Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет

информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра вычислительных машин, систем и сетей

Дисциплина: Программирование на языках высокого уровня

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе

на тему

“Система бронирования мест в кинотеатре”

Студент Ф. И. Рыбак

Руководитель Е.В. Богдан

МИНСК 2023

Учреждение образования

«Белорусский государственный университет информатики   
и радиоэлектроники»

Факультет компьютерных систем и сетей

УТВЕРЖДАЮ

Заведующий кафедрой ЭВМ

*––––––––––––––––––––––––*

             (подпись)

––––––––––––––––––2023   г.

ЗАДАНИЕ

по курсовому проектированию

Студенту  *Рыбак Феона Игоревна*

1.  Тема проекта  *Система бронирования мест в кинотеатре*        ––   ––––

2. Срок сдачи студентом законченного проекта   *15 декабря 2023 г.–––*

3. Исходные данные к проекту: картинки png для иконок мест в папке img, cinema.sqlite(база данных)

4. Содержание расчетно-пояснительной записки (перечень вопросов, которые подлежат разработке)

1. Введение.

2. Задание.

3. Обзор литературы.

  3.1. Обзор методов и алгоритмов решения поставленной задачи.

4.  Функциональное проектирование.

  4.1. Структура входных и выходных данных.

  4.2. Разработка диаграммы классов.

  4.3. Описание классов.

5. Разработка программных модулей.

  5.1. Разработка схем алгоритмов (два наиболее важных метода).

  5.2. Разработка алгоритмов (описание алгоритмов по шагам для двух методов).

6. Результаты работы.

7. Заключение

8. Литература

9. Приложения

5. Перечень графического материала (с точным обозначением обязательных чертежей и графиков)

*1. Диаграмма классов.  ––––––––––––––––––––––––––––––*

*2. Схема алгоритма on\_btnBook\_clicked().*

*3. Схема алгоритма on\_btnCancel\_clicked().––––––––––––––––––––––––––––––*

6. Консультант по проекту (с обозначением разделов проекта) Е.В.Богдан

7. Дата выдачи задания  *15 сентября 2023 г.––––––––––––––––––––––   –*

8. Календарный график работы над проектом на весь период проектирования (с обозначением сроков выполнения и трудоемкости отдельных этапов):

*1. Выбор задания.  Разработка содержания пояснительной записки. Перечень графического материала – 15 %; ––––––––––––––––––––––––––––\_\_\_\_\_\_\_\_\_*

*разделы 2, 3 – 10 %; –––––––––––––––––––––––––––––––––––––   \_\_\_\_\_\_\_\_\_\_*

*разделы 4 к   –20 %; –––––––––––––––––––––––––––––––––––––––\_\_\_\_\_\_\_\_\_*

*разделы 5 к – 35 %; –––––––––––––––––––––––––––––––––––––\_\_\_\_\_\_\_\_\_\_\_–*

*раздел 6,7,8   – 5 %; ––––––––––––––––––––––––––––––––––––––\_\_\_\_\_\_\_\_\_\_–*

*раздел 9   к – 5%; ––––––––––––––––––––––––––––––––––––––––– \_\_\_\_\_\_\_\_*

*оформление пояснительной записки и графического материала к 15.12.23 – 10 %*

*Защита курсового проекта с 21.12 по 28.12.23г.–––––––––––––––––––––––––*

РУКОВОДИТЕЛЬ Е. В. Богдан

(подпись)

Задание принял к исполнению Ф.И.Рыбак

(дата и подпись студента)

**СОДЕРЖАНИЕ**

[1 ЗАДАНИЕ НА КУРСОВУЮ РАБОТУ 2](file:///C:\Users\Феона\Downloads\Курсач%20(1).docx#_Toc150462830)

[2. ВВЕДЕНИЕ 6](file:///C:\Users\Феона\Downloads\Курсач%20(1).docx#_Toc150462831)

[3. ОБЗОР ЛИТЕРАТУРЫ 7](file:///C:\Users\Феона\Downloads\Курсач%20(1).docx#_Toc150462832)

[3.1 ОБЗОР МЕТОДОВ И АЛГОРИТМОВ РЕШЕНИЯ ПОСТАВЛЕННОЙ ЗАДАЧИ 8](file:///C:\Users\Феона\Downloads\Курсач%20(1).docx#_Toc150462833)

[4 ФУНКЦИОНАЛЬНОЕ ПРОЕКТИРОВНИЕ 9](file:///C:\Users\Феона\Downloads\Курсач%20(1).docx#_Toc150462834)

[5 РАЗРАБОТКА ПРОГРАММНЫХ МОДУЛЕЙ 14](file:///C:\Users\Феона\Downloads\Курсач%20(1).docx#_Toc150462835)

[5.1 Разработка схем алгоритмов 14](file:///C:\Users\Феона\Downloads\Курсач%20(1).docx#_Toc150462836)

[5.2 Разработка алгоритмов 14](file:///C:\Users\Феона\Downloads\Курсач%20(1).docx#_Toc150462837)

[6 РЕЗУЛЬТАТ РАБОТЫ 16](file:///C:\Users\Феона\Downloads\Курсач%20(1).docx#_Toc150462838)

[ЗАКЛЮЧЕНИЕ 20](file:///C:\Users\Феона\Downloads\Курсач%20(1).docx#_Toc150462839)

[СПИСОК ЛИТЕРАТУРЫ 21](file:///C:\Users\Феона\Downloads\Курсач%20(1).docx#_Toc150462840)

[ПРИЛОЖЕНИЕ А 22](file:///C:\Users\Феона\Downloads\Курсач%20(1).docx#_Toc150462841)

[ПРИЛОЖЕНИЕ Б 23](file:///C:\Users\Феона\Downloads\Курсач%20(1).docx#_Toc150462842)

[ПРИЛОЖЕНИЕ В 24](file:///C:\Users\Феона\Downloads\Курсач%20(1).docx#_Toc150462843)

[ПРИЛОЖЕНИЕ Г 25](#_Toc150462844)

**ЗАДАНИЕ НА КУРСОВУЮ РАБОТУ**

Овладеть практическими навыками проектирования и разработки законченного, отлаженного и протестированного программного продукта с использованием языка высокого уровня С++, овладеть практическими навыками проектирования и разработки законченного, отлаженного и протестированного программного продукта с использованием языка высокого уровня С++. Разработать систему бронирования мест в кинотеатре на языке программирования C++ с использованием фреймворка Qt и базы данных SQLite.

Реализовать основные функциональности системы:

- Отображение доступных мест в зале кинотеатра.

- Возможность выбора и бронирования конкретного места.

- Проверка валидности вводимых данных при бронировании (имя, фамилия).

- Отмена бронирования места.

- Обновление интерфейса после каждой операции бронирования или отмены.

- Реализовать взаимодействие между классами через сигналы и слоты.

Например, при успешном бронировании места необходимо обновить интерфейс и отобразить соответствующее сообщение об успешном бронировании.

- Обеспечить корректное сохранение и обновление данных в базе данных SQLite. Например, при бронировании места необходимо обновить статус места в базе данных на "забронировано" и добавить информацию о посетителе.

2. ВВЕДЕНИЕ

Кинотеатры являются популярными местами для проведения досуга и развлечений. Однако, для многих посетителей, особенно в пиковые периоды или на популярные сеансы, может быть сложно получить желаемое место в зале без предварительного бронирования. Для упрощения этого процесса и обеспечения удобства посетителям разработано программное решение на основе языка программирования C++ с использованием фреймворка Qt и базы данных SQLite.

Целью данной курсовой работы является создание системы бронирования мест в кинотеатре. Она предоставляет возможность посетителям выбирать места, бронировать их и осуществлять отмену брони в случае необходимости. Система также обеспечивает удобное отображение доступных мест и информацию о забронированных местах.

Для разработки данного проекта используется язык программирования C++ с использованием фреймворка Qt. Qt обладает мощными инструментами для разработки пользовательских интерфейсов, что позволяет создать интуитивно понятный и привлекательный интерфейс для посетителей кинотеатра. Кроме того, в качестве базы данных выбран SQLite, так как она является легкой и компактной, что особенно важно для проектов с небольшим объемом данных, таких как система бронирования мест.

В процессе разработки системы будут использованы различные классы и объекты, такие как MainWindow, book, cancel и MyDB, которые обеспечат функциональность бронирования, отмены брони и управления базой данных соответственно. Классы будут взаимодействовать друг с другом через сигналы и слоты, обеспечивая согласованность данных и корректное выполнение операций.

Ожидается, что разработанная система бронирования мест в кинотеатре значительно упростит процесс бронирования для посетителей и повысит уровень обслуживания. В результате выполнения курсовой работы был получен опыт работы с языком С++, фреймворком Qt и опыт создания системы бронирования мест в кинотеатре.

3. ОБЗОР ЛИТЕРАТУРЫ

**3.1****Обзор методов и алгоритмов решения поставленной задачи**

Создание пользовательского интерфейса. Для реализации пользовательского интерфейса системы бронирования мест в кинотеатре используется фреймворк Qt. Qt предоставляет богатый набор инструментов для создания графического интерфейса, включая виджеты, компоновщики, стили и темы.

Можно создать окно приложения с кнопками, текстовыми полями и другими элементами управления для взаимодействия с пользователями.

Управление базой данных. Для хранения информации о доступных местах, забронированных местах и других данных необходима база данных. В данном случае выбрана база данных SQLite, которая является легкой и простой в использовании. Для работы с базой данных SQLite используется класс MyDB, который обеспечивает подключение к базе данных, выполнение SQL-запросов и обработку результатов.

Отображение доступных мест. Для отображения доступных мест в зале кинотеатра можно использовать графический виджет, такой как таблица или сетка, где каждая ячейка представляет отдельное место. Места могут быть представлены в виде кнопок или других элементов, которые можно выбирать для бронирования.

**3.2 Объектно-ориентированное программирование**

Объектно-ориентированное программирование – это методология программирования, которая использует объекты для моделирования и организации кода. В C++ ООП реализуется с помощью классов и объектов [7].

Классы в C++ являются шаблонами для создания объектов. Они определяют, какие данные и функции будут содержать объекты этого класса.

Объектно-ориентированное программирование (ООП) - это парадигма программирования, которая организует программу вокруг объектов, представляющих сущности их свойствами и поведением. В ООП основное внимание уделяется моделированию реального мира, где объекты взаимодействуют друг с другом, передают сообщения и сотрудничают для выполнения задач. ООП в C++ поддерживает четыре основных принципа:

1. Инкапсуляция: Инкапсуляция означает сокрытие деталей реализации и объединение данных и методов в одном объекте.
2. Наследование: Наследование позволяет создавать новые классы на основе существующих, переиспользуя их код и добавляя новые функции.
3. Полиморфизм: Полиморфизм позволяет использовать один и тот же интерфейс для различных типов данных;
4. Абстракция: Абстракция означает представление сложных систем через более простые интерфейсы.

Итак, изучение объектно-ориентированного программирования (ООП) выдало себя как преобразующий этап в моем понимании и подходе к разработке программного обеспечения. Погружение в мир ООП расширило мои знания и компетенции, обогатив мой инструментарий в области программирования.

Одним из ключевых выводов, которые я сделал в процессе написания курсовой работы, является то, что ООП не только предоставляет эффективные средства организации кода, но и обеспечивает более высокий уровень абстракции, что делает разработку программ более интуитивной и гибкой. Использование классов и объектов позволяет создавать модульные, легко читаемые и поддерживаемые системы.

В завершение, несомненно, что изучение ООП открывает перед разработчиком множество возможностей для создания инновационных и эффективных решений. ООП становится неотъемлемым инструментом для современного разработчика, обеспечивая не только технические навыки, но и фундаментальное понимание архитектурных принципов, необходимых для успешного участия в динамичном мире разработки программного обеспечения.

**3.3 Базы данных SQLite**

Приложение также использует SQLite для работы с базой данных SQLite - это встроенная реляционная база данных, которая предоставляет легкую и эффективную систему управления базами данных. Она является простой в использовании и не требует настройки сервера или установки дополнительного программного обеспечения. Вот некоторые ключевые характеристики базы данных SQLite:

1. Легковесность: SQLite является компактной и легкой базой данных, которая может быть встроена непосредственно в приложение. База данных SQLite представляет собой один файл, который хранится на диске, и не требует запуска отдельного сервера для работы с ней.
2. Кросс-платформенность: SQLite поддерживается на различных платформах, включая Windows, macOS, Linux и другие операционные системы. Это обеспечивает универсальность и переносимость приложений, использующих SQLite.
3. Транзакционность: SQLite обеспечивает транзакционную обработку данных, что позволяет выполнить набор операций как единое целое. Это обеспечивает целостность данных и возможность отката транзакций в случае ошибок или сбоев.
4. Поддержка многопользовательского доступа: SQLite поддерживает одновременный доступ нескольких пользователей к базе данных. Однако он не поддерживает параллельную запись несколькими пользователями. Параллельная запись возможна только при условии, что каждый пользователь работает с отдельной таблицей или использует различные транзакции.
5. Надежность и стабильность: SQLite является надежной базой данных с хорошо проверенной стабильностью. Он обеспечивает целостность данных и резервное копирование базы данных для защиты от потери данных.
6. Поддержка расширений: SQLite предоставляет возможность использовать пользовательские функции и расширения, написанные на языках программирования C/C++ или других языках, для расширения возможностей базы данных.

SQLite является популярным выбором для различных типов приложений, включая мобильные приложения, настольные приложения, веб-сайты и другие проекты, где требуется легковесная и простая в использовании база данных.

**3.4 Фреймворк QT**

Qt - это кросс-платформенный фреймворк разработки приложений, который предоставляет инструменты и библиотеки для создания графического интерфейса, обработки событий, работы с сетью, базами данных и другими функциональными возможностям, это мощный инструмент для создания GUI в C++ [4]. Qt SQL. Qt – полностью объектно-ориентированная библиотека. Новая концепция ведения меж объектных коммуникаций, именуемая «сигналы и слоты», полностью заменяет былую, не вполне надежную модель обратных вызовов [6]. Графический интерфейс: Qt предоставляет мощные инструменты для создания графического интерфейса пользователя (GUI). Он включает в себя широкий набор виджетов, таких как кнопки, текстовые поля, таблицы, списки и другие элементы управления, а также возможности для настройки внешнего вида и стиля интерфейса. Также имеется возможность обработки событий, например нажатия клавиш клавиатуры, нажатия элементов интерфейса [5].

**3.5 Разработка требований к функционалу**

“Система бронирования мест в кинотеатре” - представляет собой программное приложение, разработанное с использованием фреймворка Qt и языка программирования C++.

В проекте есть несколько классов, каждый из которых представляет определенное окно или диалоговое окно в приложении.

Бронирование места. При выборе места пользователем и нажатии кнопки "Забронировать" необходимо выполнить следующие шаги:

- Проверить введенные данные (например, имя и фамилию) на валидность.

- Проверить доступность выбранного места (например, проверить его статус в базе данных).Если место доступно, обновить информацию в базе данных (изменить статус места на "забронировано" и сохранить информацию о посетителе).

- Обновить интерфейс, отобразить сообщение об успешном бронировании.

Отмена бронирования. При выборе забронированного места и нажатии кнопки "Отменить" необходимо выполнить следующие шаги:

- Проверить доступность отмены бронирования (например, проверить статус места и наличие информации о посетителе).

- Если отмена возможна, обновить информацию в базе данных (изменить статус места на "доступно" и удалить информацию о посетителе).

- Обновить интерфейс, отобразить сообщение об успешной отмене бронирования.

Обработка сигналов и слотов. Для обеспечения взаимодействия между различными классами и объектами системы используются сигналы и слоты.

Например, при успешном бронировании или отмене бронирования класс MainWindow может отправлять сигналы, которые приводят к обновлению интерфейса в классах book и cancel.

Так же при разработке была использована официальная документация Qt project. Qt — это мощный фреймворк для разработки кроссплатформенных приложений на C++. Он предоставляет широкий набор инструментов и библиотек для создания графических интерфейсов, обработки событий, работы с сетью, базами данных и другими задачами.

# **4 ФУНКЦИОНАЛЬНОЕ ПРОЕКТИРОВНИЕ**

В данном разделе описываются входные и выходные данные программы, диаграмма классов, а также приводится описание используемых классов и их методов.

**4.1 Входные данные и выходные**

Проект на тему "Система бронирования мест в кинотеатре" представляет собой приложение, которое обеспечивает удобное и автоматизированное бронирование мест для посетителей кинотеатра.

Структура входных и выходных данных моего данного проекта следующая:

Входные данные. Данные пользователя: Имя и фамилия пользователя для регистрации или идентификации, выбранные места в зале.

Выходные данные. Подтверждение бронирования: Успешное завершение бронирования мест: сообщение о успешном бронировании мест, сообщение об ошибке при неудачной попытке бронирования: возможные причины неудачи (недостаточное количество свободных мест, некорректные данные пользователя и т.д.), информация о свободных и занятых местах в зале после бронирования.

**4.2 Разработка диаграммы классов**

Диаграмма классов - это структурная диаграмма языка моделирования UML, которая демонстрирует общую структуру иерархии классов системы, их коопераций, атрибутов (полей), методов, интерфейсов и взаимосвязей (отношений) между ними. Она широко применяется не только для документирования и визуализации, но также для конструирования посредством прямого или обратного проектирования.

На диаграмме классы представлены в рамках, содержащих три компонента:

* В верхней части написано имя класса. Имя класса выравнивается по центру и пишется полужирным шрифтом;
* В средней части перечислены атрибуты (поля) класса;
* В нижней части перечислены методы класса.

Целью создания диаграммы классов является графическое представление статической структуры классов.

Диаграмма классов для приложения «Система бронирования мест в кинотеатре» приведена в приложении А.

**4.3 Описание классов**

* + 1. **Класс book:**

Конструктор book::book(QWidget \*parent) это конструктор класса book. Он принимает указатель на родительский виджет типа QWidget. Конструктор инициализирует пользовательский интерфейс для окна бронирования мест в кинотеатре. Он также инициализирует цвета для отображения валидности ввода.

Деструктор book::~book() это деструктор класса book, который будет вызываться при уничтожении объекта. Деструктор освобождает ресурсы, связанные с пользовательским интерфейсом окна.

Метод book::updateUI() этот метод обновляет пользовательский интерфейс окна бронирования мест. Он очищает комбо-бокс cmbSeats, выполняет запрос к базе данных для получения доступных мест и добавляет их в комбо-бокс. Если количество доступных мест равно или меньше нуля, кнопка btnBook становится неактивной.

Методы обработки событий. Класс book содержит несколько слотов для обработки событий, таких как нажатие кнопки и изменение текста в текстовых полях. Например, метод book::on\_btnBook\_clicked() вызывается при нажатии кнопки "Забронировать" и выполняет определенные действия.

* + 1. **Класс cancel:**

Конструктор cancel::cancel(QWidget \*parent) это конструктор класса cancel. Он принимает указатель на родительский виджет типа QWidget. Конструктор инициализирует пользовательский интерфейс для окна отмены бронирования мест в кинотеатре.

Деструктор cancel::~cancel() это деструктор класса cancel, который будет вызываться при уничтожении объекта. Деструктор освобождает ресурсы, связанные с пользовательским интерфейсом окна.

Метод cancel::updateUI() этот метод обновляет пользовательский интерфейс окна отмены бронирования мест. Он может выполнять различные действия, связанные с обновлением интерфейса, в зависимости от логики приложения.

Методы обработки событий. Класс cancel содержит один слот для обработки события - нажатие кнопки "Отменить". Например, метод cancel::on\_btnCancel\_clicked() будет вызываться при нажатии кнопки "Отменить" и выполнять соответствующие действия.

* + 1. **Класс MyDB:**

Приватный конструктор MyDB::MyDB() это приватный конструктор класса MyDB, что означает, что он может быть вызван только внутри класса. Конструктор выполняет инициализацию объекта MyDB и настройку соединения с базой данных.

Статический метод MyDB\* MyDB::getInstance() этот метод возвращает указатель на единственный экземпляр класса MyDB (реализация паттерна Singleton). Если экземпляр класса уже существует, метод возвращает указатель на него. Если он еще не создан, метод создает новый экземпляр и возвращает указатель на него.

Статический метод void MyDB::ResetInstance() этот метод сбрасывает экземпляр класса MyDB, если он был создан

* + 1. **Класс MainWindow:**

Конструктор MainWindow::MainWindow(QWidget \*parent) это конструктор класса MainWindow. Он принимает указатель на родительский виджет типа QWidget. Конструктор настраивает пользовательский интерфейс для главного окна системы бронирования мест в кинотеатре. Он также инициализирует соединение с базой данных и загружает данные для отображения.

Деструктор MainWindow::~MainWindow() это деструктор класса MainWindow, который будет вызываться при уничтожении объекта. Деструктор освобождает ресурсы, связанные с пользовательским интерфейсом окна.

Методы обработки событий. Класс MainWindow содержит несколько слотов для обработки событий, таких как нажатие кнопки и изменение текста в текстовых полях. Например, метод MainWindow::on\_btnLogout\_clicked() вызывается при нажатии кнопки "Выход" и выполняет определенные действия, такие как закрытие текущего окна и открытие окна входа в систему.

Методы для навигации. Класс MainWindow содержит методы для перехода между различными окнами и функциональностями системы бронирования, например методы openBookWindow() и openCancelWindow(), которые открывают окна бронирования и отмены бронирования соответственно.

**5 РАЗРАБОТКА ПРОГРАММНЫХ МОДУЛЕЙ**

**5.1 Разработка схем алгоритмов**

Метод on\_btnBook\_clicked() выполняет процесс бронирования места. Схема метода показана в приложении Б.

Метод on\_btnCancel\_clicked() выполняет удаление записи пользователя из базы данных, основываясь на выбранном месте. Схема метода показана в приложении В.

**5.2 Разработка алгоритмов**

**5.2.1 Метод on\_btnBook\_clicked()** **класса book**

Шаг 1: Получаем текущий выбранный элемент из cmbSeats.

Шаг 2: Получаем значения из полей ввода lineEdit и lineEdit\_2 (имя и фамилия).

Шаг 3: Проверяем валидность введенного имени и фамилии с

помощью регулярных выражений.

Шаг 4: Подсвечиваем поля ввода в зависимости от их валидности.

Шаг 5: Если имя или фамилия не валидны, выводит сообщение об

ошибке и прерывает выполнение метода.

Шаг 6: Создаём объект QSqlQuery для выполнения запросов к базе

данных, подготавливаем SQL-запрос для выбора идентификатора (Id).

Шаг 7: Привязываем значение выбранного места к подготовленному

запросу.

Шаг 8: Выполняем запрос.

Шаг 9: Если запрос выполнился успешно и возвращает результат, то

сохраняет идентификатор места (seatId).

Шаг 10: Создаем объект QSqlQuery для выполнения запросов к базе

данных. Подготавливаем SQL-запрос для вставки новой записи в

таблицу users

Шаг 11: Привязываем значения имени, фамилии и seatId к

подготовленному запросу.

Шаг 12: Выполняем запрос на вставку новой записи в базу данных.

Шаг 13: Если запрос выполнился успешно, выводит сообщение об

успешном бронировании места.

Шаг 14: Создаем объект QSqlQuery для выполнения запросов к базе

данных. : Подготавливаем SQL-запрос для обновления статуса места на

"N" (недоступно)

Шаг 15: Привязываем значение seatId к подготовленному запросу.

Шаг 16: Выполнение запроса на обновление статуса места.

Шаг 17: Если запрос выполнен успешно, выводит сообщение об

успешном обновлении статуса места.

Шаг 18: Обновляем информацию на пользовательском интерфейсе

* + 1. **Метод on\_btnCancel\_clicked()** **класса cancel**

Шаг 1: Получаем текущий выбранный элемент из cmbSeats.

Шаг 2:Создаем объект QSqlQuery для выполнения запросов к базе данных, подготавливаем SQL-запрос для выбора идентификатора (Id).

Шаг 3: Привязываем значение выбранного места к подготовленному

запросу.

Шаг 4: Выполняем запрос.

Шаг 5: Если запрос выполнился успешно и возвращает результат, то

сохраняет идентификатор места (seatId).

Шаг 6: Создаем объект QSqlQuery для выполнения запросов к базе

данных. Подготавливаем SQL-запрос для удаления записи из таблицы users.

Шаг 7: Привязывает значение seatId к подготовленному запросу.

Шаг 8: Выполняет запрос на удаление записи из базы данных.

Шаг 9: Если запрос успешно выполнен и удалена хотя бы одна запись,

то выводит сообщение и обновляет информацию на пользовательском интерфейсе.

Шаг 10: Если запрос успешно выполнен, но не удалена ни одна запись,

то выводит сообщение и обновляет информацию на пользовательском интерфейсе.

**6. РЕЗУЛЬТАТ РАБОТЫ**

На рисунке 6.1 изображено начало работы программы. Интерфейс при запуске программы в начале работы состоит из главного окна приложения MainWindow. Визуально окно имеет типичный вид для приложений, разработанных с использованием Qt Framework.

Главное окно содержит несколько элементов управления, таких как кнопки и текстовые поля, которые позволяют пользователю взаимодействовать с программой. На нижней панели окна располагаются кнопки забронировать, отменить и сбросить.

# 

Рисунок 6.1 — Начало работы программы

На рисунке 6.2 показана работа кнопки бронирования. При нажатии кнопки, позволяет пользователю перейти к диалоговому окну бронирования билетов

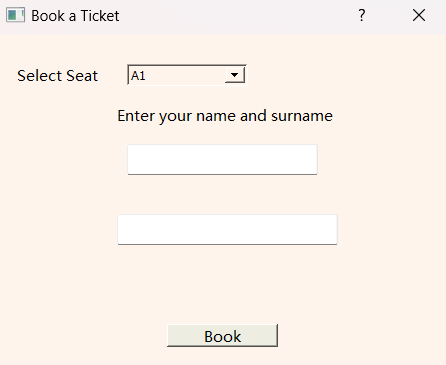


Рисунок 6.2 — Дополнительное окно кнопки бронирования

На рисунке 6.3 показано окно при неуспешном бронировании. Если при нажатии кнопки бронирования пользователь вводит неправильные данные, то программа предусматривает обработку такой ситуации и предоставляет соответствующую обратную связь пользователю.

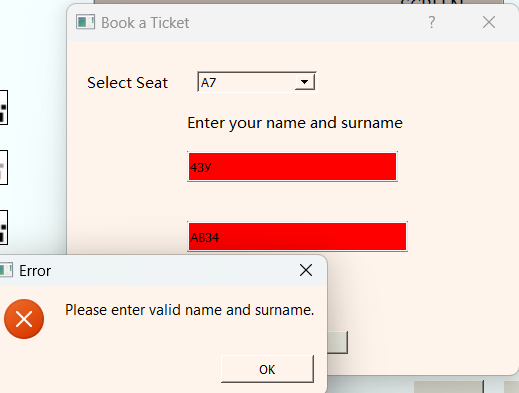


Рисунок 6.3 — Некорректный ввод данных

На рисунке 6.4 показана работа кнопки отмены. При нажатии кнопки, позволяет пользователю перейти к диалоговому окну отмены бронирования билетов

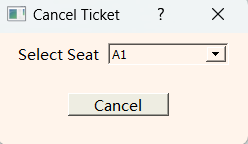


Рисунок 6.4 — Дополнительное окно кнопки отменить

На рисунке 6.5 отражена наша база данных пользователей, которая хранится в виде структурированной коллекции данных, организованной в таблице. Каждый пользователь представлен отдельной записью в таблице, где каждое поле соответствует определенной информации о пользователе.

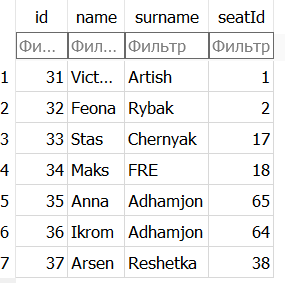


Рисунок 6.5 — Таблица базы данных

**ЗАКЛЮЧЕНИЕ**

В результате выполнения курсового проекта на тему "Система бронирования мест в кинотеатре" была разработана эффективная и удобная система, которая обеспечивает управление процессом бронирования мест в кинотеатре.

Основной акцент был сделан на создании надежной и интуитивно понятной среды для пользователей, что позволяет им легко и быстро выбирать и бронировать места в кинозале.

В ходе разработки были применены современные методы программирования и использованы соответствующие технологии. Была проведена анализ требований и характеристик системы, что позволило определить необходимые функциональные возможности и реализовать их эффективно.

Система бронирования мест в кинотеатре представляет значимую ценность для кинотеатров и их посетителей. Она упрощает процесс бронирования билетов, повышает удобство для пользователей и помогает кинотеатрам эффективно управлять своими ресурсами.

В результате выполнения курсового проекта были достигнуты поставленные цели и задачи. Разработанная система успешно реализовывает функционал бронирования мест в кинотеатре и может быть использована в реальных условиях. Работа над проектом позволила закрепить и применить полученные знания о разработке программного обеспечения и методах проектирования систем.

Таким образом, разработанная система бронирования мест в кинотеатре предоставляет удобный и эффективный способ управления местами, а ее структура и код являются примером хорошей практики программирования на языке C++ с использованием фреймворка Qt.

**СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ**

[1] Луцик Ю. А. Объектно-ориентированное программирование на языке С++: учеб. пособие /Ю. А. Луцик, В. Н. Комличенко. – Минск: БГУИР, 2008.–266 с.

[2] Дейтел, Х.М. Как программировать на С++ / Х.М. Дейтел, П.Д. Дейтел; пер. с англ. – М. : Бином, 2007. – 1152 с..

[3] "Объектно-ориентированное программирование на С++" Бьярн Страуструп

[4] "Язык программирования С++" Герберт Шилдт

[5] "C++ Primer" Липман, Лажойе, Му, Хопкинс

[6] "Алгоритмы. Построение и анализ" Кормен, Лейзерсон, Ривест, Штайн

[7] "Введение в алгоритмы" Кормен, Лейзерсон, Ривест, Штайн

[8] "Алгоритмы на C++" Роберт Седжвик, Кевин Уэйн

[9] C++: эффективное программирование. 55 способов улучшения структуры программ и стиля кода" Scott Meyers

[10] "Алгоритмы. Построение и анализ" Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein

[9] "Структуры данных и алгоритмы в C++" Robert Lafor

# ПРИЛОЖЕНИЕ А

*(обязательное)*

Диаграмма классов

# ПРИЛОЖЕНИЕ Б

*(обязательное)*

Диаграмма классов

# ПРИЛОЖЕНИЕ В

*(обязательное)*

Диаграмма классов

# ПРИЛОЖЕНИЕ Г

*(обязательное)*

Диаграмма классов