Міністерство освіти та науки України

Запорізький національний технічний університет

кафедра програмних засобів

з/к № //ХХХХХХХХХХХ//

**Пояснювальна записка**

з дисципліни «Об’єктно-орієнтоване програмування»

до курсової роботи на тему

//«Створення візуального інтерфейсу»//

Виконав

ст. гр. КНТ-415 Д. А. Рязанов

ст. гр. КНТ-415 О. В. Сергеев

ст. гр. КНТ-415 О. А. Бірюков

Прийняв

професор Г.В. Табунщик

доцент Т.І. Каплієнко

ст.. викл. Н.О. Миронова

Запоріжжя

2016

# ЗМІСТ

# Завдання на курсову роботу

Запорізький національний технічний університет

Кафедра *програмних засобів*

Дисципліна *Об’єктно-орієнтоване програмування*

Спеціальність *Програмне забезпечення автоматизованих систем*

Курс *2* Група *КНТ-415* Семестр *III*

**ЗАВДАННЯ**

**на курсову роботу студентам**

*Рязанову Данилу Андрійовичу*

*Сергєєву Олександру Володимировичу*

*Бірюкову Олександру Андрійовичу*

1. Тема проекту (роботи): *Система бронювання квитків театру*

2. Термін здачі студентом закінченого проекту (роботи):

3. Вихідні дані до проекту:  *Реалізувати систему бронювання квитків театру:*

*вхідні дані: інформація про вистави (назва, дата, час, ціна за кожний тип місця), інформація про місця(назва вистави, дата, час, тип місця, ряд, місце, стан(куплене/броньоване))*

*вихідні дані: програма повинна забезпечувати створення нових вистав, редагування існуючих вистав, видалення вистав, купівля, бронювання, повернення місць, можливість налаштування залу*

4. ЗМІСТ розрахунково-пояснювальної записки (перелік питань, що їх належить розробити*): 1 Аналіз предметної області*

*2 Аналіз програмних засобів*

*3 Основні рішення з реалізації компонентів системи*

*4 Посібник програміста*

*5 Інструкція користувача*

*Висновки*

*Додаток А Текст програми*

*Додаток Б Інтерфейс програми*

*Додаток В Результати роботи системи*

6. Дата видачі завдання: *09.09.2016*

# Перелік умовних позначень, символів, одиниць, скорочень і термінів

МП – мова програмування;

ПП – програмний продукт;

ПЗ – програмне забезпечення;

БД – база даних;

СКБД – система керування базами даних;

Qt – кроссплатформена бібліотека розробки GUI на C++;

MinGW – Minimalist GNU for Windows;

STL (Standard Template Library) – бібліотека для C++, що містить набір узгоджених узагальнених алгоритмів, контейнерів, засобів доступу до їхнього вмісту і різних допоміжних функцій;

JSON (JavaScript Object Notation) – текстовий формат обміну даними, заснований на JavaScript;

SQLite (Structured Query Language Lite) – [реляційна система керування базами даних](https://uk.wikipedia.org/wiki/%D0%A0%D0%B5%D0%BB%D1%8F%D1%86%D1%96%D0%B9%D0%BD%D0%B0_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B1%D0%B0%D0%B7%D0%B0%D0%BC%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85).

# Вступ

Програма є аналогом UCS-Премьери – сервісу, що являє собою онлайн касу для придбання квитків. Призначена для зручного придбання квитків з любого місця де є інтернет.

Метою роботи є здобуття навичок створення програми за допомогою засобів ООП.

Задачею є розробка програми для продажу квитків в театр. Під час розробки необхідно реалізувати ергономічний інтерфейс, ефективні алгоритми, для роботи з програмою та реалізувати роботу з даними в форматах SQLite і JSON.

В процесі реалізації даної задачі використати основні властивості об’єктно-орієнтованого програмування. В якості інструментів для розробки було обрано мову програмування C++ в середовищі Qt. В процесі розробки необхідно виконати наступні завдання:

- провести аналіз предметної області;

- розробити відповідні структури даних;

- створити візуальний інтерфейс;

- розробити програму;

- провести тестування;

- оформити пояснювальну записку.

# 1 Опис предметної області

## 1.1 Огляд існуючих методів вирішення завдання

У нашому сучасному світі існує багато театрів, покупка квитків в яких займає багато часу через отсутсвия відповідного прогрмманого забезпечення. Вибір даної предметної області обумовлений тим що для більшості театрів швидкість реалізації бронювання та продажу квитків є першорядним критерієм як обслуговування

У цю предметну область входять: театр, постановка, місце, оператор

## 1.2 Огляд існуючих програмних засобів, що вирішують аналогічні завдання

Ticketland (ticketland.ru), розробник - ТОВ «МДТЗК». Переваги: робить персоналізовані рекомендації користувачам, орієнтуючись на їхні інтереси, надаються спеціальні бонуси і подарунки. Недоліки: не дуже зручний інтерфейс, відсутність сортування по типу місць. Вікно виконання цієї програми представлене на рисунку 1.1.

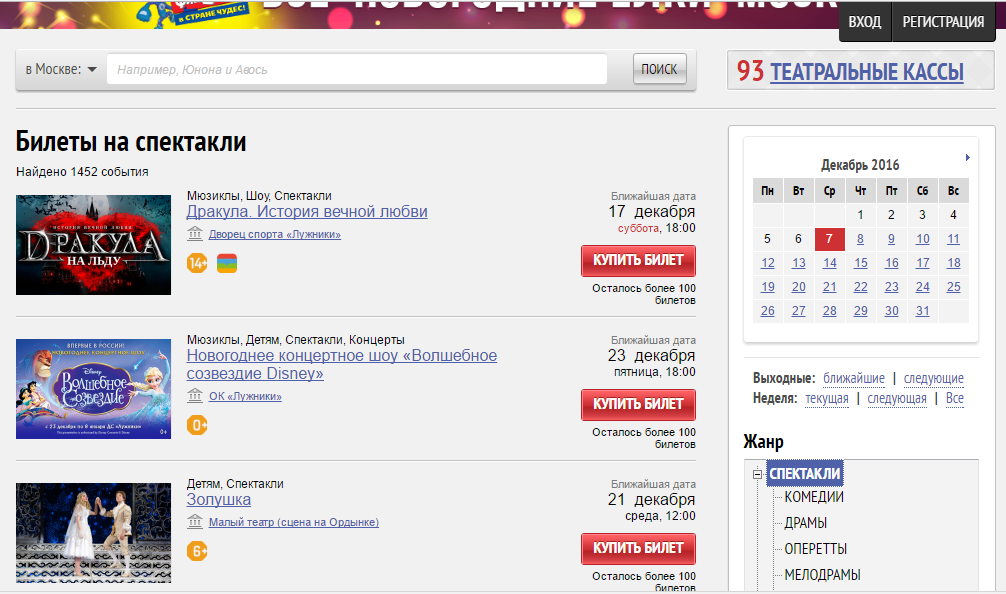


Рисунок 1.1 – Вікно виконання програми Ticketland

UCS-Премьера, розробники – UCS Original Software. Переваги: зручний інтерфейс, мультіплатформеність. Недоліки: відсутність багатомовності, скудні тона. Вікно виконання цієї програми представлене на рисунку 1.2.

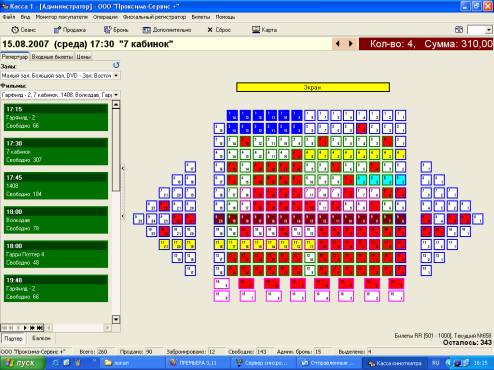


Рисунок 1.2 – Вікно виконання програми UCS-Премьера

Multiplex (multiplex.ua) розробники - INVOCO. Переваги: Переваги: кроссплатформенность, зручна навігація по сайту. Недоліки: . Вікно виконання цієї програми представлене на рисунку 1.3.

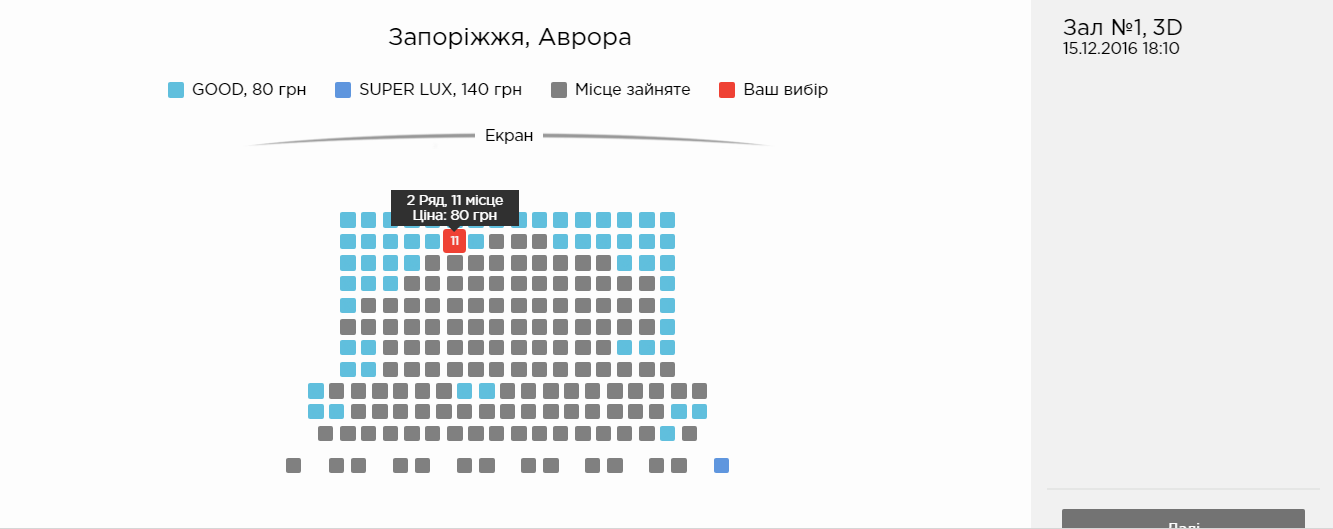


Рисунок 1.3 – Вікно виконання програми Multiplex

Kassir (kassir.ru), розробники - ТОВ «Кассир». Переваги: сортування за датою. Недоліки: незручний інтерфейс відображення місць. Вікно виконання цієї програми представлене на рисунку 1.4.



Рисунок 1.4 – Вікно виконання програми Kassir

Bileter (bileter.ru), розробники – ТОВ «Билетёр». Переваги: наявність пошуку театру по мапі. Недоліки: не ергономічний інтерфейс . Вікно виконання цієї програми представлене на рисунку 1.5.

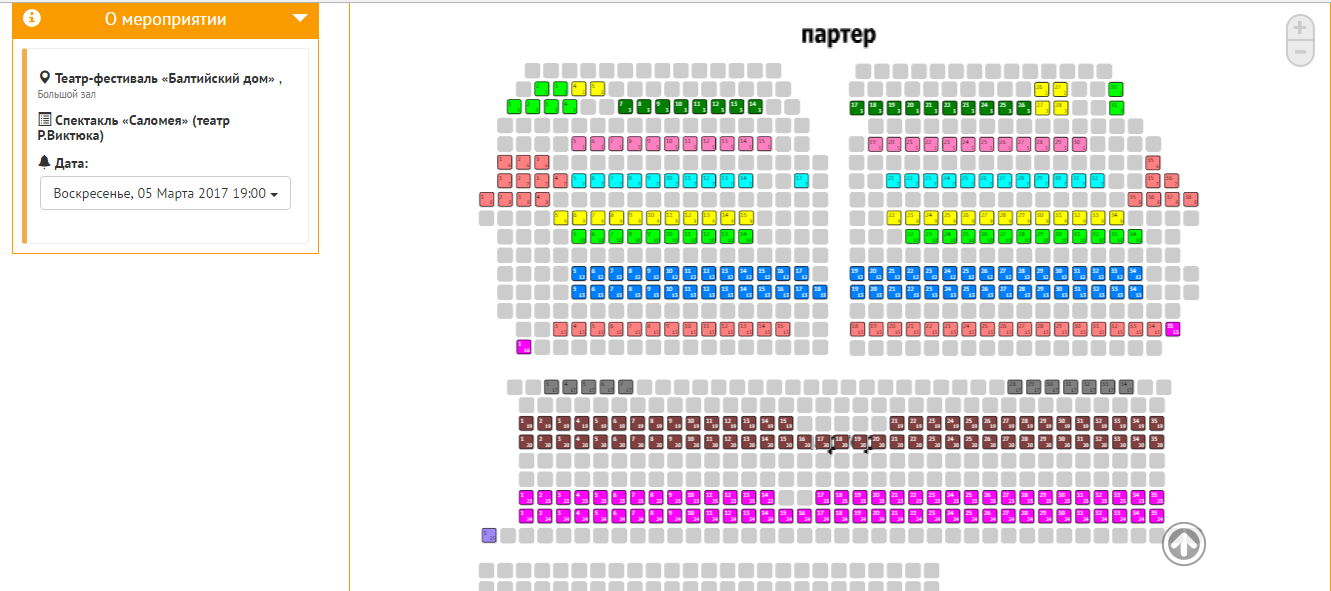


Рисунок 1.5 – Вікно виконання програми Bileter

## 1.3 Постановка завдання роботи

Створити програмних продукт основною задачею якого є купівля і бронювання квитків на вистави в театр. Він дозволяє оператору заощадити час на обслуговування покупця.

Система повинна забезпечувати:

1. бронювання та купівля квитківж

2. зручний доступ до даних кожного спектаклю;

3. інструмент добавлення нової вистави;

4. редагування та видалення вистави;

5. сортування записів за датою;

6. надання актуальної статистики;

7. налаштування розмірів залу.

# 2 Аналіз програмних засобів

## 2.1 Огляд особливостей мови програмування

Для написання програми було обрано мову програмування С++. Мова С++ - це мова програмування загального призначення, мета якого – зробити роботу програміста більш зручною. Вона надає програмісту інструмент для опису, що підлягають виконанню дій і набір концепцій, якими оперую програміст, обдумуючи, що можна зробити.

С++ підтримує такі парадигми програмування, як процедурне програмування, об'єктно-орієнтоване програмування, узагальнене програмування, забезпечує модульність, роздільну компіляцію, обробку винятків, абстракцію даних, оголошення типів (класів) об'єктів, віртуальні функції. Стандартна бібліотека включає, в тому числі, загальновживані контейнери і алгоритми.

C ++ поєднує властивості як високорівневих, так і низькорівневих мов. Найбільшу увагу приділено підтримці об'єктно-орієнтованого і узагальненого програмування.

Для розробки програми було обрано середовище Qt. Qt - кросплатформовий інструментарій розробки ПЗ мовою програмування C++. Дозволяє запускати написане за його допомогою ПЗ на більшості сучасних операційних систем шляхом простої компіляції тексту програми для кожної ОС без зміни початкового коду. Включає всі основні класи, які можуть бути потрібні при розробці прикладного програмного забезпечення, починаючи з елементів графічного інтерфейсу і закінчуючи класами для роботи з мережею, базами даних, OpenGL, SVG і XML. Бібліотека дозволяє керувати потоками, працювати з мережею, і забезпечує кросплатформовий доступ до файлів.

## 2.2 Огляд особливостей обраного компілятора

У ході розробки використовувався компілятор MinGW версії 4.1.9, що є безкоштовним компілятором та представляє собою прямий програмний порт GNU Compiler Collection (GCC) під ОС Windows. Даний компілятор дозволяє розробляти рідні (native) додатки Windows. Увесь процес компіляції такий: препроцессінг, трансляція, лінкування. В результаті отримуємо готовий бінарний файл. Більшість мов, які підтримує GCC, також підтримуються і у MinGW. Це C, C++, Objective-C, Fortran, Ada.

MinGW за замовчуванням виконує компонування з бібліотекою компонентів MSVCRT, яка є бібліотекою мови C, створеної у 1998 році. Проект підтримує і поширює кілька різних ключових компонентів і додаткових пакетів, включаючи різні порти GNU toolchain, такі як GCC і Binutils, переведені в еквіваленті пакети. Ці утиліти можуть бути використані з командного рядка вікна або інтегровані в IDE. В доповнення, компонент MinGW, відомий як MSYS (Minimal SYStem) надає win32-порти легкої Unix-подібної shell оболонки, що включає rxvt і набір інструментів POSIX, достатній для запуску скриптів Autoconf. Реалізації заголовних файлів Win32 і бібліотек імпорту Win32 для зв'язування під час виконання програми від початку до її завершення мають пермісивну ліцензію, а порти GNU доступні під GNU General Public License. Бінарні збірки повного пакету MSYS та окремих MinGW GNU утиліт доступні для завантаження на сайті MinGW.

MinGW використовує виклики Win32 API. Внаслідок цього, він не може скомпілювати деякі програми Unix, що вимагають підтримку POSIX API. Зокрема, це відноситься до додатків, які вимагають такої функціональності POSIX, як fork(), mmap() або ioctl() і допускають запуск в середовищі POSIX.

У зв'язку з тим, що в рамках початкового проекту MinGW не обіцяв, що в його кодову базу будуть вноситися оновлення пов'язані з додаванням деяких нових ключових елементів Win32 API, а також найбільш необхідної підтримки 64-бітної архітектури, був створений проект MinGW-w64 (включений до останніх версій проекту MinGW). Він є новою чистою реалізацією порту GNU Compiler Collection (GCC) під Microsoft Windows здійсненої спочатку компанією OneVision та переданої у 2008 році в громадське користування (Public Domain).

MinGW-w64 забезпечує більш повну реалізацію Win32 API та включає:

* кращу підтримку стандарту C99;
* кращу підтримку Pthreads (включаючи можливість залучити функціонал стандарту C++11 бібліотеки libstdc++ компілятора GCC);
* GCC multilib;
* точки входу в програму з підтримкою Unicode (wmain / wWinMain);
* DDK (з проекту ReactOS);
* DirectX (з проекту WINE);
* підтримка великих файлів;
* підтримка 64-розрядних ОС Windows.

## 2.3 Огляд шаблонного класу бібліотеки STL, що використовується в роботі

В програмі використано шаблони класів QVector та Vector. Клас QVector - це шаблоний клас в середовищі Qt, що представляє динамічний масив.

QVector є одним із загальних контейнерних класів Qt. Він зберігає свої елементи в суміжних комірках пам’яті та забезпечує швидкий доступ за індексом. QVector рекомендовано до використання, коли елементи є більшими, за розміром, ніж вказівник, для того щоб уникнути накладних витрат на індивідуальне виділення для них місця в "купі", під час вставки нового елемента.

QVector надає основні функції для додавання, переміщення та видалення елементів: insert(), replace(), remove(), prepend(), append(). Звернення за індексом відбувається за допомогою функцій at(), value() та перевантеженого оператора [].

В програмі, що розробляється, необхідний швидкий доступ до елементу контейнера за індексом. В контейнер поміщується обьект створенного классу, який суттєво більший за розміром, ніж вказівник. Саме через це, в якості контейнера було обрано шаблоний клас QVector.

Клас vector розташований в заголовку <vector>. Як і всі стандартні компоненти, він розташований в просторі імен std. Даний інтерфейс емулює роботу стандартного масиву C (наприклад, швидкий довільний доступ до елементів), а також деякі додаткові можливості, начебто автоматичного зміни розміру вектора при вставці або видаленні елементів.

Всі елементи вектора повинні належати одному типу. Наприклад, не можна спільно зберігати дані типів char і int в одному екземплярі вектора. Клас vector має стандартний набір методів для доступу до елементів, додавання і видалення елементів, а також отримання кількості збережених елементів.

## 2.4 Класи візуального інтерфейсу

Особливість Qt — використання Meta Object Compiller (MOC) — попередньої системи обробки початкового коду. MOC дозволяє в багато разів збільшити потужність бібліотек, вводячи такі поняття, як слоти (slots) і сигнали (signals). Qt комплектується графічним середовищем розробки графічного інтерфейсу QTDesigner, що дозволяює створювати діалоги і форми. Ідеологія створення форм у Qt базується на використанні менеджерів розташування, котрі надають «гумовий» дизайн, при якому розмір і розташування елементів форм визначаються автоматично, що значно прискорює розробку графічного інтерфейсу. В поставці Qt є «Qt Linguist» — могутня графічна утиліта, що дозволяє спростити локалізацію й переклад програми багатьма мовами, та «Qt Assistant» — довідкова система Qt, що спрощує роботу з документацією для бібліотек і дозволяє створювати крос-платформову довідку для ПЗ, розробленого на основі Qt.

Графічна бібліотека Qt розділена на декілька модулів:

QtDesigner — класи створення розширень QtDesigner'а для своїх власних віджетів;

QtGui — компоненти графічного інтерфейсу;

QtWidgets — містить класи для класичних застосувань на основі віджетів, модуль виділений з QtGui в Qt 5.

Ці модулі були використанні для створення інтерфейсу програми.

Інтерфейс розроблено за макетом «Центральна сцена». Тобто візуально в центрі робочого вікна програми будуть розміщені основні функціональні елементи інтерфейсу користувача.

Інтерфейс побудовано на основі Головного вікна програми з використанням додаткових діалогових вікон за необхідності. На Головному вікні розміщені: Панель меню, Рядок стану, Панель інструментів, різні варіанти списків, поля вводу, кнопки, таблиці. Всі елементі доповнені відповідними підписами. Є підтримка комбінацій гарячих клавіш та табуляції.

Додаткові діалогові вікна можуть бути функціональними або вікнами-повідомленнями з відповідними їм графічними елементами.

## 2.6 Висновки з розділу 2

Мовою програмування було обрано C++ з Qt Framework за зручність створення GUI, потужність та гнучкість мови, широкий спектр можливостей.

В якості компілятора було обрано MinGW 4.8.1, який є доступним, зручним, налаштованим під ОС Windows.

Інтерфейс було розроблено засобами Qt Designer за макетом «Центральна сцена».

**3 Основні рішення з реалізації компонентів системи**

**3.1 Основні рішення щодо уявлення даних системи**

Дані розробленої програми можна абстрактно розділити на 4 групи: дані для заповнення місць, дані, які використовуються для представлення інформації про вистави, дані, які використовуються для відображення статистики по придбаним місцям, та дані, які відповідають за налаштування залу.

Дані для заповнення місць є одними з основних для роботи з алгоритмами програми, дані зберігаються в БД та представленні у формі таблиці яка зберігає в собі всю необхідну інформацію для зручного та швидкого відображення даних в інтерфейсі програми. Вона складається з наступних полів: поле з кодом (використовується для швидкого знаходження броньованих місць), поле з назвою вистави, поле з часом вистави, поле з датою вистави, поле яке відповідає за тип місця, поле яке відповідає за статус місця (куплене або броньоване), а також основні поля які зберігають в собі значення ряду та місця в залі.

Дані, які використовуються для представлення інформації про вистави, зберігаються також у БД. Всі дані з цієї таблиці потрапляють в клас Scene, який за допомогою методів достає значення та виводить їх на панель вистав. Таблиця з виставами зберігає наступні поля: назва, дата, час, та ціна за кожний тип місць на цю виставу.

Дані які відповідають за відображення статистики проданих місць за певний проміжок часу. Зберігаються в БД, та за допомогою методів класу DataforStatistic потрапляють на графік у вікні статистики. Таблиця зберігає поля з датою, типом місця та лічильником проданих місць.

З усіма даними які є у нас в програмі ми працюємо через класи, які не прив’язані до інтерфейсу. Дані для налаштувань залу зберігаються в файлі формату JSON, Зберігає 3 об’єкти, в кожному з яких зберігається інформація про кількість рядів та місць в ряді для конкретного типу місця. Для роботи з даними ми використовуємо СКБД SQLite.

**3.2 Основні розроблені алгоритми**

Для функціонування програми було реалізовано алгоритми додавання нової вистави, видалення та редагування існуючої вистави, пошуку існуючих вистав на певний день, бронювання місця разом з генерацією коду, купівлі повернення вибраних місць, зміни налаштувань залу, пошуку місць за даним кодом бронювання, а також будування графіків.

При додаванні нової вистави в першу чергу програма перевіряє наявність вже існуючої вистави на цей день, час та з тією ж назвою. Якщо така вистава відсутня, то програма заносить у БД дані про нову.

Після зміни поточної дати програма здійснює пошук вистав які відповідають поточній даті, та виводить їх у таблицю вистав. Після кліку по назві вистави відкриється контекстне меню з двома пунктами: «Редагувати» та «Видалити». Якщо кристувач обрав перший пункт, то відкриється вікно редагування, в якому вже будуть завантажені дані цієї вистави з БД. Користувач може їх змінити та зберегти, після чого БД оновить дані. Якщо ж він обере пункт «Видалити», то програма видалить усі дані що були пов'язані з цією виставою з БД.

Якщо користувач вибере виставу, то з БД завантажаться дані про неї і оновиться таблиця місць, де кожен рядок таблиці відповідає ряду, а кожен стопвець ­­– місцю у ряді. Якщо натиснути на будь-яку клітинку, то вона стане виділеною.

При натисканні на кнопку «Бронювати» усі виділені місця, якщо вони не є вже заброньованими або купленими, стають заброньваними і генерується спеціальний код. Якщо його ввести у відповідне поле, та натиснути кнопку «Пошук», програма сама знайде ті місця, яким відповідає даний код та зробить їх виділеними.

При натискані на кнопку «Купити» дані про усі виділені місця, якщо вони не є вже купленими, заносяться до таблиці діалогового вікна, де користувач перевіряє що вибрав саме ті місця, що йому потрібні, та купує їх.

При натискані на кнопку «Повернути» дані про усі місця, які є заброньованими або купленими видаляються з БД і місце стає вільним.

При натискані на пункт «Налаштування залу» відкріється нове вікно, де користувач може виставити нову кількість рядів та місць у ньому для кожного типу місць.

Дані про усі продані білети заносяться у відповудну таблицю БД. Коли користувач відкріває вікно статистики то програма завантажує їх і будує графік, який відображає число білетів, які було продано у певний день для кожного з типів місць.

**3.3 Основні рішення щодо модульного уявлення системи**

Програма складається з таких файлів:

Заголовні файли:

**About.h** містить оголошення класу About, який успадковується від класу QDialog. Клас містить макрос Q\_OBJECT.

Поле **private** містить наступні дані:

Ui::About \*ui – вказівник на об’єкт форми About;

Поле **public** містить наступні дані:

explicit About(QWidget \*parent = 0) – конструктор класу About;

~About() – деструктор класу About;

**AddScene.h** містить оголошення класу AddScene який успадковується від класу QDialog. Клас містить макрос Q\_OBJECT.

Поле **private** містить наступні дані:

Ui::AddScene \*ui – вказівник на об’єкт форми AddScene;

QSqlDatabase mydb – змінна типу QSqlDatabase;

Поле **public** містить наступні дані:

explicit AddScene(QWidget \*parent = 0) – конструктор класу AddScene;

explicit AddScene(bool edit, Scene \*temp, QWidget \*parent = 0) – конструкторкласу ddScene;

~AddScene() – деструктор класу AddScene;

bool Editing – змінна типу bool;

QString PriviosName

PriviosTime,

PriviosDate; – змінні типу QString,

Поле **private slots** містить наступні дані:

void on\_ButtonAdd\_clicked() – метод типу void;

**MainWindow.h** містить оголошення класу MainWindow, який успадковується від класу QDialog. Клас містить макрос Q\_OBJECT.

Поле **private** містить наступні дані:

Ui::MainWindow \*ui – вказівник на об’єкт форми MainWindow;

QSqlDatabase mydb –змінна типу QSqlDatabase;

void places\_fill() – метод типу void;

void cleasing\_places() – метод типу void;

void create\_a\_MainTable() – метод типу void;

void places\_overwrite(int row, int column) – метод типу void;

void pix\_close(int row, int column) – метод типу void;

void pix\_checking(int row, int column) – метод типу void;

void pix\_reserv(int row, int column) – метод типу void;

void pix\_standart(int row, int column) – метод типу void;

void customizeTableInf() – метод типу void;

Поле **public** містить наступні дані:

explicit MainWindow(QWidget \*parent = 0) – конструктор класу MainWindow;

~*MainWindow*() – деструктор класу MainWindow;

Scene \*CurScene – вказівник на клас Scene;

QDateTime dt – змінна типу QDateTime;

int row\_height – змінна типу int;

int column\_width – змінна типу int;

bool \*\*coordinates\_of\_places – вказівник на вказівник типу bool;

void coordinates\_of\_places\_cleaning(int temp) – метод типу void;

std::vector <short> SelectedPlacesRow – змінна типу vector <short>;

std::vector <short> SelectedPlacesCol – змінна типу vector <short>;

int PreviousIndex – змінна типу int;

int CountRow – змінна типу int;

int CountColumn – змінна типу int;

int CountPurchased – змінна типу int;

int CountBooked – змінна типу int;

Поле **private slots** містить наступні дані:

void on\_informationAbout\_triggered() – метод типу void;

void on\_action\_triggered() – метод типу void;

void on\_tableWidget\_cellClicked(int row, int column) – метод типу void;

void on\_comboBox\_currentIndexChanged(int index) – метод типу void;

void on\_tableSeans\_cellClicked(int row, int column) – метод типу void;

void on\_dateEdit\_dateChanged(const QDate &date) – метод типу void;

void on\_pushButton\_clicked() – метод типу void;

void on\_pushButton\_2\_clicked() – метод типу void;

void on\_action\_exit\_triggered() – метод типу void;

void on\_action\_statistic\_sale\_triggered() – метод типу void;

void on\_pushButton\_3\_clicked() – метод типу void;

void on\_options\_room\_triggered() – метод типу void;

void on\_action\_addScene\_triggered() – метод типу void;

void on\_action\_hovered() – метод типу void;

void on\_tableSeans\_customContextMenuRequested(const QPoint &pos) – метод типу void;

void slotEditRecord() – метод типу void;

**OptionsForHall.h** містить оголошення класу OptionsForHall, який успадковується від класу QDialog. Клас містить макрос Q\_OBJECT.

Поле **private** містить наступні дані:

void parseJSON(QJsonDocument& jsonDoc) – метод типу void;

Ui::OptionsForHall \*ui – вказівник на об’єкт форми OptionsForHall;

QSqlDatabase mydb – змінна типу QSqlDatabase;

void DrawTable() – змінна типу void;

Поле **public** містить наступні дані:

explicit OptionsForHall(QWidget \*parent = 0) – конструктор класу OptionsForHall;

~*OptionsForHall*() – деструктор класу OptionsForHall;

double dataJsonFile[3][2] – змінна типу double;

QString type\_placeJsonFile[3] – змінна типу QString;

Поле **private slots** містить наступні дані:

void on\_spinBox\_Row\_valueChanged(int arg1) – метод типу void;

void on\_spinBox\_Column\_valueChanged(int arg1) – метод типу void;

void on\_ComboBoxType\_currentIndexChanged(int index) – метод типу void;

void on\_pushButtonOK\_clicked() – метод типу void;

**Scene.h** містить оголошення класу Scene, який успадковується від класу QDialog. Клас містить макрос Q\_OBJECT.

Поле **public** містить наступні дані:

Scene() – конструктор класу Scene;

void set\_name(QString arg) – метод типу void;

void set\_date(QString arg) – метод типу void;

void set\_time(QString arg) – метод типу void;

void set\_cost() – метод типу void;

void SetArrayCountPlaces() – метод типу void;

void SetDataToTables() – метод типу void;

void DeleteTables() – метод типу void;

void InsertTablesToDataBase(std::vector<short> SelectedPlacesRow, std::vector<short> SelectedPlacesCol, int Index, int operation) – метод типу void;

int ArrayCountPlaces[3][2] – змінна типу int;

int \*\*\*TablesPlaces – вказівник на вказівник на вказівник типу int;

int CountOfTypesPlaces – зміна типу int;

QString name, date, time –змінна типу QString;

double cost\_parter, cost\_benuar, cost\_beletaj – змінні типу double;

QString nameofplace[3] – змінна типу QString;

**Statistic.h** містить оголошення класу Statistic, який успадковується від класу QDialog. Клас містить макрос Q\_OBJECT.

Поле **private** містить наступні дані:

Ui::Statistic \*ui – вказівник на об’єкт форми Statistic;

Поле **public** містить наступні дані:

explicit Statistic(QDateTime datenow, QWidget \*parent = 0) – конструктор класу Statistic;

~*Statistic*() – деструктор класу Statistic;

Файли реалізації:

Statistic.cpp – містить реалізацію методів класу Statistic;

Scene.cpp – містить реалізацію методів класу;

OptionsForHall.cpp – містить реалізацію методів класу Options;

MainWindow.cpp – містить реалізацію методів класу MainWindow;

main.cpp – являє собою точку запуску програми;

FinalyWindow.cpp – містить реалізацію методів класу FinalyWindow;

DataForStatistic.cpp – містить реалізацію методів класу DataForStatistic;

AddScene.cpp – містить реалізацію методів класу AddScene;

About.cpp – містить реалізацію методів класу About.

**3.4 Особливості реалізації системи**

**3.5 Результати тестування системи**

Під час тестування програми, знайдені недоліки було виправлено. Під час фінального тестування недоліків виявлено не було. Програма функціонує правильно, та стабільно.

**3.6 Висновки з розділу 3**

В результаті виконання курсової роботи було розроблено комплекс класів з використанням засобів Qt, який повністю реалізує поставлену задачу. Були використані основні методи об’єктно - орієнтованого програмування.

Було проведено тестування програми. Програма працює коректно та містить весь, запланований до реалізалізації, функціонал.

# 4 посібник програміста

## 4.1 Призначення й умови застосування програми

Даний програмний засіб представляє собою програмне забезпечення для ефективної роботи касира в театрі

В програмі реалізовані такі основні функції як: додавання вистави, редагування вистави, придбання, бронювання та повернення квитів, генерація унікального коду, пошук за унікальним кодом, видалення вистав з БД якщо за датою вони застаріли, налаштування кількості міст в залі а також відображення актуальної статистики.

Далі представлені основні вимоги до системи, необхідні для стабільної роботи і функціональності VTeatre. В характеристиках вказані мінімальні параметри.

Системні вимоги:

1. процесор з тактовою частотою 800 MHz;
2. 128 Мб та більше DDR2 RAM;
3. вільне місце на ПЗУ: 40 Мб;
4. архітектура системи з розрядністю 32- або 64-біт;
5. операційна система: Windows (з версії ХР та вище);
6. стандартні пристрої вводу-виводу (клавіатура, миша, монітор);

## 4.2 Характеристика програми

Часові рамки коректної роботи програми обмежені 2026 роком, це пов’язано з реалізацією системного календаря в більшості операційних систем.

В програмі реалізовано єдиний режим роботи – в якості оператора.

Контроль правильності виконання і самовідновлення програми реалізовано за допомогою обробки виключних ситуацій.

## 4.3 Звертання до програми

Виклик програми проводиться за допомогою стандартних засобів операційної системи з exe файлу.

Внесення даних у програми може відбуватися як в ручному режимі, так і з SQLite чи JSON файлу.

## 4.4 Початкові та вихідні дані

Початкові дані містять інформацію про вистави, придбані, броньовані місця, значення для налаштувань залу, значення для відображення статистики.

Вихідні дані полягають у забезпеченні можливості створення нових вистав, редагування існуючих вистав, видалення вистав, також забезпечують купівлю, бронювання та повернення місця, змін налаштувань залу.

Початкові та вихідні дані можуть бути завантажені та збережені в XML та JSON файлах.

## 4.5 Повідомлення

Повідомлення:

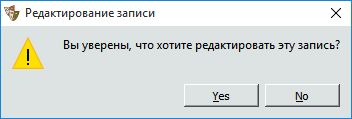


Рисунок 4.1 – Повідомлення при натисканні редагування вистави

Це повідомлення виникає перед користувачем коли він намагається редагувати вистави. Реагувати на це повідомлення потрібно залежно від ваших намірів

Попередження:

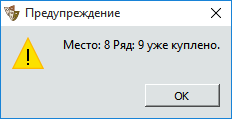


Рисунок 4.2 – Попереджує про змогу купити куплене місце

Це попередження з’являється перед користувачем у випадках, коли оператор хоче придбати місце яке вже було до цього придбано. Ця помилка не додасть дане місце в фінальне вікно оплати.

Повідомлення про помилку:

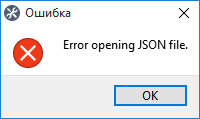


Рисунок 4.3 – Повідомлення про помилку відкриття файлу за даними

За випадків коли файл з даними відсутній в кореневому каталозі, виникає така помилка.

# 5 Інструкція користувача

## 5.1 Призначення та функції програми

Програмний засіб призначений для оператора театру. Він має на меті допомогти оператору прискорити роботу по бронюванню та продажі квитків на вистави.

Використовувати VTeatre може абсолютно будь-яка людина з мінімальними навичками володіння комп’ютером, а ергономічний інтерфейс спрощує пристосування до нового програмного засобу. Програма має такі функції:

1. бронювання та придбання квитка на будь-яку виставу театру;
2. налаштування залу, а саме: кількість рядів та місць;
3. редагування, додавання, видалення будь-якої вистави;
4. відображення статистики по продажам квитків.

## 5.2 Умови виконання програми

Далі представлені основні вимоги до системи, необхідні для стабільної роботи і функціональності VTeatre. В характеристиках вказані мінімальні параметри.

Системні вимоги:

1. процесор з тактовою частотою 800 MHz;
2. 128 Мб та більше DDR2 RAM;
3. вільне місце на ПЗУ: 40 Мб;
4. архітектура системи з розрядністю 32- або 64-біт;
5. операційна система: Windows (з версії ХР та вище);
6. стандартні пристрої вводу-виводу (клавіатура, миша, монітор);

## 5.3 Виконання програми

Для запуску програми необхідно виконати наступні кроки:

1. упевнитися у відповідності характеристик ПК, на яку встановлюється програма, до мінімальних системних вимог програми;
2. упевнитися у справності всіх компонентів ПК та електронного носія, з якого здійснюватиметься запуск програми;
3. у каталозі із завантаженою програмою знайти виконуваний файл VTeatre.exe та здійснити його запуск.

Подальша робота з програмою здійснюється за допомогою графічного інтерфейсу в режимі WYSIWYN. Для виконання певних функцій необхідно взаємодіяти з відповідними компонентами GUI. Інтерфейс головного вікна програми представлений на рисунку 5.1.

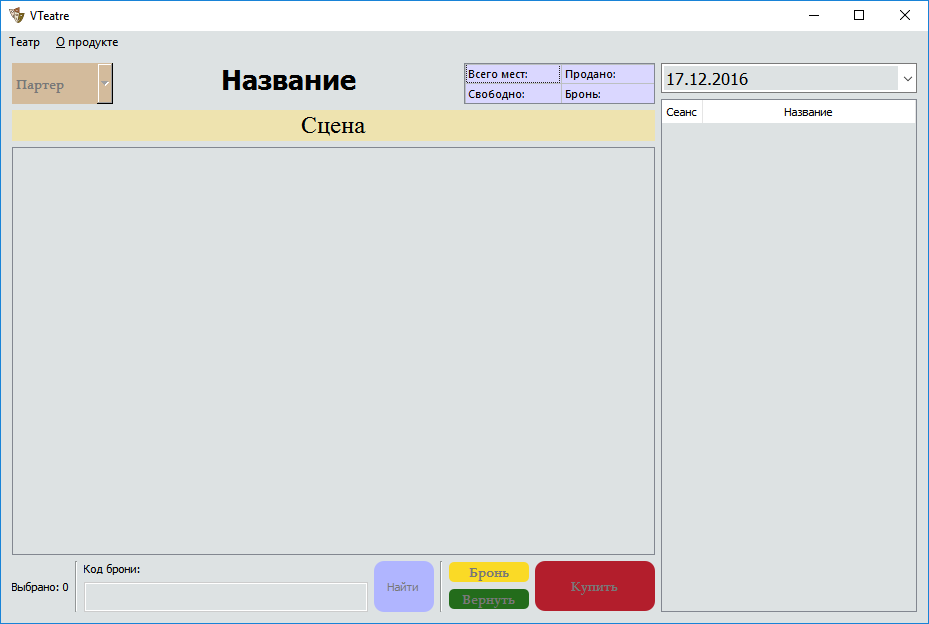


Рисунок 5.1 – Інтерфейс головного вікна програми

Для того щоб продивитися інформації о місцях вистави, треба натиснути на поле с датою, вибрати необхідну дату (рис. 5.2), натиснути на потрібну назву вистави в відповідній таблиці (рис. 5.3).

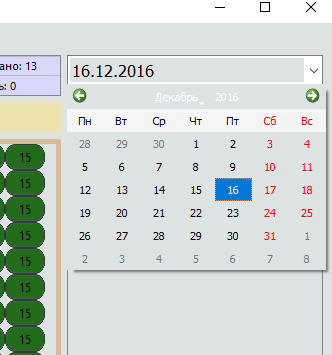


Рисунок 5.2 – Поле з датою

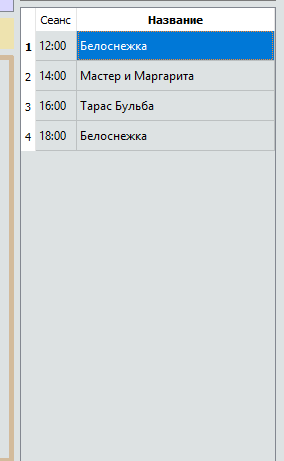


Рисунок 5.3 – Таблиця з виставами

Кожна вистава має декілька типів місць. Для того щоб продивитися інформацію по місцям якогось типу треба змінити цей тип в випадаючому списку (рис 5.4).

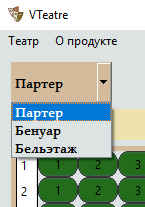


Рисунок 5.4 – Випадаючий список з типами місць

Для того щоб забронювати чи продати місце, оператор повинен один раз натиснути на відповідне місце в таблиці, місце перефарбується в блакитний колір (рис. 5.5). Щоб придбати, забронювати чи відновити вибрані місця потрібно натиснути на кнопку «Купить», «Бронь» или «Вернуть» відповідно (рис. 5.6). Кількість натиснутих місць відображається у вигляді надпису в лівій нижній частині екрану (рис. 5.7.1), основна інформація про таблицю місць відображається в верхній правій частині екрану у вигляді таблиці з 4 комірками (рис. 5.7.2): «Всего мест», «Продано», «Свободно», «Бронь».



Рисунок 5.5 – Вибір місць



Рисунок 5.6 – Функціональні кнопки



Рисунок 5.7.1 – Відображення кількості вибраних місць



Рисунок 5.7.2 – Таблиця з інформацією про таблицю місць

Якщо оператор вибрав місце та натиснув на «Купить», відкривається нове вікно для кінцевого підтвердження покупки. Вікно покупки складається з таблиці з детальною інформацією про вибране місце, надпису з загальною вартістю та функціональної кнопки (рис 5.8).

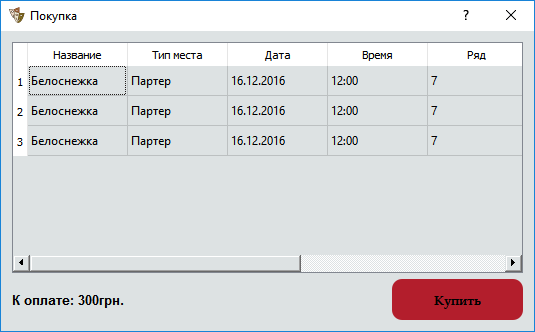


Рисунок 5.8 – Вікно покупки

Якщо оператор вибрав місце та натиснув на «Бронь», викликається вікно з кодом броні (рис. 5.9), для того щоб в майбутньому придбати заброньовані місця, покупець повинен вказати оператору цей код. Для того щоб виділити місця, які забронював покупець, оператор має можливість ввести код броні в відповідне поле та натиснути кнопку «Найти» (рис 5.10).

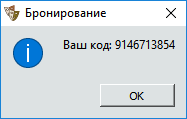


Рисунок 5.9 – Код броні

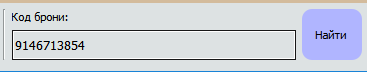


Рисунок 5.10 – Поле для вводу коду броні та функціональна кнопка

Для того щоб перейти до налаштувань залу, додати нову виставу, чи проглянути статистику, потрібно натиснути на відповідний пункт в меню «Театр» (рис. 5.11).

Вікно налаштувань складається з випадаючого списку типів місць, двох лічильників, відповідающих за кількість рядів та місць, таблиці для демонстрації зовнішнього вигляду в залежності від параметрів лічильників, кнопки «Принять» (рис. 5.12).

Кнопка «Принять» фіксує нові розміри таблиці місць. Після фіксації можна закривати вікно налаштувань.

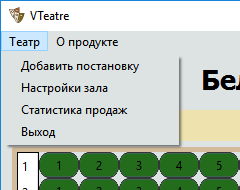


Рисунок 5.11 – Пункти меню «Театр»

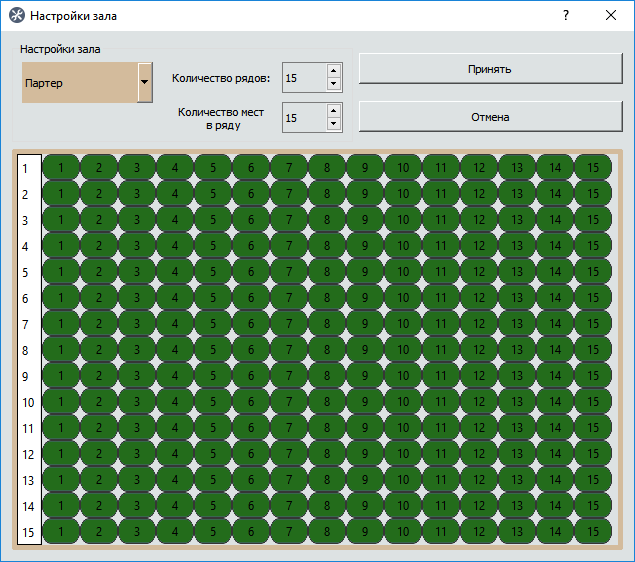


Рисунок 5.12 – Вікно налаштувань залу

Вікно додавання вистави складається з поля з назвою вистави, двох часових полів, трьох лічильників цін для кожного типу місць, кнопки «Добавить» (рис 5.13).

Щоб додати нову виставу, потрібно заповнити всі поля та натиснути кнопку «Добавить», після цього можна закрити вікно, чи продовжити додавання вистав.

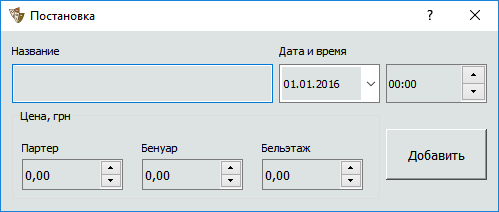


Рисунок 5.13 – Вікно додання вистав

Вікно статистики складається з трьох кнопок, кожна з якої відповідає за відображення статистики за різний проміжок часу, та координатної площини, де вісь Х відповідає за дату, а вісь Y за кількість придбаних місць (рис 5.14). На площині відображаються три графіка, кожен з яких відповідає за тип місця. Легенда відображає назви цих графіків.

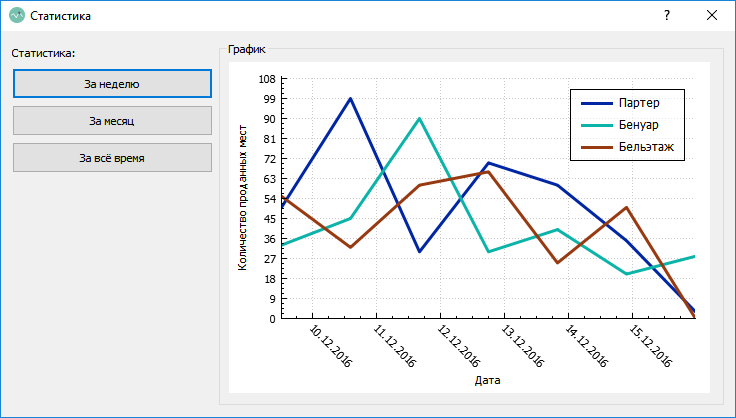


Рисунок 5.14 – Вікно статистики

При натисненні правою кнопкою на вистави, з’явиться контекстне мені, в якому можна здійснити такі операції як редагування вистави та її видалення (рис 5.15).

Якщо обрати вибрати пункт «Редактировать» то з’явиться вікно на аналог вікна створення вистави (рис 5.13) тільки з вже з заповненими даними про виставу. По завершенню редагування необхідних даних обов’язково натиснути кнопку «Редактировать» (рис 5.16) та закрити вікно

Якщо обрати пункт «Удалить» то дані про виставу назавжди зникнуть з програми і БД.

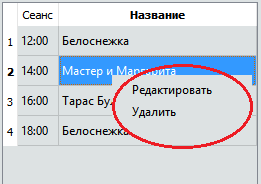


Рисунок 5.15 – Контекстне меню

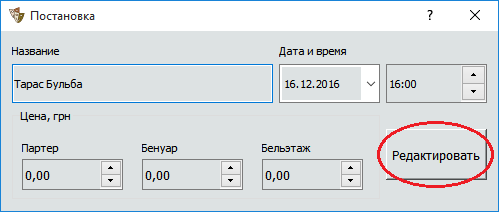


Рисунок 5.16 – Вікно редагування вистави

## 4.5 Повідомлення

Повідомлення:

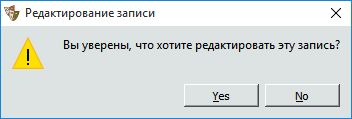


Рисунок 4.1 – Повідомлення при натисканні редагування вистави

Це повідомлення виникає перед користувачем коли він намагається редагувати вистави. Реагувати на це повідомлення потрібно залежно від ваших намірів

Попередження:

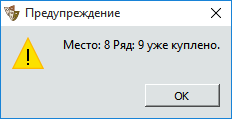


Рисунок 4.2 – Попереджує про змогу купити куплене місце

Це попередження з’являється перед користувачем у випадках, коли оператор хоче придбати місце яке вже було до цього придбано. Ця помилка не додасть дане місце в фінальне вікно оплати.

Повідомлення про помилку:

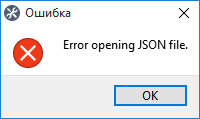


Рисунок 4.3 – Повідомлення про помилку відкриття файлу за даними

За випадків коли файл з даними відсутній в кореневому каталозі, виникає така помилка.

**Висновки**

Під час розробки курсової роботи було створено програмне забезпечення, яке виконує функції TODO list (списка справ, або списка задач). Було реалізований такий функціонал програми: створення, редагування та видалення задач, генерація нагадувань, можливість збереження та завантаження даних з файлів форматів XML і JSON.

Програма розроблена на мові програмування С++ в середовищі Qt. Під час розробки програми, були широко використані засоби Qt та ООП.

За результатами тестування програми, всі її функції працюють коректно.

Також було оформлено пояснювальну записку до курсової роботи, в якій було проведено аналіз програмної області та програмних засобів, описано основні рішення з реалізації компонентів системи, написано посібник програміста та інструкцію користувача.

**Список використаної літератури**

1. Методичні вказівки до виконання курсової роботи з дисципліни “Об’єктно-орієнтоване програмування” для студентів напряму підготовки 6.050103 «Програмна інженерія» всіх форм навчання. Частина 1. Організаційна частина / Г.В. Табунщик, Г.В. Неласа, Н.О. Миронова – Запоріжжя: ЗНТУ, 2010. – 22 с.

2. Лафоре Р. Объектно-ориентированное программирование в C++ / Роберт Лафоре. – СПБ: Питер, 2004. – 924 с.

3. Шлее М. Qt 5.3. Профессиональное программирование на C++ / Макс Шлее. – СПБ: БХВ-Петербург, 2015. – 928 с.:ил.

4. Qt Documentation [Електронний ресурс] – Режим доступу до ресурсу: <http://doc.qt.io/>.