**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”**

Кафедра ЕОМ



**Курсова робота**

**з предмету: «Програмування, частина 2 (Об’єктно- орієнтоване програмування)»**

**на тему:**

**«Базові принципи об’єктно-орієнтованого програмування»**

Індивідуальне завдання: «Кафе»

**Виконав:** ст. гр. КІ-15

Папіш О.Р.

**Прийняла:** асистент

Лихотоп Д.В.

**Львів - 2019**

**Анотації**

Одна з основних завдань програміста - це боротьба зі складністю, набагато зручніше при описі можливостей майбутньої системи говорити в термінах існуючих речей, а не просто вирваними з контексту шматками коду.

ООП надає новий рівень абстракції, що дозволяє оперувати поняттями з реального світу.

Для розв’язання багатьох практичних задач з різних галузей, необхідно використовувати базові принципи об’єктно-орієнтованого програмування, оскільки вони можуть адаптувати вирішення поставлених задач до їх швидкої зміни в реальному часі. Одним з шляхів забезпечення таких вимог є організація об’єктно-орієнтованих процесів і відповідних технічних засобів їх реалізації.

Причому, ефективність застосування об’єктно-орієнтованого програмування у

вирішенні поставленого завдання залежить, як від продуктивності

комп’ютерів, так і від розмірів і структури пам’яті, пропускної здатності каналів зв’язку, використаних об’єктно-орієнтованих мов програмування, компіляторів, операційних систем, чисельних методів та інших математичних досліджень. Такий широкий обсяг параметрів вимагає проведення досліджень на різних рівнях: на рівні опису алгоритмів, створення спеціальних мов програмування, компіляторів, багатопроцесорних систем, неоднорідних систем, кластерів.

Для скорочення термінів розробки об’єктно-орієнтованих програм та дослідження їх роботи використовується моделювання.

Метою виконання курсової роботи є засвоєння основних методів та алгоритмів об’єктно-орієнтованих структур, принципів побудови відповідних структур, набуття початкових практичних навиків проектування таких засобів.

**Зміст:**

1. Анотації 2
2. Огляд та обґрунтовування вибору технологій об'єктно-орієнтованого програмування. 4
3. Аналіз та розробка алгоритму згідно індивідуального завдання. 7
4. Програмна реалізація алгоритму згідно індивідуального завдання.

10

1. Відлагодження та тестування реалізованої програми згідно індивідуального завдання. 12
2. Аналіз продуктивності (часу роботи) та структурної складності реалізованої програми згідно індивідуального завдання. 13
3. Висновки 18
4. Література 19
5. Додаток 1 20

Огляд та обґрунтовування вибору технологій об'єктно-орієнтованого програмування.

**Об'єктно-орієнтоване програмування (ООП)** — одна з парадигм програмування, яка розглядає програму як множину «об'єктів», що взаємодіють між собою. Основу ООП складають три основні концепції: інкапсуляція, успадкування та поліморфізм. Одною з переваг ООП є краща модульність програмного забезпечення.

**Успадкування (наслідування).** Клас може мати «підкласи», спеціалізовані, розширені версії надкласу. Можуть навіть утворюватися цілі дерева успадкування. Підкласи успадковують атрибути та поведінку своїх батьківських класів, і можуть вводити свої власні. Успадкування може бути одиничне (один безпосередній батьківський клас) та множинне (кілька батьківських класів).

**Інкапсуляція**. Приховування деталей про роботу класів від об'єктів, що їх використовують або надсилають їм повідомлення. Інкапсуляція досягається шляхом вказування, які класи можуть звертатися до членів об'єкта. Як наслідок, кожен об'єкт представляє кожному іншому класу певний інтерфейс — члени, доступні іншим класам. Інкапсуляція потрібна для того, аби запобігти використанню користувачами інтерфейсу тих частин реалізації, які, швидше за все, будуть змінюватись. Це дозволить полегшити внесення змін, тобто, без потреби змінювати і користувачів інтерфейсу. Часто, члени класу позначаються як публічні (англ. public), захищені (англ. protected) та приватні (англ.private), визначаючи, чи доступні вони всім класам, підкласам, або лише до класу в якому їх визначено.

**Поліморфізм**. Поліморфізм означає залежність поведінки від класу, в якому ця поведінка викликається, тобто, два або більше класів можуть реагувати по-різному на однакові повідомлення. На практиці - це реалізовується шляхом реалізації ряду підпрограм (функцій, процедур, методів тощо) з однаковими іменами, але з різними параметрами. В залежності від того, що передається і вибирається відповідна підпрограма.

Основна відмінність будь-якої об’єктно-орієнтованої мови програмування від інших не об’єктно-орієнтованих мов програмування є можливість створення нових визначених користувачем типів, що називаються класами. Клас – це визначений користувачем тип з даними-елементами (властивостями) та функціями-елементами (методами), що являються членами класу. Він зазвичай описує певну абстракцію якоїсь сутності реального світу з її властивостями і можливими діями. Змінна типу клас називається об’єктом. Об’єкт – це вже не абстракція, а елемент реального світу, який може володіти певними характеристиками, які задаються властивостями в класі, та діяти згідно операцій заданих методами у класі.

Члени класу (властивості і методи), оголошені після ключового слова *publіc* стають відкритими членами. Це означає, що вони доступні:

* усім іншим членам класу;
* дружнім конструкціям (класам, методам, функціям);
* членам похідних класів;
* з-під об’єктів класу після створення об’єктів; після створення об’єкту класу до його членів можна звертатися лише якщо вони є загальнодоступними.

Члени класу, оголошені після ключового слова *prіvate*, стають закритими членами. Вони доступні:

* для інших членів того ж класу;
* друзям класу.

Якщо в класі не оголошено секцій, але визначено методи чи властивості, то вони вважатимуться такими, що оголошені у секції prіvate. Для модифікації властивостей, що оголошені в секції private створюються спеціальні методи, які здійснюють цю модифікацію. Назви таких методів прийнято починати зі слова set. На жаргоні такі методи інколи називають сетерами. Для читання властивостей, що оголошені в секції private створюються спеціальні методи, які безпечно повертають значення цих властивостей. Назви таких методів прийнято починати зі слова get. На жаргоні такі методи інколи називають ґетерами.

Члени класу, оголошені після ключового слова *protected*, стають захищеними членами.

Вони доступні:

* для інших членів того ж класу;
* членам похідних класів;
* друзям класу.

[ Інформацію взято з сайту <https://www.ua5.org/oop/384-osnovn-ponjattja-obktno-orntovanogo.html> ]

Аналіз та розробка алгоритму згідно індивідуального завдання.

Індивідуальне завдання: «Кафе».

Вхідні дані взяті з меню кафе «Diner&Grill», а саме: назва позиції меню, ціна, ID .

* 1. Опис алгоритму та оцінка його складності.



*Рис. 2.1 Діаграма прецедентів*

Діаграма включає одного актора : Касир; і 6 прецедентів (варіантів використання). Касиру доступні прецеденти: створити нове замовлення, додати до замовлення, закрити замовлення, видалити замовлення, побачити інформацію про окремий столик, показати зайняті столики.

2.2 Аналіз особливостей алгоритму.

В алгоритмі наявна одна базова форма, що утворює меню програми з графічним інтерфейсом. Форма відповідає за меню що включає такі можливості:

а) Додати нове замовлення;

б) Додати позиції до замовлення;

в) Закрити замовлення;

г) Видалити замовлення;

д) Показати інформацію про певний столик;

е) Показати зайняті столики;

При виборі ‘Додати нове замовлення’ перед касиром з’являється нова форма для введення даних (позиції та кількість товару). Після успішного введення інформації , рахунок додається до списку всіх рахунків.

При виборі ‘Додати позиції до замовлення’ перед касиром з’являється нова форма для введення даних (позиції, кількість товару, столик) які треба додати до певного рахунка. Після успішного введення , рахунок замінюється на новий.

При виборі ‘Закрити замовлення’ перед касиром з’являється нова форма для вибору рахунка, який треба закрити. Після успішного вибору номера рахунка, з’являється текстове поле з сумою, яку повинен заплатити клієнт.

При виборі ‘Видалити замовлення’ перед касиром з’являється нова форма для вибору рахунка, який треба видалити. Після чого рахунок видаляється.

При виборі ‘Показати інформацію про певний столик’ перед касиром з’являється нова форма для вибору рахунка, який треба показати.

При виборі ‘Показати зайняті столики’ перед касиром з’являється нова форма з списком зайнятих столиків.

При запуску програми зчитуються дані з текстових файлів "MenuDish.txt", "MenuDrink.txt", "MenuAlcoDrink.txt" які потрібні для збереження інформації про меню кафе, створюється клас для зберігання позицій меню, після чого заповнюються вектори, які зберігають повну інформацію про кожену позицію.

2.3 Опис алгоритму у псевдокоді

Програма міститиме 13 класів:

1. TableClass
2. MenuClass
3. CafeClass
4. PositionOrder
5. OrdersClass
6. Position
7. MainForm
8. NewOrderForm
9. AddToOrderForm
10. CloseOrderForm
11. DeleteOrderForm
12. InfoAboutOrderForm
13. InfoOrdersForm

Клас TableClass міститиме всю інформацію про вільні та зайняті столики а також міститиме методи для усіх властивостей класу.

Клас MenuClass міститиме вектори типів меню класу та володітиме методами, які заповнюють поля класу.

Клас OrderClass міститиме всі замовлення, зроблені касиром.

Клас CaféClass мітитиме інформацію про саме кафе. Унаслідуваний від класів TableClass , MenuClass, OrdersClass.

В інших класах реалізовано використання CafeClass.

**Розробка блок-схеми алгоритму.**

****

*Рис 2.4 Схема алгоритму роботи програми.*

Програмна реалізація алгоритму згідно індивідуального завдання.

# **Опис послідосної версіїї програмної реалізії**

# **Опис структур даних програми**

.

3.1 Опис реалізації основних функцій/методів

Функція створення нового замовлення:

void NewOrderForm::**on\_AddOrderButton\_clicked**();

Функція додавання позицій до замовлення:

void AddToOrderForm::**on\_AddOrderButton\_clicked**();

Фунція закриття рахунка:

void CloseOrderForm::**on\_CloseButton\_clicked**();

Функція видалення рахунка:

void DeleteOrderForm::on\_DeleteButton\_clicked();

Функція видачі інформації про певний столик:

void InfoAboutOrderForm:: **on\_CheckButton\_clicked**();

# **3.2 Опис програмної реалізації алгоритму**

# **3.2.1 Діаграма класів і її опис**



*Рис 3.2.1 Діаграма класів*

На діаграмі зображено 11 класів .

Відлагодження та тестування реалізованої програми згідно індивідуального завдання.

4.1 Відлагодження та тестування програми в цілому

Програма тестувалася неодноразово. Виявлено і усунено наступні баги:

додавання до замовлення декілька однакових позицій, можливість використання тільки кнопки «Нове замовлення», коли ще не здійснено замовлення.

Аналіз продуктивності (часу роботи) та структурної складності реалізованої програми згідно індивідуального завдання.

5.1 Аналіз структурної складності виконання програми.

У контексті теоретичного аналізу центральним для інформатики є поняття складності. Неформально складність - це "інформаційна опірність середовища", тобто те, що заважає вирішувати завдання, принципово забороняє автоматизувати ті чи інші речі. Тому в інформатиці під поняттям структурної складності виконання програми зазвичай ховається алгоритмічна складність , тобто складність розроблення алгоритму для автоматизації деякого процесу чи задачі. Формально складність алгоритму це кількість елементарних операцій, що витрачаються алгоритмом для вирішення конкретного завдання. Складність залежить не тільки від розмірності вхідних даних, але і від самих даних. Очевидно, що чим складніше алгоритм в обчислювальному плані, тим більше часу і обчислювальних ресурсів вимагатиме його виконання.

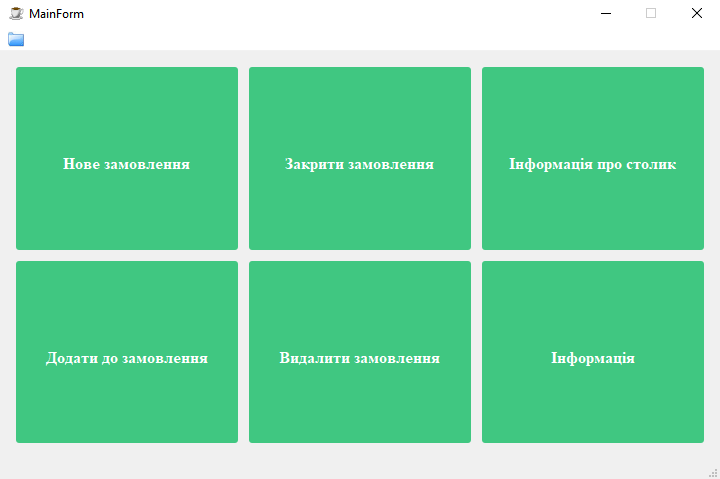
Розрізняють тимчасову і просторову складність. Перша визначає час, необхідний на вирішення завдання заданої розмірності за допомогою даного алгоритму, а друга - кількість необхідних ресурсів (пам'яті) при тих же умовах.

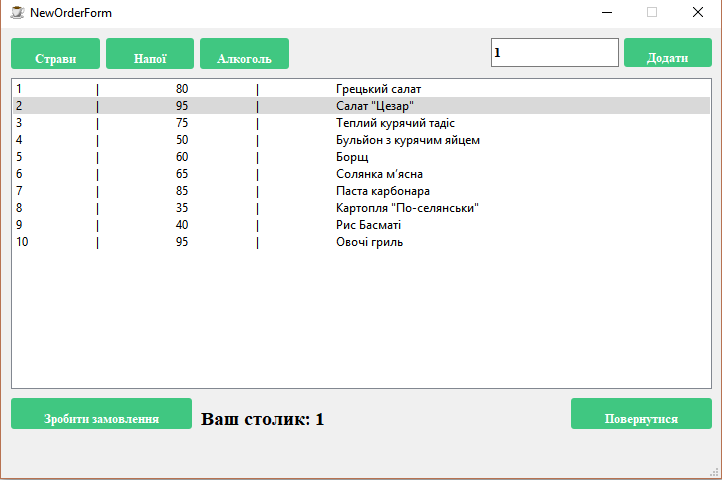
Кожен обчислювальний алгоритм може бути віднесений до одного з двох класів складності. В даному випадку це безліч завдань, для вирішення яких відомі алгоритми, схожі за трудомісткістю. У класі P обчислювальні витрати лінійно зростають зі збільшенням розмірності. Наприклад, час, необхідний на прибирання снігу, прямо пропорційно площі. Якщо її збільшити вдвічі, то і тимчасові витрати також зростуть в два рази. Клас NP включає завдання, для вирішення яких відомі тільки алгоритми, складність яких експоненціально залежить від розмірності даних. Тому вони, як правило, неефективні при роботі з великими множинами. Прикладом є задача пошуку виходу з лабіринту, тимчасові витрати на який експоненціально зростають зі збільшенням числа розгалужень.

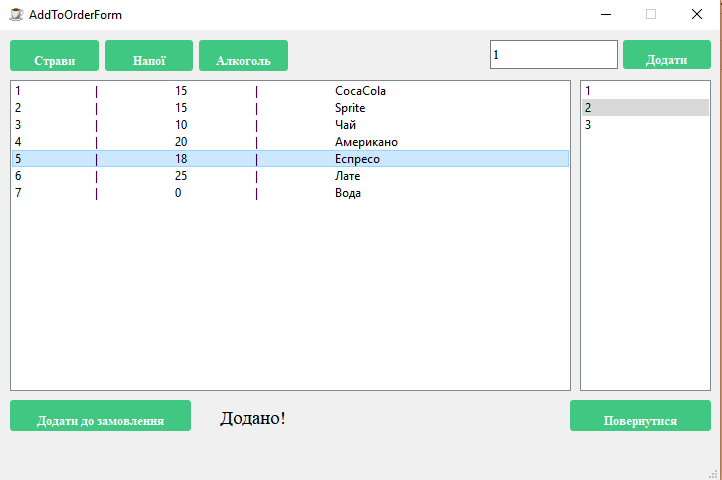
Отже структурну складність виконання програми можна вимірювати за часом виконання алгоритму та за кількістю необхідної пам’яті для збереження даних . Відповідно якщо складність велика то це означає що алгоритм потребує багато часу та великого обсягу пам’яті для вирішення задачі.

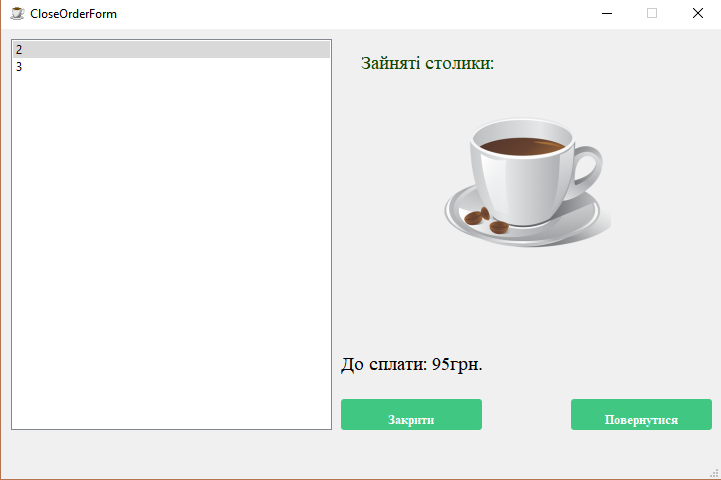
5.2 Аналіз часу виконання програми в залежності від об'єму вхідних даних.

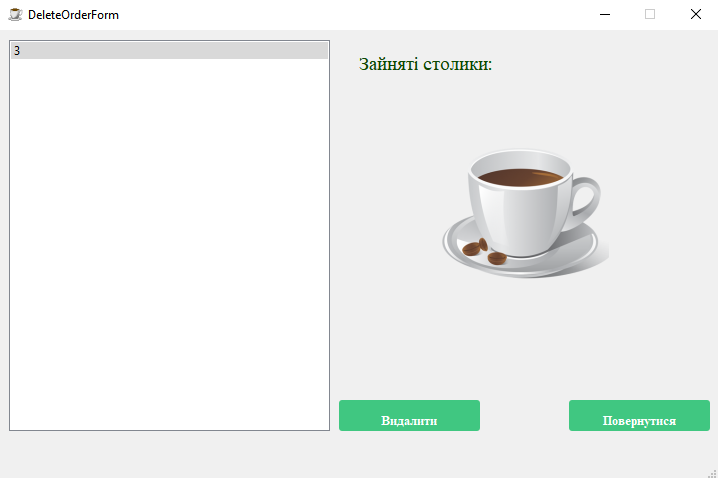
Всі операції всередині програми виконуються миттєво, немає затримки між переходами по меню та та переходами між формами. Читання даних з файлів та виконання операцій з числами виконується бездоганно при коректному використанні програми.

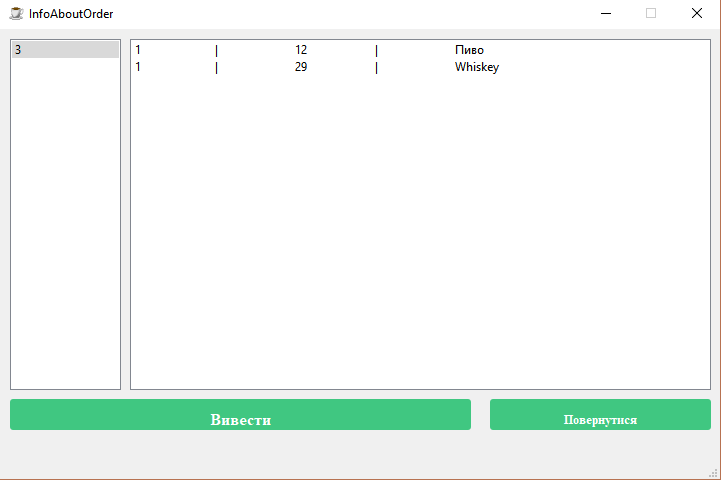


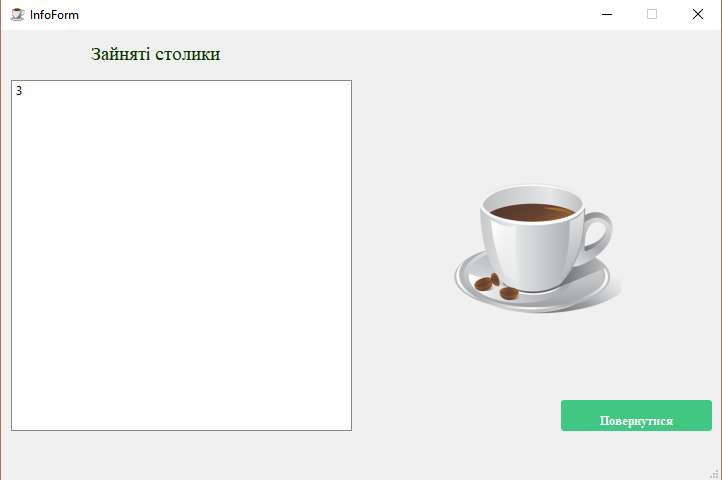












**Висновки**

В цій курсовій роботі я дізнався про основні принципи Об’єктно-орієнтованого програмування, та навчився використовувати ці принципи. Я дізнався про основні методи, алгоритми і засоби опрацювання інформації, засоби програмування об’єктно-орієнтованих структур, склад апаратних засобів та програмного забезпечення обчислювальних систем.

Також навчився виконувати елементарні вправи з обєктно-орієнтованого програмування, описувати алгоритми поставленого перед мною завдання, використовувати класи об’єкти, застосовувати модульне програмування. Засвоїв такі принципи як поліморфізм, спадкування та інкапсуляція.

**Література**

1. Прата С. Язык программирования C++. Лекции и упражнения. 6-е издание / Стивен Прата; пер. с англ. – М.: ООО «И.Д. Вильямс», 2012. – 1248 с.: ил. – Парал. тит. англ.
2. Полный справочник по C++, 4-е издание / Герберт Шилдт — М.: «Вильямс», 2011. - 800 с.
3. Chris Giametta «Pro Flex on Spring», 2009.- p.445.
4. Тамре Л. «Введение в тестирование програмного обеспечения ».- М.: Издательский дом «Вильямс», 2003.- С. 368.
5. Липаев В.В. «Обеспечение качества программных средств. Методы и стандарты». - М.: Синтег, 2001.- С. 246.
6. Макгрегор Дж., Сайкс Д. «Тестирование объектно-ориентированного програмного обеспечения».- К: Диасофт, 2002.- С. 432.
7. Макгрегор Дж., Сайкс Д. Тестирование объектно-ориентированного програмного обеспечения.- К: Диасофт, 2002.- С. 432.
8. Тамре Л. Введение в тестирование програмного обеспечения.- М.: Издательский дом―Вильямс‖, 2003.- С. 368.
9. Э. Гамм, Р. Хелм, Р. Джонсон, Д. Влиссидес «Приѐмы объектно-ориентированного проектирования. Паттерны проектирования», Питер, Москва, 2007.- С. 366.
10. Джон Влиссидес «Применение шаблонов проектирования», Москва, 2003.- С. 130.
11. Джереми Сик, Лай-Кван Ли, Эндрю Ламсдэйн  «C++ Boost Graph Library» Издательство: Питер, 2006, c. 301

Додаток 1

**//addtoorderform.h**

#ifndef ADDTOORDERFORM\_H

#define ADDTOORDERFORM\_H

#include <QMainWindow>

#include "mainform.h"

namespace Ui {

class AddToOrderForm;

}

class AddToOrderForm : public QMainWindow

{

Q\_OBJECT

public:

explicit AddToOrderForm(QWidget \*parent = 0);

~***AddToOrderForm***();

enum {Dish = 1, Drink, AlcoDrink};

int MenuValue;

QVector<PositionOrder> vector;

CafeClass \*Cafe;

void **SetCafe**(CafeClass \*Cafe){

this->Cafe = Cafe;

}

QVector<int> ReserTables;

void **FillResTable**();

void **FillResTableList**();

bool DoOrder = false;

private slots:

void **on\_MenuDishButton\_clicked**();

void **on\_MenuDrinkButton\_clicked**();

void **on\_MenuAlcoDrinkButton\_clicked**();

void **on\_ReturnButton\_clicked**();

void **on\_AddButton\_clicked**();

void **on\_AddOrderButton\_clicked**();

private:

Ui::AddToOrderForm \*ui;

};

#endif // ADDTOORDERFORM\_H

//cafeclass.h

#ifndef CAFECLASS\_H

#define CAFECLASS\_H

#include "tableclass.h"

#include "menuclass.h"

#include "orderclass.h"

class CafeClass: public TableClass, public MenuClass, public OrderClass

{

public:

CafeClass();

};

#endif // CAFECLASS\_H

//closeorderform.h

#ifndef CLOSEORDERFORM\_H

#define CLOSEORDERFORM\_H

#include <QMainWindow>

#include "mainform.h"

namespace Ui {

class CloseOrderForm;

}

class CloseOrderForm : public QMainWindow

{

Q\_OBJECT

public:

explicit CloseOrderForm(QWidget \*parent = 0);

~***CloseOrderForm***();

CafeClass \*Cafe;

void **SetCafe**(CafeClass \*Cafe){

this->Cafe = Cafe;

}

QVector<int> ReserTables;

void **FillResTable**();

void **FillResTableList**();

private slots:

void **on\_ReturnButton\_clicked**();

void **on\_CloseButton\_clicked**();

private:

Ui::CloseOrderForm \*ui;

};

#endif // CLOSEORDERFORM\_H

//deleteorderform.h

#ifndef DELETEORDER\_H

#define DELETEORDER\_H

#include <QMainWindow>

#include "mainform.h"

namespace Ui {

class DeleteOrder;

}

class DeleteOrder : public QMainWindow

{

Q\_OBJECT

public:

explicit DeleteOrder(QWidget \*parent = 0);

~***DeleteOrder***();

CafeClass \*Cafe;

void **SetCafe**(CafeClass \*Cafe){

this->Cafe = Cafe;

}

QVector<int> ReserTables;

void **FillResTable**();

void **FillResTableList**();

private slots:

void **on\_ReturnButton\_2\_clicked**();

void **on\_DeleteButton\_clicked**();

private:

Ui::DeleteOrder \*ui;

};

#endif // DELETEORDER\_H

//infoaboutorderform.h

#ifndef INFOABOUTORDERFORM\_H

#define INFOABOUTORDERFORM\_H

#include <QMainWindow>

#include "mainform.h"

namespace Ui {

class InfoAboutOrderForm;

}

class InfoAboutOrderForm : public QMainWindow

{

Q\_OBJECT

public:

explicit InfoAboutOrderForm(QWidget \*parent = 0);

~***InfoAboutOrderForm***();

CafeClass \*Cafe;

void **SetCafe**(CafeClass \*Cafe){

this->Cafe = Cafe;

}

QVector<int> ReserTables;

void **FillResTable**();

void **FillResTableList**();

private slots:

void **on\_ReturnButton\_clicked**();

void **on\_CheckButton\_clicked**();

private:

Ui::InfoAboutOrderForm \*ui;

};

#endif // INFOABOUTORDERFORM\_H

//infoform.h

#ifndef INFOFORM\_H

#define INFOFORM\_H

#include <QMainWindow>

#include "mainform.h"

namespace Ui {

class InfoForm;

}

class InfoForm : public QMainWindow

{

Q\_OBJECT

public:

explicit InfoForm(QWidget \*parent = 0);

~***InfoForm***();

CafeClass \*Cafe;

void **SetCafe**(CafeClass \*Cafe){

this->Cafe = Cafe;

}

QVector<int> ReserTables;

void **FillResTable**();

void **FillResTableList**();

private slots:

void **on\_ReturnButton\_clicked**();

private:

Ui::InfoForm \*ui;

};

#endif // INFOFORM\_H

//mainform.h

#ifndef MAINFORM\_H

#define MAINFORM\_H

#include <QMainWindow>

#include <QMessageBox>

#include <QFile>

#include <QString>

#include <QApplication>

#include <QDialog>

#include "cafeclass.h"

#include "neworderform.h"

#include "addtoorderform.h"

#include "infoaboutorderform.h"

#include "infoform.h"

#include "deleteorder.h"

#include "closeorderform.h"

using namespace std;

namespace Ui {

class MainForm;

}

class MainForm : public QMainWindow

{

Q\_OBJECT

public:

CafeClass Cafe;

explicit MainForm(QWidget \*parent = 0);

~***MainForm***();

private slots:

void **on\_InfoAboutDev\_triggered**();

void **on\_Money\_triggered**();

void **on\_Exit\_triggered**();

void **on\_NewOrderButton\_clicked**();

void **on\_AddToOrderButton\_clicked**();

void **on\_InfoAboutOrderButton\_clicked**();

void **on\_Info\_clicked**();

void **on\_DeleteOrderButton\_clicked**();

void **on\_CloseOrderButton\_clicked**();

private:

Ui::MainForm \*ui;

};

#endif // MAINFORM\_H

//menuclass.h

#ifndef MENUCLASS\_H

#define MENUCLASS\_H

#include <QVector>

#include <QFile>

#include <QString>

class Position{

public:

QString id, cost, name;

};

class MenuClass

{

private:

QVector<Position> MenuDish;

QVector<Position> MenuDrink;

QVector<Position> MenuAlcoDrink;

public:

MenuClass();

Position **GetMenuDish**(int index);

int **GetMenuDishSize**(){

return MenuDish.size();

}

Position **GetMenuDrink**(int index);

int **GetMenuDrinkSize**(){

return MenuDrink.size();

}

Position **GetMenuAlcoDrink**(int index);

int **GetMenuAlcoDrinkSize**(){

return MenuAlcoDrink.size();

}

};

#endif // MENUCLASS\_H

//neworderform.h

#ifndef NEWORDERFORM\_H

#define NEWORDERFORM\_H

#include <QMainWindow>

#include "mainform.h"

namespace Ui {

class NewOrderForm;

}

class NewOrderForm : public QMainWindow

{

Q\_OBJECT

public:

enum {Dish = 1, Drink, AlcoDrink};

int MenuValue;

bool DoOrder = false;

QVector<PositionOrder> vector;

CafeClass \*Cafe;

explicit NewOrderForm( QWidget \*parent = 0);

void **SetCafe**(CafeClass \*Cafe){

this->Cafe = Cafe;

}

~***NewOrderForm***();

private slots:

void **on\_pushButton\_clicked**();

void **on\_ReturnButton\_clicked**();

void **on\_MenuDishButton\_clicked**();

void **on\_MenuDrinkButton\_clicked**();

void **on\_MenuAlcoDrinkButton\_clicked**();

void **on\_AddButton\_clicked**();

void **on\_AddOrderButton\_clicked**();

private:

Ui::NewOrderForm \*ui;

};

#endif // NEWORDERFORM\_H

//orderclass.h

#ifndef ORDERCLASS\_H

#define ORDERCLASS\_H

#include <QVector>

#include<QString>

class PositionOrder{

public:

QString name, cost, nums;

bool **operator**==(PositionOrder &obj){

if(this->name == obj.name){

return true;

}

else{

return false;

}

}

PositionOrder& **operator**=(const PositionOrder &obj){

if(this == &obj){

return \*this;

}

else{

this->cost = obj.cost;

this->name = obj.name;

this->nums = obj.nums;

}

return \*this;

}

PositionOrder **operator**+(PositionOrder &X){

PositionOrder result;

result.cost = X.cost;

result.name = X.name;

result.nums = QString::number(this->nums.toInt()+X.nums.toInt());

return result;

}

};

class OrderClass

{

private:

QVector<QVector<PositionOrder>> Orders;

public:

OrderClass();

QVector<PositionOrder> **GetOrder**(int index);

void **SetOrder**(int index, QVector<PositionOrder> Order);

void **ClearOrder**(int index);

bool IsOrders = false;

};

#endif // ORDERCLASS\_H

//tableclass.h

#ifndef TABLECLASS\_H

#define TABLECLASS\_H

#include <QVector>

class TableClass

{

private:

bool FreeTables[15];

public:

TableClass();

void **ReservTable**(int index);

int **FreeTable**();

void **UnReservTable**(int index);

bool **GetReservTable**(int index);

};

#endif // TABLECLASS\_H

//addtoorderform.cpp

#include "addtoorderform.h"

#include "ui\_addtoorderform.h"

AddToOrderForm::**AddToOrderForm**(QWidget \*parent) :

QMainWindow(parent),

ui(new Ui::AddToOrderForm)

{

vector.clear();

MenuValue = 0;

ui->setupUi(this);

}

void AddToOrderForm::**FillResTable**(){

for(int i = 0; i<15;i++){

if(!(Cafe->GetReservTable(i))){

ReserTables.push\_back(i);

}

}

}

void AddToOrderForm::**FillResTableList**(){

for(int i = 0; i<ReserTables.size();i++){

ui->TableList->addItem(QString::number(ReserTables[i]+1));

}

ui->TableList->setCurrentRow(0);

}

AddToOrderForm::~***AddToOrderForm***()

{

delete ui;

}

void AddToOrderForm::**on\_MenuDishButton\_clicked**()

{

ui->listWidget->clear();

QString str;

Position pos;

for(int i = 0; i<Cafe->GetMenuDishSize();i++){

pos = Cafe->GetMenuDish(i);

str = pos.id + "\t|\t" + pos.cost + "\t|\t" + pos.name;

str.replace("\n", "");

ui->listWidget->addItem(str);

}

MenuValue = Dish;

ui->listWidget->setCurrentRow(0);

}

void AddToOrderForm::**on\_MenuDrinkButton\_clicked**()

{

ui->listWidget->clear();

QString str;

Position pos;

for(int i = 0; i<Cafe->GetMenuDrinkSize();i++){

pos = Cafe->GetMenuDrink(i);

str = pos.id + "\t|\t" + pos.cost + "\t|\t" + pos.name;

str.replace("\n", "");

ui->listWidget->addItem(str);

}

MenuValue = Drink;

ui->listWidget->setCurrentRow(0);

}

void AddToOrderForm::**on\_MenuAlcoDrinkButton\_clicked**()

{

ui->listWidget->clear();

QString str;

Position pos;

for(int i = 0; i<Cafe->GetMenuAlcoDrinkSize();i++){

pos = Cafe->GetMenuAlcoDrink(i);

str = pos.id + "\t|\t" + pos.cost + "\t|\t" + pos.name;

str.replace("\n", "");

ui->listWidget->addItem(str);

}

MenuValue = AlcoDrink;

ui->listWidget->setCurrentRow(0);

}

void AddToOrderForm::**on\_ReturnButton\_clicked**()

{

this->close();

}

void AddToOrderForm::**on\_AddButton\_clicked**()

{

ui->TableLabel->setText("---");

bool is\_true = true;

int num= -1;

num = (ui->lineEdit->text()).toInt();

if(num<=0 || ui->listWidget->count() == 0){

is\_true = false;

ui->TableLabel->setText("Не вірний формат!");

}

if(is\_true){

int item = -1;

item = ui->listWidget->currentRow();

if(item>=0){

Position pos;

PositionOrder ord;

switch(MenuValue){

case Dish:

pos = Cafe->GetMenuDish(item);

break;

case Drink:

pos = Cafe->GetMenuDrink(item);

break;

case AlcoDrink:

pos = Cafe->GetMenuAlcoDrink(item);

break;

}

ord.cost = pos.cost.replace("\n", "");

ord.name = pos.name.replace("\n", "");

ord.nums = QString::number(num);

vector.push\_back(ord);

DoOrder = true;

}

}

}

void AddToOrderForm::**on\_AddOrderButton\_clicked**()

{

if(DoOrder){

int index =(ui->TableList->currentItem()->text()).toInt()-1;

if(index>=0){

QVector<PositionOrder> temp = Cafe->GetOrder(index);

for(int i = 0; i<vector.size();i++){

temp.push\_back(vector[i]);

}

for(int i = 0; i<temp.size();i++){

PositionOrder pos = temp[i];

for(int j = 0;j<temp.size();j++){

if(i == j){

continue;

}

else if(pos == temp[j]){

pos = pos + temp[j];

temp.remove(i);

temp.insert(i,pos);

temp.remove(j);

j--;

}

}

}

Cafe->SetOrder(index, temp);

vector.clear();

ui->TableLabel->setText("Додано!");

DoOrder = false;

}

}

}

//cafeclass.cpp

#include "cafeclass.h"

CafeClass::**CafeClass**()

{

}

//closeorderform.cpp

#include "closeorderform.h"

#include "ui\_closeorderform.h"

CloseOrderForm::CloseOrderForm(QWidget \*parent) :

QMainWindow(parent),

ui(new Ui::CloseOrderForm)

{

ui->setupUi(this);

}

CloseOrderForm::~CloseOrderForm()

{

delete ui;

}

void CloseOrderForm::on\_ReturnButton\_clicked()

{

this->close();

}

void CloseOrderForm::FillResTable(){

for(int i = 0; i<15;i++){

if(!(Cafe->GetReservTable(i))){

ReserTables.push\_back(i);

}

}

}

void CloseOrderForm::FillResTableList(){

for(int i = 0; i<ReserTables.size();i++){

ui->TableList->addItem(QString::number(ReserTables[i]+1));

}

ui->TableList->setCurrentRow(0);

}

void CloseOrderForm::on\_CloseButton\_clicked()

{

ui->MoneyLabel->setText("До сплати:");

int index = -1;

if(ui->TableList->count()>0){

index= (ui->TableList->currentItem()->text()).toInt() -1;

if(index >= 0){

QVector<PositionOrder> vector;

vector = Cafe->GetOrder(index);

int money = 0;

for(int i = 0; i<vector.size();i++){

money += (vector[i].cost.toInt()) \*(vector[i].nums.toInt());

}

ui->MoneyLabel->setText("До сплати: "+ QString::number(money) +"грн.");

Cafe->ClearOrder(index);

Cafe->UnReservTable(index);

ui->TableList->clear();

ReserTables.clear();

FillResTable();

FillResTableList();

if(ReserTables.size() == 0){

Cafe->IsOrders = false;

}

}

else{

this->close();

}

}

}

//deleteorder.cpp

#include "deleteorder.h"

#include "ui\_deleteorder.h"

DeleteOrder::DeleteOrder(QWidget \*parent) :

QMainWindow(parent),

ui(new Ui::DeleteOrder)

{

ui->setupUi(this);

}

DeleteOrder::~DeleteOrder()

{

delete ui;

}

void DeleteOrder::on\_ReturnButton\_2\_clicked()

{

this->close();

}

void DeleteOrder::FillResTable(){

for(int i = 0; i<15;i++){

if(!(Cafe->GetReservTable(i))){

ReserTables.push\_back(i);

}

}

}

void DeleteOrder::FillResTableList(){

for(int i = 0; i<ReserTables.size();i++){

ui->TableList->addItem(QString::number(ReserTables[i]+1));

}

ui->TableList->setCurrentRow(0);

}

void DeleteOrder::on\_DeleteButton\_clicked()

{

int index = -1;

if(ui->TableList->count()>0){

index= (ui->TableList->currentItem()->text()).toInt() -1;

if(index >= 0){

Cafe->ClearOrder(index);

Cafe->UnReservTable(index);

ui->TableList->clear();

ReserTables.clear();

FillResTable();

FillResTableList();

if(ReserTables.size() == 0){

Cafe->IsOrders = false;

}

}

else{

this->close();

}

}

}

//infoaboutorder.cpp

#include "infoaboutorderform.h"

#include "ui\_infoaboutorderform.h"

InfoAboutOrderForm::InfoAboutOrderForm(QWidget \*parent) :

QMainWindow(parent),

ui(new Ui::InfoAboutOrderForm)

{

ui->setupUi(this);

}

void InfoAboutOrderForm::FillResTable(){

for(int i = 0; i<15;i++){

if(!(Cafe->GetReservTable(i))){

ReserTables.push\_back(i);

}

}

}

void InfoAboutOrderForm::FillResTableList(){

for(int i = 0; i<ReserTables.size();i++){

ui->TableList->addItem(QString::number(ReserTables[i]+1));

}

ui->TableList->setCurrentRow(0);

}

InfoAboutOrderForm::~InfoAboutOrderForm()

{

delete ui;

}

void InfoAboutOrderForm::on\_ReturnButton\_clicked()

{

this->close();

}

void InfoAboutOrderForm::on\_CheckButton\_clicked()

{

int index = -1;

index = (ui->TableList->currentItem()->text()).toInt() -1;

if(index>=0){

ui->OrdersList->clear();

QVector<PositionOrder> pos;

pos = Cafe->GetOrder(index);

for(int i = 0; i<pos.size();i++){

ui->OrdersList->addItem(pos[i].nums + "\t|\t"+pos[i].cost + "\t|\t"+pos[i].name);

}

}

}

//infoform.cpp

#include "infoform.h"

#include "ui\_infoform.h"

InfoForm::InfoForm(QWidget \*parent) :

QMainWindow(parent),

ui(new Ui::InfoForm)

{

ui->setupUi(this);

}

InfoForm::~InfoForm()

{

delete ui;

}

void InfoForm::FillResTable(){

for(int i = 0; i<15;i++){

if(!(Cafe->GetReservTable(i))){

ReserTables.push\_back(i);

}

}

}

void InfoForm::FillResTableList(){

for(int i = 0; i<ReserTables.size();i++){

ui->TableList->addItem(QString::number(ReserTables[i]+1));

}

}

void InfoForm::on\_ReturnButton\_clicked()

{

this->close();

}

//main.cpp

#include "mainform.h"

#include <QApplication>

int main(int argc, char \*argv[])

{

QApplication a(argc, argv);

MainForm w;

w.show();

return a.exec();

}

//mainform.cpp

#include "mainform.h"

#include "ui\_mainform.h"

MainForm::MainForm(QWidget \*parent) :

QMainWindow(parent),

ui(new Ui::MainForm)

{

ui->setupUi(this);

}

MainForm::~MainForm()

{

delete ui;

}

void MainForm::on\_InfoAboutDev\_triggered()

{

QMessageBox::information(this, "Інформація про розробника", "Program was created by Oleksandr Papish(@Alex\_Fix251)");

}

void MainForm::on\_Money\_triggered()

{

}

void MainForm::on\_Exit\_triggered()

{

QApplication::quit();

}

void MainForm::on\_NewOrderButton\_clicked()

{

NewOrderForm \*form = new NewOrderForm;

form->SetCafe(&Cafe);

form->show();

}

void MainForm::on\_AddToOrderButton\_clicked()

{

if(Cafe.IsOrders){

AddToOrderForm \*form = new AddToOrderForm;

form->SetCafe(&Cafe);

form->FillResTable();

form->FillResTableList();

form->show();

}

}

void MainForm::on\_InfoAboutOrderButton\_clicked()

{

if(Cafe.IsOrders){

InfoAboutOrderForm \*form = new InfoAboutOrderForm;

form->SetCafe(&Cafe);

form->FillResTable();

form->FillResTableList();

form->show();

}

}

void MainForm::on\_Info\_clicked()

{

if(Cafe.IsOrders){

InfoForm \*form = new InfoForm;

form->SetCafe(&Cafe);

form->FillResTable();

form->FillResTableList();

form->show();

}

}

void MainForm::on\_DeleteOrderButton\_clicked()

{

if(Cafe.IsOrders){

DeleteOrder \*form = new DeleteOrder;

form->SetCafe(&Cafe);

form->FillResTable();

form->FillResTableList();

form->show();

}

}

void MainForm::on\_CloseOrderButton\_clicked()

{

if(Cafe.IsOrders){

CloseOrderForm \*form = new CloseOrderForm;

form->SetCafe(&Cafe);

form->FillResTable();

form->FillResTableList();

form->show();

}

}

//menuclass.cpp

#include "menuclass.h"

MenuClass::**MenuClass**()

{

Position temp;

QFile file(":/res/MenuDish.txt");

if((file.exists())&&(file.*open*(QIODevice::ReadOnly))){

while(!file.*atEnd*()){

temp.id = file.readLine();

temp.cost = file.readLine();

temp.name = file.readLine();

if(temp.id == ""){

break;

}

MenuDish.push\_back(temp);

}

file.*close*();

}

QFile file1(":/res/MenuDrink.txt");

if((file1.exists())&&(file1.*open*(QIODevice::ReadOnly))){

while(!file1.*atEnd*()){

temp.id = file1.readLine();

temp.cost = file1.readLine();

temp.name = file1.readLine();

if(temp.id == ""){

break;

}

MenuDrink.push\_back(temp);

}

file1.*close*();

}

QFile file2(":/res/MenuAlcoDrink.txt");

if((file2.exists())&&(file2.*open*(QIODevice::ReadOnly))){

while(!file2.*atEnd*()){

temp.id = file2.readLine();

temp.cost = file2.readLine();

temp.name = file2.readLine();

if(temp.id == ""){

break;

}

MenuAlcoDrink.push\_back(temp);

}

file2.*close*();

}

}

Position MenuClass::**GetMenuDish**(int index){

return MenuDish.at(index);

}

Position MenuClass::**GetMenuDrink**(int index){

return MenuDrink.at(index);

}

Position MenuClass::**GetMenuAlcoDrink**(int index){

return MenuAlcoDrink.at(index);

}

//neworderform.cpp

#include "neworderform.h"

#include "ui\_neworderform.h"

NewOrderForm::NewOrderForm(QWidget \*parent) :

QMainWindow(parent),

ui(new Ui::NewOrderForm)

{

vector.clear();

MenuValue = 0;

ui->setupUi(this);

}

NewOrderForm::~NewOrderForm()

{

delete ui;

}

void NewOrderForm::on\_ReturnButton\_clicked()

{

this->close();

}

void NewOrderForm::on\_pushButton\_clicked()

{

}

void NewOrderForm::on\_MenuDishButton\_clicked()

{

ui->listWidget->clear();

QString str;

Position pos;

for(int i = 0; i<Cafe->GetMenuDishSize();i++){

pos = Cafe->GetMenuDish(i);

str = pos.id + "\t|\t" + pos.cost + "\t|\t" + pos.name;

str.replace("\n", "");

ui->listWidget->addItem(str);

}

MenuValue = Dish;

ui->listWidget->setCurrentRow(0);

}

void NewOrderForm::on\_MenuDrinkButton\_clicked()

{

ui->listWidget->clear();

QString str;

Position pos;

for(int i = 0; i<Cafe->GetMenuDrinkSize();i++){

pos = Cafe->GetMenuDrink(i);

str = pos.id + "\t|\t" + pos.cost + "\t|\t" + pos.name;

str.replace("\n", "");

ui->listWidget->addItem(str);

}

MenuValue = Drink;

ui->listWidget->setCurrentRow(0);

}

void NewOrderForm::on\_MenuAlcoDrinkButton\_clicked()

{

ui->listWidget->clear();

QString str;

Position pos;

for(int i = 0; i<Cafe->GetMenuAlcoDrinkSize();i++){

pos = Cafe->GetMenuAlcoDrink(i);

str = pos.id + "\t|\t" + pos.cost + "\t|\t" + pos.name;

str.replace("\n", "");

ui->listWidget->addItem(str);

}

MenuValue = AlcoDrink;

ui->listWidget->setCurrentRow(0);

}

void NewOrderForm::on\_AddButton\_clicked()

{

bool is\_true = true;

int num= -1;

num = (ui->lineEdit->text()).toInt();

if(num<=0 || ui->listWidget->count() == 0){

ui->TableLabel->setText("Невірний формат!");

is\_true = false;

}

if(is\_true){

ui->TableLabel->setText("Ваш столик: ");

int item = ui->listWidget->currentRow();

Position pos;

PositionOrder ord;

switch(MenuValue){

case Dish:

pos = Cafe->GetMenuDish(item);

break;

case Drink:

pos = Cafe->GetMenuDrink(item);

break;

case AlcoDrink:

pos = Cafe->GetMenuAlcoDrink(item);

break;

}

ord.cost = pos.cost.replace("\n", "");

ord.name = pos.name.replace("\n", "");

ord.nums = QString::number(num);

vector.push\_back(ord);

DoOrder = true;

}

}

void NewOrderForm::on\_AddOrderButton\_clicked()

{

if(DoOrder){

if(Cafe->FreeTable() != 0){

int table = Cafe->FreeTable();

Cafe->ReservTable(table-1);

for(int i = 0; i<vector.size();i++){

PositionOrder pos = vector[i];

for(int j = 0;j<vector.size();j++){

if(i == j){

continue;

}

else if(pos == vector[j]){

pos = pos + vector[j];

vector.remove(i);

vector.insert(i,pos);

vector.remove(j);

j--;

}

}

}

Cafe->SetOrder(table-1, vector);

QString tablestr = "Ваш столик: ";

tablestr = tablestr + QString::number(table);

ui->TableLabel->setText(tablestr);

vector.clear();

DoOrder = false;

}

else{

ui->TableLabel->setText("Немає вільних столиків!");

}

}

}

//orderclass.cpp

#include "orderclass.h"

OrderClass::**OrderClass**()

{

for(int i = 0; i<15;i++){

Orders.push\_back(QVector<PositionOrder>());

}

}

QVector<PositionOrder> OrderClass::**GetOrder**(int index){

return Orders.at(index);

}

void OrderClass::**SetOrder**(int index, QVector<PositionOrder> Order){

IsOrders = true;

Orders.remove(index);

Orders.insert(index, Order);

}

void OrderClass::**ClearOrder**(int index){

QVector<PositionOrder> vector;

Orders.remove(index);

Orders.insert(index,vector);

}

//tableclass.cpp

#include "tableclass.h"

TableClass::**TableClass**()

{

for(int i = 0; i<15;i++){

FreeTables[i] = true;

}

}

void TableClass::**ReservTable**(int index){

if(index>=0 && index<15){

FreeTables[index] = false;

}

else{

throw "IndexOutRangeException";

}

}

int TableClass::**FreeTable**(){

for(int i = 0; i<15;i++){

if(FreeTables[i]){

return i+1;

}

}

return 0;

}

void TableClass::**UnReservTable**(int index){

if(index >= 0 && index<15){

FreeTables[index] = true;

}

else{

throw "IndexOutRangeException";

}

}

bool TableClass::**GetReservTable**(int index){

if((FreeTables[index]) == true){

return true;

}

else{

return false;

}

}

//addtoorderform.ui

<?xml version="1.0" encoding="UTF-8"?>

<ui version="4.0">

<class>AddToOrderForm</class>

<widget class="QMainWindow" name="AddToOrderForm">

<property name="windowModality">

<enum>Qt::ApplicationModal</enum>

</property>

<property name="geometry">

<rect>

<x>0</x>

<y>0</y>

<width>720</width>

<height>450</height>

</rect>

</property>

<property name="sizePolicy">

<sizepolicy hsizetype="Fixed" vsizetype="Fixed">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="minimumSize">

<size>

<width>720</width>

<height>450</height>

</size>

</property>

<property name="maximumSize">

<size>

<width>720</width>

<height>450</height>

</size>

</property>

<property name="font">

<font>

<family>Times New Roman</family>

<pointsize>10</pointsize>

</font>

</property>

<property name="windowTitle">

<string>AddToOrderForm</string>

</property>

<property name="windowIcon">

<iconset resource="resource.qrc">

<normaloff>:/res/img/cafe-server.gif</normaloff>:/res/img/cafe-server.gif</iconset>

</property>

<widget class="QWidget" name="centralwidget">

<widget class="QPushButton" name="ReturnButton">

<property name="geometry">

<rect>

<x>570</x>

<y>370</y>

<width>141</width>

<height>31</height>

</rect>

</property>

<property name="sizePolicy">

<sizepolicy hsizetype="Maximum" vsizetype="Maximum">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="font">

<font>

<weight>87</weight>

<bold>true</bold>

<underline>false</underline>

<strikeout>false</strikeout>

</font>

</property>

<property name="styleSheet">

<string notr="true">QPushButton

{

font-weight: 700;

color: white;

text-decoration: none;

padding: .8em 1em calc(.8em + 3px);

border-radius: 3px;

background: rgb(64,199,129);

box-shadow: 0 -3px rgb(53,167,110) inset;

transition: 0.2s;

}

QPushButton:pressed

{

background: rgb(33,147,90);

box-shadow: 0 3px rgb(33,147,90) inset;

}

QPushButton:hover

{

background: rgb(53, 167, 110);

}

QPushButton:focus {

/\*background-color: red;\*/

}</string>

</property>

<property name="text">

<string>Повернутися</string>

</property>

</widget>

<widget class="QListWidget" name="listWidget">

<property name="geometry">

<rect>

<x>10</x>

<y>50</y>

<width>561</width>

<height>311</height>

</rect>

</property>

</widget>

<widget class="QPushButton" name="AddOrderButton">

<property name="geometry">

<rect>

<x>10</x>

<y>370</y>

<width>181</width>

<height>31</height>

</rect>

</property>

<property name="sizePolicy">

<sizepolicy hsizetype="Maximum" vsizetype="Maximum">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="font">

<font>

<weight>87</weight>

<bold>true</bold>

<underline>false</underline>

<strikeout>false</strikeout>

</font>

</property>

<property name="styleSheet">

<string notr="true">QPushButton

{

font-weight: 700;

color: white;

text-decoration: none;

padding: .8em 1em calc(.8em + 3px);

border-radius: 3px;

background: rgb(64,199,129);

box-shadow: 0 -3px rgb(53,167,110) inset;

transition: 0.2s;

}

QPushButton:pressed

{

background: rgb(33,147,90);

box-shadow: 0 3px rgb(33,147,90) inset;

}

QPushButton:hover

{

background: rgb(53, 167, 110);

}

QPushButton:focus {

/\*background-color: red;\*/

}</string>

</property>

<property name="text">

<string>Додати до замовлення</string>

</property>

</widget>

<widget class="QSplitter" name="splitter">

<property name="geometry">

<rect>

<x>490</x>

<y>10</y>

<width>221</width>

<height>29</height>

</rect>

</property>

<property name="orientation">

<enum>Qt::Horizontal</enum>

</property>

<widget class="QLineEdit" name="lineEdit">

<property name="maximumSize">

<size>

<width>128</width>

<height>16777215</height>

</size>

</property>

</widget>

<widget class="QPushButton" name="AddButton">

<property name="sizePolicy">

<sizepolicy hsizetype="Preferred" vsizetype="Preferred">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="maximumSize">

<size>

<width>88</width>

<height>16777215</height>

</size>

</property>

<property name="font">

<font>

<weight>87</weight>

<bold>true</bold>

<underline>false</underline>

<strikeout>false</strikeout>

</font>

</property>

<property name="styleSheet">

<string notr="true">QPushButton

{

font-weight: 700;

color: white;

text-decoration: none;

padding: .8em 1em calc(.8em + 3px);

border-radius: 3px;

background: rgb(64,199,129);

box-shadow: 0 -3px rgb(53,167,110) inset;

transition: 0.2s;

}

QPushButton:pressed

{

background: rgb(33,147,90);

box-shadow: 0 3px rgb(33,147,90) inset;

}

QPushButton:hover

{

background: rgb(53, 167, 110);

}

QPushButton:focus {

/\*background-color: red;\*/

}</string>

</property>

<property name="text">

<string>Додати</string>

</property>

</widget>

</widget>

<widget class="QWidget" name="layoutWidget">

<property name="geometry">

<rect>

<x>10</x>

<y>10</y>

<width>278</width>

<height>31</height>

</rect>

</property>

<layout class="QHBoxLayout" name="horizontalLayout">

<item>

<widget class="QPushButton" name="MenuDishButton">

<property name="sizePolicy">

<sizepolicy hsizetype="Preferred" vsizetype="Preferred">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="font">

<font>

<weight>87</weight>

<bold>true</bold>

<underline>false</underline>

<strikeout>false</strikeout>

</font>

</property>

<property name="styleSheet">

<string notr="true">QPushButton

{

font-weight: 700;

color: white;

text-decoration: none;

padding: .8em 1em calc(.8em + 3px);

border-radius: 3px;

background: rgb(64,199,129);

box-shadow: 0 -3px rgb(53,167,110) inset;

transition: 0.2s;

}

QPushButton:pressed

{

background: rgb(33,147,90);

box-shadow: 0 3px rgb(33,147,90) inset;

}

QPushButton:hover

{

background: rgb(53, 167, 110);

}

QPushButton:focus {

/\*background-color: red;\*/

}</string>

</property>

<property name="text">

<string>Страви</string>

</property>

</widget>

</item>

<item>

<widget class="QPushButton" name="MenuDrinkButton">

<property name="sizePolicy">

<sizepolicy hsizetype="Preferred" vsizetype="Preferred">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="font">

<font>

<weight>87</weight>

<bold>true</bold>

<underline>false</underline>

<strikeout>false</strikeout>

</font>

</property>

<property name="styleSheet">

<string notr="true">QPushButton

{

font-weight: 700;

color: white;

text-decoration: none;

padding: .8em 1em calc(.8em + 3px);

border-radius: 3px;

background: rgb(64,199,129);

box-shadow: 0 -3px rgb(53,167,110) inset;

transition: 0.2s;

}

QPushButton:pressed

{

background: rgb(33,147,90);

box-shadow: 0 3px rgb(33,147,90) inset;

}

QPushButton:hover

{

background: rgb(53, 167, 110);

}

QPushButton:focus {

/\*background-color: red;\*/

}</string>

</property>

<property name="text">

<string>Напої</string>

</property>

</widget>

</item>

<item>

<widget class="QPushButton" name="MenuAlcoDrinkButton">

<property name="sizePolicy">

<sizepolicy hsizetype="Preferred" vsizetype="Preferred">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="font">

<font>

<weight>87</weight>

<bold>true</bold>

<underline>false</underline>

<strikeout>false</strikeout>

</font>

</property>

<property name="styleSheet">

<string notr="true">QPushButton

{

font-weight: 700;

color: white;

text-decoration: none;

padding: .8em 1em calc(.8em + 3px);

border-radius: 3px;

background: rgb(64,199,129);

box-shadow: 0 -3px rgb(53,167,110) inset;

transition: 0.2s;

}

QPushButton:pressed

{

background: rgb(33,147,90);

box-shadow: 0 3px rgb(33,147,90) inset;

}

QPushButton:hover

{

background: rgb(53, 167, 110);

}

QPushButton:focus {

/\*background-color: red;\*/

}</string>

</property>

<property name="text">

<string>Алкоголь</string>

</property>

</widget>

</item>

</layout>

</widget>

<widget class="QListWidget" name="TableList">

<property name="geometry">

<rect>

<x>580</x>

<y>50</y>

<width>131</width>

<height>311</height>

</rect>

</property>

<property name="font">

<font>

<pointsize>9</pointsize>

</font>

</property>

</widget>

<widget class="QLabel" name="TableLabel">

<property name="geometry">

<rect>

<x>220</x>

<y>380</y>

<width>261</width>

<height>16</height>

</rect>

</property>

<property name="font">

<font>

<pointsize>14</pointsize>

</font>

</property>

<property name="text">

<string>---</string>

</property>

</widget>

</widget>

<widget class="QMenuBar" name="menubar">

<property name="geometry">

<rect>

<x>0</x>

<y>0</y>

<width>720</width>

<height>21</height>

</rect>

</property>

</widget>

<widget class="QStatusBar" name="statusbar"/>

</widget>

<resources>

<include location="resource.qrc"/>

</resources>

<connections/>

</ui>

//closeorderform.ui

<?xml version="1.0" encoding="UTF-8"?>

<ui version="4.0">

<class>CloseOrderForm</class>

<widget class="QMainWindow" name="CloseOrderForm">

<property name="windowModality">

<enum>Qt::ApplicationModal</enum>

</property>

<property name="geometry">

<rect>

<x>0</x>

<y>0</y>

<width>720</width>

<height>450</height>

</rect>

</property>

<property name="sizePolicy">

<sizepolicy hsizetype="Fixed" vsizetype="Fixed">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="minimumSize">

<size>

<width>720</width>

<height>450</height>

</size>

</property>

<property name="maximumSize">

<size>

<width>720</width>

<height>450</height>

</size>

</property>

<property name="font">

<font>

<family>Times New Roman</family>

<pointsize>10</pointsize>

</font>

</property>

<property name="windowTitle">

<string>CloseOrderForm</string>

</property>

<property name="windowIcon">

<iconset resource="resource.qrc">

<normaloff>:/res/img/cafe-server.gif</normaloff>:/res/img/cafe-server.gif</iconset>

</property>

<widget class="QWidget" name="centralwidget">

<widget class="QPushButton" name="ReturnButton">

<property name="geometry">

<rect>

<x>570</x>

<y>370</y>

<width>141</width>

<height>31</height>

</rect>

</property>

<property name="sizePolicy">

<sizepolicy hsizetype="Maximum" vsizetype="Maximum">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="font">

<font>

<weight>87</weight>

<bold>true</bold>

<underline>false</underline>

<strikeout>false</strikeout>

</font>

</property>

<property name="styleSheet">

<string notr="true">QPushButton

{

font-weight: 700;

color: white;

text-decoration: none;

padding: .8em 1em calc(.8em + 3px);

border-radius: 3px;

background: rgb(64,199,129);

box-shadow: 0 -3px rgb(53,167,110) inset;

transition: 0.2s;

}

QPushButton:pressed

{

background: rgb(33,147,90);

box-shadow: 0 3px rgb(33,147,90) inset;

}

QPushButton:hover

{

background: rgb(53, 167, 110);

}

QPushButton:focus {

/\*background-color: red;\*/

}</string>

</property>

<property name="text">

<string>Повернутися</string>

</property>

</widget>

<widget class="QListWidget" name="TableList">

<property name="geometry">

<rect>

<x>10</x>

<y>10</y>

<width>321</width>

<height>391</height>

</rect>

</property>

</widget>

<widget class="QPushButton" name="CloseButton">

<property name="geometry">

<rect>

<x>340</x>

<y>370</y>

<width>141</width>

<height>31</height>

</rect>

</property>

<property name="sizePolicy">

<sizepolicy hsizetype="Maximum" vsizetype="Maximum">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="font">

<font>

<weight>87</weight>

<bold>true</bold>

<underline>false</underline>

<strikeout>false</strikeout>

</font>

</property>

<property name="styleSheet">

<string notr="true">QPushButton

{

font-weight: 700;

color: white;

text-decoration: none;

padding: .8em 1em calc(.8em + 3px);

border-radius: 3px;

background: rgb(64,199,129);

box-shadow: 0 -3px rgb(53,167,110) inset;

transition: 0.2s;

}

QPushButton:pressed

{

background: rgb(33,147,90);

box-shadow: 0 3px rgb(33,147,90) inset;

}

QPushButton:hover

{

background: rgb(53, 167, 110);

}

QPushButton:focus {

/\*background-color: red;\*/

}</string>

</property>

<property name="text">

<string>Закрити</string>

</property>

</widget>

<widget class="QLabel" name="TableLabel">

<property name="geometry">

<rect>

<x>360</x>

<y>20</y>

<width>191</width>

<height>31</height>

</rect>

</property>

<property name="text">

<string>&lt;html&gt;&lt;head/&gt;&lt;body&gt;&lt;p&gt;&lt;span style=&quot; font-size:14pt; color:#0f4703;&quot;&gt;Зайняті столики:&lt;/span&gt;&lt;/p&gt;&lt;/body&gt;&lt;/html&gt;</string>

</property>

</widget>

<widget class="QLabel" name="label">

<property name="geometry">

<rect>

<x>440</x>

<y>80</y>

<width>191</width>

<height>151</height>

</rect>

</property>

<property name="text">

<string/>

</property>

<property name="pixmap">

<pixmap resource="resource.qrc">:/res/img/cup\_PNG1964-170x170.png</pixmap>

</property>

</widget>

<widget class="QLabel" name="MoneyLabel">

<property name="geometry">

<rect>

<x>340</x>

<y>320</y>

<width>371</width>

<height>31</height>

</rect>

</property>

<property name="font">

<font>

<pointsize>14</pointsize>

</font>

</property>

<property name="text">

<string>&lt;html&gt;&lt;head/&gt;&lt;body&gt;&lt;p&gt;&lt;span style=&quot; font-size:14pt; color:#0f4703;&quot;&gt;До сплати:&lt;/span&gt;&lt;/p&gt;&lt;/body&gt;&lt;/html&gt;</string>

</property>

</widget>

</widget>

<widget class="QMenuBar" name="menubar">

<property name="geometry">

<rect>

<x>0</x>

<y>0</y>

<width>720</width>

<height>21</height>

</rect>

</property>

</widget>

<widget class="QStatusBar" name="statusbar"/>

</widget>

<resources>

<include location="resource.qrc"/>

</resources>

<connections/>

</ui>

//deleteorderform.ui

<?xml version="1.0" encoding="UTF-8"?>

<ui version="4.0">

<class>DeleteOrder</class>

<widget class="QMainWindow" name="DeleteOrder">

<property name="windowModality">

<enum>Qt::ApplicationModal</enum>

</property>

<property name="geometry">

<rect>

<x>0</x>

<y>0</y>

<width>720</width>

<height>450</height>

</rect>

</property>

<property name="sizePolicy">

<sizepolicy hsizetype="Fixed" vsizetype="Fixed">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="minimumSize">

<size>

<width>720</width>

<height>450</height>

</size>

</property>

<property name="maximumSize">

<size>

<width>720</width>

<height>450</height>

</size>

</property>

<property name="font">

<font>

<family>Times New Roman</family>

<pointsize>10</pointsize>

</font>

</property>

<property name="windowTitle">

<string>DeleteOrderForm</string>

</property>

<property name="windowIcon">

<iconset resource="resource.qrc">

<normaloff>:/res/img/cafe-server.gif</normaloff>:/res/img/cafe-server.gif</iconset>

</property>

<widget class="QWidget" name="centralwidget">

<widget class="QPushButton" name="ReturnButton\_2">

<property name="geometry">

<rect>

<x>570</x>

<y>370</y>

<width>141</width>

<height>31</height>

</rect>

</property>

<property name="sizePolicy">

<sizepolicy hsizetype="Maximum" vsizetype="Maximum">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="font">

<font>

<weight>87</weight>

<bold>true</bold>

<underline>false</underline>

<strikeout>false</strikeout>

</font>

</property>

<property name="styleSheet">

<string notr="true">QPushButton

{

font-weight: 700;

color: white;

text-decoration: none;

padding: .8em 1em calc(.8em + 3px);

border-radius: 3px;

background: rgb(64,199,129);

box-shadow: 0 -3px rgb(53,167,110) inset;

transition: 0.2s;

}

QPushButton:pressed

{

background: rgb(33,147,90);

box-shadow: 0 3px rgb(33,147,90) inset;

}

QPushButton:hover

{

background: rgb(53, 167, 110);

}

QPushButton:focus {

/\*background-color: red;\*/

}</string>

</property>

<property name="text">

<string>Повернутися</string>

</property>

</widget>

<widget class="QListWidget" name="TableList">

<property name="geometry">

<rect>

<x>10</x>

<y>10</y>

<width>321</width>

<height>391</height>

</rect>

</property>

</widget>

<widget class="QLabel" name="TableLabel">

<property name="geometry">

<rect>

<x>360</x>

<y>20</y>

<width>191</width>

<height>31</height>

</rect>

</property>

<property name="text">

<string>&lt;html&gt;&lt;head/&gt;&lt;body&gt;&lt;p&gt;&lt;span style=&quot; font-size:14pt; color:#0f4703;&quot;&gt;Зайняті столики:&lt;/span&gt;&lt;/p&gt;&lt;/body&gt;&lt;/html&gt;</string>

</property>

</widget>

<widget class="QLabel" name="label">

<property name="geometry">

<rect>

<x>440</x>

<y>110</y>

<width>191</width>

<height>151</height>

</rect>

</property>

<property name="text">

<string/>

</property>

<property name="pixmap">

<pixmap resource="resource.qrc">:/res/img/cup\_PNG1964-170x170.png</pixmap>

</property>

</widget>

<widget class="QPushButton" name="DeleteButton">

<property name="geometry">

<rect>

<x>340</x>

<y>370</y>

<width>141</width>

<height>31</height>

</rect>

</property>

<property name="sizePolicy">

<sizepolicy hsizetype="Maximum" vsizetype="Maximum">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="font">

<font>

<weight>87</weight>

<bold>true</bold>

<underline>false</underline>

<strikeout>false</strikeout>

</font>

</property>

<property name="styleSheet">

<string notr="true">QPushButton

{

font-weight: 700;

color: white;

text-decoration: none;

padding: .8em 1em calc(.8em + 3px);

border-radius: 3px;

background: rgb(64,199,129);

box-shadow: 0 -3px rgb(53,167,110) inset;

transition: 0.2s;

}

QPushButton:pressed

{

background: rgb(33,147,90);

box-shadow: 0 3px rgb(33,147,90) inset;

}

QPushButton:hover

{

background: rgb(53, 167, 110);

}

QPushButton:focus {

/\*background-color: red;\*/

}</string>

</property>

<property name="text">

<string>Видалити</string>

</property>

</widget>

</widget>

<widget class="QMenuBar" name="menubar">

<property name="geometry">

<rect>

<x>0</x>

<y>0</y>

<width>720</width>

<height>21</height>

</rect>

</property>

</widget>

<widget class="QStatusBar" name="statusbar"/>

</widget>

<resources>

<include location="resource.qrc"/>

</resources>

<connections/>

</ui>

//infoaboutorder.ui

<?xml version="1.0" encoding="UTF-8"?>

<ui version="4.0">

<class>InfoAboutOrderForm</class>

<widget class="QMainWindow" name="InfoAboutOrderForm">

<property name="windowModality">

<enum>Qt::ApplicationModal</enum>

</property>

<property name="geometry">

<rect>

<x>0</x>

<y>0</y>

<width>720</width>

<height>450</height>

</rect>

</property>

<property name="sizePolicy">

<sizepolicy hsizetype="Fixed" vsizetype="Fixed">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="minimumSize">

<size>

<width>720</width>

<height>450</height>

</size>

</property>

<property name="maximumSize">

<size>

<width>720</width>

<height>450</height>

</size>

</property>

<property name="font">

<font>

<family>Times New Roman</family>

<pointsize>10</pointsize>

</font>

</property>

<property name="windowTitle">

<string>InfoAboutOrder</string>

</property>

<property name="windowIcon">

<iconset resource="resource.qrc">

<normaloff>:/res/img/cafe-server.gif</normaloff>:/res/img/cafe-server.gif</iconset>

</property>

<widget class="QWidget" name="centralwidget">

<widget class="QListWidget" name="TableList">

<property name="geometry">

<rect>

<x>10</x>

<y>10</y>

<width>111</width>

<height>351</height>

</rect>

</property>

</widget>

<widget class="QListWidget" name="OrdersList">

<property name="geometry">

<rect>

<x>130</x>

<y>10</y>

<width>581</width>

<height>351</height>

</rect>

</property>

</widget>

<widget class="QPushButton" name="CheckButton">

<property name="geometry">

<rect>

<x>10</x>

<y>370</y>

<width>461</width>

<height>31</height>

</rect>

</property>

<property name="font">

<font>

<weight>87</weight>

<bold>true</bold>

<underline>false</underline>

<strikeout>false</strikeout>

</font>

</property>

<property name="styleSheet">

<string notr="true">QPushButton

{

font-weight: 700;

color: white;

text-decoration: none;

padding: .8em 1em calc(.8em + 3px);

border-radius: 3px;

background: rgb(64,199,129);

box-shadow: 0 -3px rgb(53,167,110) inset;

transition: 0.2s;

}

QPushButton:pressed

{

background: rgb(33,147,90);

box-shadow: 0 3px rgb(33,147,90) inset;

}

QPushButton:hover

{

background: rgb(53, 167, 110);

}

QPushButton:focus {

/\*background-color: red;\*/

}</string>

</property>

<property name="text">

<string>Вивести</string>

</property>

</widget>

<widget class="QPushButton" name="ReturnButton">

<property name="geometry">

<rect>

<x>490</x>

<y>370</y>

<width>221</width>

<height>31</height>

</rect>

</property>

<property name="font">

<font>

<weight>87</weight>

<bold>true</bold>

<underline>false</underline>

<strikeout>false</strikeout>

</font>

</property>

<property name="styleSheet">

<string notr="true">QPushButton

{

font-weight: 700;

color: white;

text-decoration: none;

padding: .8em 1em calc(.8em + 3px);

border-radius: 3px;

background: rgb(64,199,129);

box-shadow: 0 -3px rgb(53,167,110) inset;

transition: 0.2s;

}

QPushButton:pressed

{

background: rgb(33,147,90);

box-shadow: 0 3px rgb(33,147,90) inset;

}

QPushButton:hover

{

background: rgb(53, 167, 110);

}

QPushButton:focus {

/\*background-color: red;\*/

}</string>

</property>

<property name="text">

<string>Повернутися</string>

</property>

</widget>

</widget>

<widget class="QMenuBar" name="menubar">

<property name="geometry">

<rect>

<x>0</x>

<y>0</y>

<width>720</width>

<height>21</height>

</rect>

</property>

</widget>

<widget class="QStatusBar" name="statusbar"/>

</widget>

<resources>

<include location="resource.qrc"/>

</resources>

<connections/>

</ui>

//infoform.ui

<?xml version="1.0" encoding="UTF-8"?>

<ui version="4.0">

<class>InfoForm</class>

<widget class="QMainWindow" name="InfoForm">

<property name="windowModality">

<enum>Qt::ApplicationModal</enum>

</property>

<property name="geometry">

<rect>

<x>0</x>

<y>0</y>

<width>720</width>

<height>450</height>

</rect>

</property>

<property name="sizePolicy">

<sizepolicy hsizetype="Fixed" vsizetype="Fixed">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="minimumSize">

<size>

<width>720</width>

<height>450</height>

</size>

</property>

<property name="maximumSize">

<size>

<width>720</width>

<height>450</height>

</size>

</property>

<property name="font">

<font>

<family>Times New Roman</family>

<pointsize>10</pointsize>

</font>

</property>

<property name="windowTitle">

<string>InfoForm</string>

</property>

<property name="windowIcon">

<iconset resource="resource.qrc">

<normaloff>:/res/img/cafe-server.gif</normaloff>:/res/img/cafe-server.gif</iconset>

</property>

<widget class="QWidget" name="centralwidget">

<widget class="QListWidget" name="TableList">

<property name="geometry">

<rect>

<x>10</x>

<y>50</y>

<width>341</width>

<height>351</height>

</rect>

</property>

</widget>

<widget class="QLabel" name="TableLabel">

<property name="geometry">

<rect>

<x>90</x>

<y>10</y>

<width>161</width>

<height>31</height>

</rect>

</property>

<property name="font">

<font>

<pointsize>14</pointsize>

</font>

</property>

<property name="text">

<string>&lt;html&gt;&lt;head/&gt;&lt;body&gt;&lt;p&gt;&lt;span style=&quot; color:#0b3601;&quot;&gt;Зайняті столики&lt;/span&gt;&lt;/p&gt;&lt;/body&gt;&lt;/html&gt;</string>

</property>

</widget>

<widget class="QPushButton" name="ReturnButton">

<property name="geometry">

<rect>

<x>560</x>

<y>370</y>

<width>151</width>

<height>31</height>

</rect>

</property>

<property name="sizePolicy">

<sizepolicy hsizetype="Maximum" vsizetype="Maximum">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="font">

<font>

<weight>87</weight>

<bold>true</bold>

<underline>false</underline>

<strikeout>false</strikeout>

</font>

</property>

<property name="styleSheet">

<string notr="true">QPushButton

{

font-weight: 700;

color: white;

text-decoration: none;

padding: .8em 1em calc(.8em + 3px);

border-radius: 3px;

background: rgb(64,199,129);

box-shadow: 0 -3px rgb(53,167,110) inset;

transition: 0.2s;

}

QPushButton:pressed

{

background: rgb(33,147,90);

box-shadow: 0 3px rgb(33,147,90) inset;

}

QPushButton:hover

{

background: rgb(53, 167, 110);

}

QPushButton:focus {

/\*background-color: red;\*/

}</string>

</property>

<property name="text">

<string>Повернутися</string>

</property>

</widget>

<widget class="QLabel" name="label">

<property name="geometry">

<rect>

<x>450</x>

<y>140</y>

<width>181</width>

<height>161</height>

</rect>

</property>

<property name="text">

<string/>

</property>

<property name="pixmap">

<pixmap resource="resource.qrc">:/res/img/cup\_PNG1964-170x170.png</pixmap>

</property>

</widget>

</widget>

<widget class="QMenuBar" name="menubar">

<property name="geometry">

<rect>

<x>0</x>

<y>0</y>

<width>720</width>

<height>21</height>

</rect>

</property>

</widget>

<widget class="QStatusBar" name="statusbar"/>

</widget>

<resources>

<include location="resource.qrc"/>

</resources>

<connections/>

</ui>

//mainform.ui

<?xml version="1.0" encoding="UTF-8"?>

<ui version="4.0">

<class>MainForm</class>

<widget class="QMainWindow" name="MainForm">

<property name="windowModality">

<enum>Qt::NonModal</enum>

</property>

<property name="enabled">

<bool>true</bool>

</property>

<property name="geometry">

<rect>

<x>0</x>

<y>0</y>

<width>720</width>

<height>450</height>

</rect>

</property>

<property name="sizePolicy">

<sizepolicy hsizetype="Fixed" vsizetype="Fixed">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="minimumSize">

<size>

<width>720</width>

<height>450</height>

</size>

</property>

<property name="maximumSize">

<size>

<width>720</width>

<height>450</height>

</size>

</property>

<property name="font">

<font>

<family>Times New Roman</family>

<pointsize>12</pointsize>

<weight>75</weight>

<bold>true</bold>

</font>

</property>

<property name="mouseTracking">

<bool>false</bool>

</property>

<property name="windowTitle">

<string>MainForm</string>

</property>

<property name="windowIcon">

<iconset resource="resource.qrc">

<normaloff>:/res/img/cafe-server.gif</normaloff>:/res/img/cafe-server.gif</iconset>

</property>

<property name="autoFillBackground">

<bool>false</bool>

</property>

<property name="styleSheet">

<string notr="true"/>

</property>

<property name="toolButtonStyle">

<enum>Qt::ToolButtonTextOnly</enum>

</property>

<property name="animated">

<bool>true</bool>

</property>

<widget class="QWidget" name="centralWidget">

<layout class="QGridLayout" name="gridLayout">

<item row="1" column="4">

<spacer name="horizontalSpacer\_2">

<property name="orientation">

<enum>Qt::Horizontal</enum>

</property>

<property name="sizeType">

<enum>QSizePolicy::Fixed</enum>

</property>

<property name="sizeHint" stdset="0">

<size>

<width>5</width>

<height>20</height>

</size>

</property>

</spacer>

</item>

<item row="3" column="0">

<spacer name="horizontalSpacer\_3">

<property name="orientation">

<enum>Qt::Horizontal</enum>

</property>

<property name="sizeType">

<enum>QSizePolicy::Fixed</enum>

</property>

<property name="sizeHint" stdset="0">

<size>

<width>5</width>

<height>20</height>

</size>

</property>

</spacer>

</item>

<item row="2" column="3">

<spacer name="verticalSpacer">

<property name="orientation">

<enum>Qt::Vertical</enum>

</property>

<property name="sizeType">

<enum>QSizePolicy::Fixed</enum>

</property>

<property name="sizeHint" stdset="0">

<size>

<width>20</width>

<height>5</height>

</size>

</property>

</spacer>

</item>

<item row="4" column="3">

<spacer name="verticalSpacer\_2">

<property name="orientation">

<enum>Qt::Vertical</enum>

</property>

<property name="sizeType">

<enum>QSizePolicy::Fixed</enum>

</property>

<property name="sizeHint" stdset="0">

<size>

<width>20</width>

<height>5</height>

</size>

</property>

</spacer>

</item>

<item row="1" column="6">

<spacer name="horizontalSpacer\_4">

<property name="orientation">

<enum>Qt::Horizontal</enum>

</property>

<property name="sizeType">

<enum>QSizePolicy::Fixed</enum>

</property>

<property name="sizeHint" stdset="0">

<size>

<width>5</width>

<height>20</height>

</size>

</property>

</spacer>

</item>

<item row="1" column="1">

<widget class="QPushButton" name="NewOrderButton">

<property name="sizePolicy">

<sizepolicy hsizetype="MinimumExpanding" vsizetype="MinimumExpanding">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="font">

<font>

<weight>87</weight>

<bold>true</bold>

<underline>false</underline>

<strikeout>false</strikeout>

</font>

</property>

<property name="autoFillBackground">

<bool>false</bool>

</property>

<property name="styleSheet">

<string notr="true">QPushButton

{

font-weight: 700;

color: white;

text-decoration: none;

padding: .8em 1em calc(.8em + 3px);

border-radius: 3px;

background: rgb(64,199,129);

box-shadow: 0 -3px rgb(53,167,110) inset;

transition: 0.2s;

}

QPushButton:pressed

{

background: rgb(33,147,90);

box-shadow: 0 3px rgb(33,147,90) inset;

}

QPushButton:hover

{

background: rgb(53, 167, 110);

}

QPushButton:focus {

/\*background-color: red;\*/

}</string>

</property>

<property name="text">

<string>Нове замовлення</string>

</property>

</widget>

</item>

<item row="1" column="3">

<widget class="QPushButton" name="CloseOrderButton">

<property name="sizePolicy">

<sizepolicy hsizetype="MinimumExpanding" vsizetype="MinimumExpanding">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="font">

<font>

<weight>87</weight>

<bold>true</bold>

<underline>false</underline>

<strikeout>false</strikeout>

</font>

</property>

<property name="autoFillBackground">

<bool>false</bool>

</property>

<property name="styleSheet">

<string notr="true">QPushButton

{

font-weight: 700;

color: white;

text-decoration: none;

padding: .8em 1em calc(.8em + 3px);

border-radius: 3px;

background: rgb(64,199,129);

box-shadow: 0 -3px rgb(53,167,110) inset;

transition: 0.2s;

}

QPushButton:pressed

{

background: rgb(33,147,90);

box-shadow: 0 3px rgb(33,147,90) inset;

}

QPushButton:hover

{

background: rgb(53, 167, 110);

}

QPushButton:focus {

/\*background-color: red;\*/

}</string>

</property>

<property name="text">

<string>Закрити замовлення</string>

</property>

</widget>

</item>

<item row="3" column="1">

<widget class="QPushButton" name="AddToOrderButton">

<property name="sizePolicy">

<sizepolicy hsizetype="MinimumExpanding" vsizetype="MinimumExpanding">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="font">

<font>

<weight>87</weight>

<bold>true</bold>

<underline>false</underline>

<strikeout>false</strikeout>

</font>

</property>

<property name="autoFillBackground">

<bool>false</bool>

</property>

<property name="styleSheet">

<string notr="true">QPushButton

{

font-weight: 700;

color: white;

text-decoration: none;

padding: .8em 1em calc(.8em + 3px);

border-radius: 3px;

background: rgb(64,199,129);

box-shadow: 0 -3px rgb(53,167,110) inset;

transition: 0.2s;

}

QPushButton:pressed

{

background: rgb(33,147,90);

box-shadow: 0 3px rgb(33,147,90) inset;

}

QPushButton:hover

{

background: rgb(53, 167, 110);

}

QPushButton:focus {

/\*background-color: red;\*/

}</string>

</property>

<property name="text">

<string>Додати до замовлення</string>

</property>

</widget>

</item>

<item row="3" column="3">

<widget class="QPushButton" name="DeleteOrderButton">

<property name="sizePolicy">

<sizepolicy hsizetype="MinimumExpanding" vsizetype="MinimumExpanding">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="font">

<font>

<weight>87</weight>

<bold>true</bold>

<underline>false</underline>

<strikeout>false</strikeout>

</font>

</property>

<property name="autoFillBackground">

<bool>false</bool>

</property>

<property name="styleSheet">

<string notr="true">QPushButton

{

font-weight: 700;

color: white;

text-decoration: none;

padding: .8em 1em calc(.8em + 3px);

border-radius: 3px;

background: rgb(64,199,129);

box-shadow: 0 -3px rgb(53,167,110) inset;

transition: 0.2s;

}

QPushButton:pressed

{

background: rgb(33,147,90);

box-shadow: 0 3px rgb(33,147,90) inset;

}

QPushButton:hover

{

background: rgb(53, 167, 110);

}

QPushButton:focus {

/\*background-color: red;\*/

}</string>

</property>

<property name="text">

<string>Видалити замовлення</string>

</property>

</widget>

</item>

<item row="1" column="5">

<widget class="QPushButton" name="InfoAboutOrderButton">

<property name="sizePolicy">

<sizepolicy hsizetype="MinimumExpanding" vsizetype="MinimumExpanding">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="font">

<font>

<weight>87</weight>

<bold>true</bold>

<underline>false</underline>

<strikeout>false</strikeout>

</font>

</property>

<property name="autoFillBackground">

<bool>false</bool>

</property>

<property name="styleSheet">

<string notr="true">QPushButton

{

font-weight: 700;

color: white;

text-decoration: none;

padding: .8em 1em calc(.8em + 3px);

border-radius: 3px;

background: rgb(64,199,129);

box-shadow: 0 -3px rgb(53,167,110) inset;

transition: 0.2s;

}

QPushButton:pressed

{

background: rgb(33,147,90);

box-shadow: 0 3px rgb(33,147,90) inset;

}

QPushButton:hover

{

background: rgb(53, 167, 110);

}

QPushButton:focus {

/\*background-color: red;\*/

}</string>

</property>

<property name="text">

<string>Інформація про столик</string>

</property>

</widget>

</item>

<item row="3" column="5">

<widget class="QPushButton" name="Info">

<property name="sizePolicy">

<sizepolicy hsizetype="MinimumExpanding" vsizetype="MinimumExpanding">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="font">

<font>

<weight>87</weight>

<bold>true</bold>

<underline>false</underline>

<strikeout>false</strikeout>

</font>

</property>

<property name="autoFillBackground">

<bool>false</bool>

</property>

<property name="styleSheet">

<string notr="true">QPushButton

{

font-weight: 700;

color: white;

text-decoration: none;

padding: .8em 1em calc(.8em + 3px);

border-radius: 3px;

background: rgb(64,199,129);

box-shadow: 0 -3px rgb(53,167,110) inset;

transition: 0.2s;

}

QPushButton:pressed

{

background: rgb(33,147,90);

box-shadow: 0 3px rgb(33,147,90) inset;

}

QPushButton:hover

{

background: rgb(53, 167, 110);

}

QPushButton:focus {

/\*background-color: red;\*/

}</string>

</property>

<property name="text">

<string>Інформація</string>

</property>

</widget>

</item>

<item row="1" column="2">

<spacer name="horizontalSpacer">

<property name="orientation">

<enum>Qt::Horizontal</enum>

</property>

<property name="sizeType">

<enum>QSizePolicy::Fixed</enum>

</property>

<property name="sizeHint" stdset="0">

<size>

<width>5</width>

<height>20</height>

</size>

</property>

</spacer>

</item>

<item row="0" column="3">

<spacer name="verticalSpacer\_3">

<property name="orientation">

<enum>Qt::Vertical</enum>

</property>

<property name="sizeType">

<enum>QSizePolicy::Fixed</enum>

</property>

<property name="sizeHint" stdset="0">

<size>

<width>20</width>

<height>5</height>

</size>

</property>

</spacer>

</item>

</layout>

</widget>

<widget class="QMenuBar" name="menuBar">

<property name="geometry">

<rect>

<x>0</x>

<y>0</y>

<width>720</width>

<height>22</height>

</rect>

</property>

<widget class="QMenu" name="menu">

<property name="tearOffEnabled">

<bool>false</bool>

</property>

<property name="title">

<string>Додаткові дії</string>

</property>

<property name="icon">

<iconset resource="resource.qrc">

<normaloff>:/res/img/Без названия.jpg</normaloff>:/res/img/Без названия.jpg</iconset>

</property>

<property name="separatorsCollapsible">

<bool>false</bool>

</property>

<property name="toolTipsVisible">

<bool>false</bool>

</property>

<addaction name="InfoAboutDev"/>

<addaction name="Exit"/>

</widget>

<addaction name="menu"/>

</widget>

<widget class="QStatusBar" name="statusBar"/>

<action name="InfoAboutDev">

<property name="icon">

<iconset resource="resource.qrc">

<normaloff>:/res/img/Button White Info.png</normaloff>:/res/img/Button White Info.png</iconset>

</property>

<property name="text">

<string>Інформація про розробника</string>

</property>

<property name="toolTip">

<string>Інформація</string>

</property>

</action>

<action name="Exit">

<property name="icon">

<iconset resource="resource.qrc">

<normaloff>:/res/img/exit.png</normaloff>:/res/img/exit.png</iconset>

</property>

<property name="text">

<string>Вийти</string>

</property>

</action>

</widget>

<layoutdefault spacing="6" margin="11"/>

<resources>

<include location="resource.qrc"/>

</resources>

<connections/>

</ui>

//neworderform.ui

<?xml version="1.0" encoding="UTF-8"?>

<ui version="4.0">

<class>NewOrderForm</class>

<widget class="QMainWindow" name="NewOrderForm">

<property name="windowModality">

<enum>Qt::ApplicationModal</enum>

</property>

<property name="geometry">

<rect>

<x>0</x>

<y>0</y>

<width>720</width>

<height>450</height>

</rect>

</property>

<property name="sizePolicy">

<sizepolicy hsizetype="Fixed" vsizetype="Fixed">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="minimumSize">

<size>

<width>720</width>

<height>450</height>

</size>

</property>

<property name="maximumSize">

<size>

<width>720</width>

<height>450</height>

</size>

</property>

<property name="font">

<font>

<family>Times New Roman</family>

<pointsize>10</pointsize>

<weight>75</weight>

<bold>true</bold>

</font>

</property>

<property name="windowTitle">

<string>NewOrderForm</string>

</property>

<property name="windowIcon">

<iconset resource="resource.qrc">

<normaloff>:/res/img/cafe-server.gif</normaloff>:/res/img/cafe-server.gif</iconset>

</property>

<property name="tabShape">

<enum>QTabWidget::Rounded</enum>

</property>

<widget class="QWidget" name="centralwidget">

<widget class="QPushButton" name="ReturnButton">

<property name="geometry">

<rect>

<x>570</x>

<y>370</y>

<width>141</width>

<height>31</height>

</rect>

</property>

<property name="sizePolicy">

<sizepolicy hsizetype="Maximum" vsizetype="Maximum">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="font">

<font>

<weight>87</weight>

<bold>true</bold>

<underline>false</underline>

<strikeout>false</strikeout>

</font>

</property>

<property name="styleSheet">

<string notr="true">QPushButton

{

font-weight: 700;

color: white;

text-decoration: none;

padding: .8em 1em calc(.8em + 3px);

border-radius: 3px;

background: rgb(64,199,129);

box-shadow: 0 -3px rgb(53,167,110) inset;

transition: 0.2s;

}

QPushButton:pressed

{

background: rgb(33,147,90);

box-shadow: 0 3px rgb(33,147,90) inset;

}

QPushButton:hover

{

background: rgb(53, 167, 110);

}

QPushButton:focus {

/\*background-color: red;\*/

}</string>

</property>

<property name="text">

<string>Повернутися</string>

</property>

</widget>

<widget class="QListWidget" name="listWidget">

<property name="geometry">

<rect>

<x>10</x>

<y>50</y>

<width>701</width>

<height>311</height>

</rect>

</property>

</widget>

<widget class="QLabel" name="TableLabel">

<property name="geometry">

<rect>

<x>200</x>

<y>380</y>

<width>321</width>

<height>21</height>

</rect>

</property>

<property name="font">

<font>

<pointsize>14</pointsize>

</font>

</property>

<property name="text">

<string>&lt;html&gt;&lt;head/&gt;&lt;body&gt;&lt;p&gt;&lt;span style=&quot; color:#116604;&quot;&gt;Ваш столик:&lt;/span&gt;&lt;/p&gt;&lt;/body&gt;&lt;/html&gt;</string>

</property>

</widget>

<widget class="QPushButton" name="AddOrderButton">

<property name="geometry">

<rect>

<x>10</x>

<y>370</y>

<width>181</width>

<height>31</height>

</rect>

</property>

<property name="sizePolicy">

<sizepolicy hsizetype="Maximum" vsizetype="Maximum">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="font">

<font>

<weight>87</weight>

<bold>true</bold>

<underline>false</underline>

<strikeout>false</strikeout>

</font>

</property>

<property name="styleSheet">

<string notr="true">QPushButton

{

font-weight: 700;

color: white;

text-decoration: none;

padding: .8em 1em calc(.8em + 3px);

border-radius: 3px;

background: rgb(64,199,129);

box-shadow: 0 -3px rgb(53,167,110) inset;

transition: 0.2s;

}

QPushButton:pressed

{

background: rgb(33,147,90);

box-shadow: 0 3px rgb(33,147,90) inset;

}

QPushButton:hover

{

background: rgb(53, 167, 110);

}

QPushButton:focus {

/\*background-color: red;\*/

}</string>

</property>

<property name="text">

<string>Зробити замовлення</string>

</property>

</widget>

<widget class="QWidget" name="layoutWidget">

<property name="geometry">

<rect>

<x>10</x>

<y>10</y>

<width>278</width>

<height>31</height>

</rect>

</property>

<layout class="QHBoxLayout" name="horizontalLayout">

<item>

<widget class="QPushButton" name="MenuDishButton">

<property name="sizePolicy">

<sizepolicy hsizetype="Preferred" vsizetype="Preferred">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="font">

<font>

<weight>87</weight>

<bold>true</bold>

<underline>false</underline>

<strikeout>false</strikeout>

</font>

</property>

<property name="styleSheet">

<string notr="true">QPushButton

{

font-weight: 700;

color: white;

text-decoration: none;

padding: .8em 1em calc(.8em + 3px);

border-radius: 3px;

background: rgb(64,199,129);

box-shadow: 0 -3px rgb(53,167,110) inset;

transition: 0.2s;

}

QPushButton:pressed

{

background: rgb(33,147,90);

box-shadow: 0 3px rgb(33,147,90) inset;

}

QPushButton:hover

{

background: rgb(53, 167, 110);

}

QPushButton:focus {

/\*background-color: red;\*/

}</string>

</property>

<property name="text">

<string>Страви</string>

</property>

</widget>

</item>

<item>

<widget class="QPushButton" name="MenuDrinkButton">

<property name="sizePolicy">

<sizepolicy hsizetype="Preferred" vsizetype="Preferred">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="font">

<font>

<weight>87</weight>

<bold>true</bold>

<underline>false</underline>

<strikeout>false</strikeout>

</font>

</property>

<property name="styleSheet">

<string notr="true">QPushButton

{

font-weight: 700;

color: white;

text-decoration: none;

padding: .8em 1em calc(.8em + 3px);

border-radius: 3px;

background: rgb(64,199,129);

box-shadow: 0 -3px rgb(53,167,110) inset;

transition: 0.2s;

}

QPushButton:pressed

{

background: rgb(33,147,90);

box-shadow: 0 3px rgb(33,147,90) inset;

}

QPushButton:hover

{

background: rgb(53, 167, 110);

}

QPushButton:focus {

/\*background-color: red;\*/

}</string>

</property>

<property name="text">

<string>Напої</string>

</property>

</widget>

</item>

<item>

<widget class="QPushButton" name="MenuAlcoDrinkButton">

<property name="sizePolicy">

<sizepolicy hsizetype="Preferred" vsizetype="Preferred">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="font">

<font>

<weight>87</weight>

<bold>true</bold>

<underline>false</underline>

<strikeout>false</strikeout>

</font>

</property>

<property name="styleSheet">

<string notr="true">QPushButton

{

font-weight: 700;

color: white;

text-decoration: none;

padding: .8em 1em calc(.8em + 3px);

border-radius: 3px;

background: rgb(64,199,129);

box-shadow: 0 -3px rgb(53,167,110) inset;

transition: 0.2s;

}

QPushButton:pressed

{

background: rgb(33,147,90);

box-shadow: 0 3px rgb(33,147,90) inset;

}

QPushButton:hover

{

background: rgb(53, 167, 110);

}

QPushButton:focus {

/\*background-color: red;\*/

}</string>

</property>

<property name="text">

<string>Алкоголь</string>

</property>

</widget>

</item>

</layout>

</widget>

<widget class="QSplitter" name="splitter">

<property name="geometry">

<rect>

<x>490</x>

<y>10</y>

<width>221</width>

<height>29</height>

</rect>

</property>

<property name="orientation">

<enum>Qt::Horizontal</enum>

</property>

<widget class="QLineEdit" name="lineEdit">

<property name="maximumSize">

<size>

<width>128</width>

<height>16777215</height>

</size>

</property>

</widget>

<widget class="QPushButton" name="AddButton">

<property name="sizePolicy">

<sizepolicy hsizetype="Preferred" vsizetype="Preferred">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="maximumSize">

<size>

<width>88</width>

<height>16777215</height>

</size>

</property>

<property name="font">

<font>

<weight>87</weight>

<bold>true</bold>

<underline>false</underline>

<strikeout>false</strikeout>

</font>

</property>

<property name="styleSheet">

<string notr="true">QPushButton

{

font-weight: 700;

color: white;

text-decoration: none;

padding: .8em 1em calc(.8em + 3px);

border-radius: 3px;

background: rgb(64,199,129);

box-shadow: 0 -3px rgb(53,167,110) inset;

transition: 0.2s;

}

QPushButton:pressed

{

background: rgb(33,147,90);

box-shadow: 0 3px rgb(33,147,90) inset;

}

QPushButton:hover

{

background: rgb(53, 167, 110);

}

QPushButton:focus {

/\*background-color: red;\*/

}</string>

</property>

<property name="text">

<string>Додати</string>

</property>

</widget>

</widget>

</widget>

<widget class="QMenuBar" name="menubar">

<property name="geometry">

<rect>

<x>0</x>

<y>0</y>

<width>720</width>

<height>21</height>

</rect>

</property>

</widget>

<widget class="QStatusBar" name="statusbar"/>

</widget>

<resources>

<include location="resource.qrc"/>

</resources>

<connections/>

</ui>