла function.cpp

```
#include "function.h"
#include "class.h"


int Auth(){
    string Login;
    string Password;
    int chs;
    std::stringstream* L = new stringstream[100];

    system("cls");
    cout << "Login: " << endl;
    cin >> Login;
    cout << "Password: " << endl;
    cin >> Password;
    if (Login == "Casir" && Password == "cash")
    {
        system("cls");
        cout << "Profile: Casir, select menu item..." << endl;
                Casir CasirM;
```

```
                Casir M.CasirMenu();
    }

    if (Login == "Boss" && Password == "wallet")
    {
        system("cls");
        cout << "Profile: Boss, select menu item..." << endl;
                Boss BossM;
                BossM.BossMenu();
    }
    if (Login == "Admin" && Password == "admin")
    {
        system("cls");
        cout << "Profile: Admin, select menu item..." << endl;
                Admin AdminM;
                AdminM.AdminMenu();
    }
    if (L)
    {
        system("cls");
        cout << "Wrong login or password!!!" << endl
             << "Preess any button to get back to autorisation menu" << endl;
        system("pause");
        delete L;
        Auth();
    }
      return 0;
}
void DataEntry(Data* (&d), int& n)
{

    int Choice;
    productOrder productorder;
    Info info;
    string s;

    cout << "How many positions? ";
    cin >> n;

    d = new Data[n];

    for (int i = 0; i < n; i++)
    {
        cout << "Choose the type of product" << endl;
        cout << "1. Cake" << endl;
        cout << "2. Bun" << endl;
        cout << "3. Drink" << endl;
        cout << "Choice: ";
        cin >> Choice;
        switch(Choice){
        case 1:
        {
            productorder.productType = "Cake";
            system("cls");
            cout << "Choose the type of product" << endl;
            cout << "1. Snickers cake" << endl;
            cout << "2. Napoleon cake" << endl;
            cout << "3. Cheese cake" << endl;
            cout << "4. Sour cream cake " << endl;
            cout << "Choice: ";
            cin >> Choice;
            switch(Choice){
            case 1:
            {   system ("cls");
```

```
        productorder.productName = "Snickers cake";
        cout << "U selected: " << productorder.productName<< endl
             << "How many Snickers cakes u want to order?: ";
        info.numbers = Check(s);
        system ("cls");
        cout << "What is the price of one Snickers cakes? (RUB): ";
        info.price = Check(s);
        info.fullPrice = info.numbers * info.price;
        d[i].DataEntry(productorder, info);
        system ("cls");
        cout << "Data saved!" << endl;
        goto checkpoint;


    }
    case 2:
    {   system ("cls");
        productorder.productName = "Napoleon cake";
        cout    << "U selected: " << productorder.productName<< endl
                << "How many Napoleon cakes u want to order?: ";
        info.numbers = Check(s);
        system ("cls");
        cout << "What is the price of one Napoleon cakes? (RUB): ";
        info.price = Check(s);
        info.fullPrice = info.numbers * info.price;
        d[i].DataEntry(productorder, info);
        system ("cls");
        cout << "Data saved!" << endl;
        goto checkpoint;

    }
    case 3:
    {   system ("cls");
        productorder.productName = "Cheese cake";
        cout    << "U selected: " << productorder.productName<< endl
                << "How many Cheese cakes u want to order?: ";
        info.numbers = Check(s);
        system ("cls");
        cout << "What is the price of one Cheese cakees? (RUB): ";
        info.price = Check(s);
        info.fullPrice = info.numbers * info.price;
        d[i].DataEntry(productorder, info);
        system ("cls");
        cout << "Data saved!" << endl;
        goto checkpoint;

    }
    case 4:
    {   system ("cls");
        productorder.productName = "Sour cream cake";
        cout    << "U selected: " << productorder.productName<< endl
                << "How many Sour cream cakes u want to order?: ";
        info.numbers = Check(s);
        system ("cls");
        cout << "What is the price of one Sour cream cakes? (RUB): ";
        info.price = Check(s);
        info.fullPrice = info.numbers * info.price;
        d[i].DataEntry(productorder, info);
        system ("cls");
        cout << "Data saved!" << endl;
        goto checkpoint;

    }
} // switch case 1.X
```

```
        } // case 1
        case 2:
        {
            productorder.productType = "Bun";
            system("cls");
            cout << "Choose the type of product" << endl;
            cout << "1. Sausage in the dough " << endl;
            cout << "2. Pizza application" << endl;
            cout << "3. Cabbage pie" << endl;
            cout << "4. Pie with filling rice egg " << endl;
            cout << "Choice: ";
            cin >> Choice;
            switch(Choice){
            case 1:
            {   system ("cls");
                productorder.productName = "Sausage in the dough";
                cout    << "U selected: " << productorder.productName<< endl
                        << "How many Sausages in the dough u want to order?:
";
                info.numbers = Check(s);
                system ("cls");
                cout << "What is the price of one Sausage in the dough?
(RUB): ";
                info.price = Check(s);
                info.fullPrice = info.numbers * info.price;
                d[i].DataEntry(productorder, info);
                system ("cls");
                cout << "Data saved!" << endl;
                goto checkpoint;

            }
            case 2:
            {   system ("cls");
                productorder.productName = "Pizza application";
                cout    << "U selected: " << productorder.productName<< endl
                        << "How many Pizza applications u want to order?: ";
                info.numbers = Check(s);
                system ("cls");
                cout << "What is the price of one Pizza application? (RUB):
";
                info.price = Check(s);
                info.fullPrice = info.numbers * info.price;
                d[i].DataEntry(productorder, info);
                system ("cls");
                cout << "Data saved!" << endl;
                goto checkpoint;

            }
            case 3:
            {   system ("cls");
                productorder.productName = "Cabbage pie";
                cout    << "U selected: " << productorder.productName<< endl
                        << "How many Cabbage pie u want to order?: ";
                info.numbers = Check(s);
                system ("cls");
                cout << "What is the price of one Cabbage pies? (RUB): ";
                info.price = Check(s);
                info.fullPrice = info.numbers * info.price;
                d[i].DataEntry(productorder, info);
                system ("cls");
                cout << "Data saved!" << endl;
                goto checkpoint;

            }
```

```
            case 4:
            {   system ("cls");
                productorder.productName = "Pie with filling rice egg";
                cout    << "U selected: " << productorder.productName<< endl
                        << "How many Pies with filling rice egg u want to
order?: ";
                info.numbers = Check(s);
                system ("cls");
                cout << "What is the price of one Pie with filling rice egg?
(RUB): ";
                info.price = Check(s);
                info.fullPrice = info.numbers * info.price;
                d[i].DataEntry(productorder, info);
                system ("cls");
                cout << "Data saved!" << endl;
                goto checkpoint;

            }
           } // switch case 2.X
        } // case 2
        case 3:
        {
            productorder.productType = "Bun";
            system("cls");
            cout << "Choose the type of product" << endl;
            cout << "1. Tea " << endl;
            cout << "2. Coffee" << endl;
            cout << "3. Coca-cola" << endl;
            cout << "4. Fanta " << endl;
            cout << "5. Sprite " << endl;
            cout << "Choice: ";
            cin >> Choice;
            switch (Choice){
            case 1:
            {   system ("cls");
                productorder.productName = "Tea";
                cout    << "U selected: " << productorder.productName<< endl
                        << "How many Tea packs u want to order?: ";
                info.numbers = Check(s);
                system ("cls");
                cout << "What is the price of one Tea pack? (RUB): ";
                info.price = Check(s);
                info.fullPrice = info.numbers * info.price;
                d[i].DataEntry(productorder, info);
                system ("cls");
                cout << "Data saved!" << endl;
                goto checkpoint;
            }
            case 2:
            {   system ("cls");
                productorder.productName = "Coffee";
                cout    << "U selected: " << productorder.productName<< endl
                        << "How many Coffee packs u want to order?: ";
                info.numbers = Check(s);
                system ("cls");
                cout << "What is the price of one Coffee pack? (RUB): ";
                info.price = Check(s);
                info.fullPrice = info.numbers * info.price;
                d[i].DataEntry(productorder, info);
                system ("cls");
                cout << "Data saved!" << endl;
                goto checkpoint;
            }
            case 3:
```

```
            {   system ("cls");
                productorder.productName = "Coca-cola";
                cout    << "U selected: " << productorder.productName<< endl
                        << "How many Coca-cola bottles u want to order?: ";
                info.numbers = Check(s);
                system ("cls");
                cout << "What is the price of one Coca-cola bottle? (RUB): ";
                info.price = Check(s);
                info.fullPrice = info.numbers * info.price;
                d[i].DataEntry(productorder, info);
                system ("cls");
                cout << "Data saved!" << endl;
                goto checkpoint;
            }
            case 4:
            {   system ("cls");
                productorder.productName = "Fanta";
                cout    << "U selected: " << productorder.productName<< endl
                        << "How many Fanta bottles u want to order?: ";
                info.numbers = Check(s);
                system ("cls");
                cout << "What is the price of one Fanta bottle? (RUB): ";
                info.price = Check(s);
                info.fullPrice = info.numbers * info.price;
                d[i].DataEntry(productorder, info);
                system ("cls");
                cout << "Data saved!" << endl;
                goto checkpoint;
            }
            case 5:
            {   system ("cls");
                productorder.productName = "Sprite";
                cout    << "U selected: " << productorder.productName<< endl
                        << "How many Sprite bottles u want to order?: ";
                info.numbers = Check(s);
                system ("cls");
                cout << "What is the price of one Sprite bottle? (RUB): ";
                info.price = Check(s);
                info.fullPrice = info.numbers * info.price;
                d[i].DataEntry(productorder, info);
                system ("cls");
                cout << "Data saved!" << endl;
                goto checkpoint;
            }
           } // switch 3.X
        } // case 3
     } // switch X
    checkpoint:
    cout << endl;
    }
}
void Print(Data* d, int n)
{
    for (int i = 0; i < n; i++)
    {
        cout << endl << "Order number " << i+1  << endl;

        d[i].Print();
    }
}
void SaveData(Data* d, int n)
{
    ofstream record("OrderList.txt", ios_base::app);
    if (record)
```

```
    {
        record  << n << endl;

        for (int i = 0; i < n; i++)
        {


                            record << "Product Type: "
                            << d[i].GetProductOrder().productType << endl
                            << "Product Name: "
                            << d[i].GetProductOrder().productName << endl
                            << "Amount: "
                            << d[i].GetInfo().numbers << endl
                            << "Price: "
                            << d[i].GetInfo().price << endl
                            << "FullPrice: "
                            << d[i].GetInfo().fullPrice << endl
                            <<"_____________________________________________"
<< endl;
            if (i < n-1)
                    record << endl;

        }
    }
    else
    cout << "Error to open a file!" << endl;
    record.close();
}
double Check(std::string s){
    double x;// переменная для ввода числовых данных
    start:
    cin >> s;
    if (s.find_first_not_of("0123456789.") != string::npos)
    {
        system("cls");
        cout << "Error!!! Put the 0123456789 OR ." << endl;
        goto start;
    }
    else
    {
        x = stod(s); // вывод на экран введенного числа
    }
return x;
}
int pExit()
{
    cout << "Press any button to close the programm..." << endl;
    exit(0);
}
```

Конец файла function.cpp

Начало файла bekkerapp.cpp

```
#include "function.h"
#include "class.h"
```

```
int main()
{
    UserInterface theUserInterface;
    theUserInterface.interact();
    pExit();

    return 0;
}
```

Конец файла bekkerapp.cpp