**САНКТ-ПЕТЕРБУРГСКОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ**

**ПРОФЕССИОНАЛЬНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ**

**«САНКТ-ПЕТЕРБУРГСКИЙ ТЕХНИЧЕСКИЙ КОЛЛЕДЖ**

**УПРАВЛЕНИЯ И КОММЕРЦИИ»**

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**к дипломному проекту**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Тема:** | *Разработка информационной системы* | | | | | | | | | | | | | |
|  | *для загородного Клуб-бара* «Лайт» | | | | | | | | | | | | | |
|  |  | | | | | | | | |  | | | | |
| **Руководитель** | | | |  | | | | |  | | |
| *преподаватель профессиональных дисциплин* | | | | |  |  |  | *А.А. Латыпова* | | | | | |
| *(должность)* | | | | |  | *(подпись)* |  | *(И.О. Фамилия)* | | | | | |
| **Студент** | | |  | | | | | | | | | | |
| *9ПО-42к/т* | | | | |  |  |  | *П.А. Маймин* | | | | | |
| *(группа)* | | | | |  | *(подпись)* |  | *(И.О. Фамилия)* | | | | | |
| **Специальность** | | *09.02.07 «Информационные системы и программирование »* | | | | | | | | |
|  | | | | (шифр и наименование специальности) | | | | | | | | |
|  | | | | | | | | | | | | | |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Работа допущена к защите** | | | | | | | |  | | |
| **Председатель ПЦК** | | | |  | |  | | | *А.В. Смирнова* | | |
|  | |  | | *(подпись)* | |  | | | *(И.О. Фамилия)* | | |
| **Зав. отделением** | | | |  | |  | | | *О.В. Бондарук* | | |
|  | |  | | *(подпись)* | |  | | | *(И.О. Фамилия)* | | |

Санкт-Петербург

2023 г.

**САНКТ-ПЕТЕРБУРГСКОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ**

**ПРОФЕССИОНАЛЬНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ**

**«САНКТ-ПЕТЕРБУРГСКИЙ ТЕХНИЧЕСКИЙ КОЛЛЕДЖ УПРАВЛЕНИЯ**

**И КОММЕРЦИИ»**

**УТВЕРЖДАЮ**

Председатель ПЦК\_\_\_\_\_\_\_\_\_\_\_\_/Смирнова А.В.

« 19 » апреля 2023 г.

**ЗАДАНИЕ**

**на дипломный проект**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Студент** | Маймин Пьер Александрович | | | | |
| ***Специальность*** | 09.02.07 «Информационные системы и программирование » | | | | |
| **Группа** | 9ПО-42к/т | | | | |
|  | | | | | |
| **Тема дипломной работы** | | *Разработка информационной системы для* | | | |
| *загородного клуб-бара «Лайт»* | | | | | |
|  | | | | | |
|  | | |  |  |  |
| Тема утверждена приказом по колледжу от « 05 » апреля 2023 г. №188 | | | | | |
| ***Срок сдачи дипломной*** *работы* « 08 » июня 2023 г. | | | | | |
| ***Содержание*** *работы (перечень вопросов, подлежащих рассмотрению)****:*** | | | | | |

|  |
| --- |
| **Введение:** |
| * назначение программного обеспечения, актуальность темы и практическое значение, цель, поставленные задачи |
| **Основная часть** |
| *Теоретическая часть:* |
| * характеристика клуб-бара «Лайт» |
| * компоненты информационной системы |
| * сравнительный анализ СУБД |
| * обоснование выбора языка для разработки |
| *Практическая часть:* |
| * постановка задачи; |
| * проектирование функциональных возможностей системы |
| * проектирование и разработка базы данных |
| * разработка пользовательского приложения |
| * тестирование разработанной автоматизированной системы |
| **Заключение:** |
| * выводы и предложения как теоретического, так и практического характера, полученные в результате дипломного проектирования. |

|  |
| --- |
| **Перечень иллюстрированного материала** *(кол-во листов и их содержание)* |
| Таблицы, рисунки, листинг программного кода |
| **Руководитель** Антонина Александровна Латыпова |
| (Имя, Отчество, Фамилия) |

**График выполнения проекта***(работы)*

|  |  |  |
| --- | --- | --- |
| Раздел проекта (работы) | Календарный срок выполнения | Отметка о выполнении |
| Подбор материала, его анализ и обобщение | 05.05 |  |
| Обзор и анализ предметной области | 12.05 |  |
| Программная реализация системы | 17.05 |  |
| Представление раздела «Теоретическая часть» | 23.05 |  |
| Представление раздела «Практическая часть» | 31.05 |  |
| Проверка ВКР, составление отзыва | 03.06 |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Дата выдачи задания |  |  | *А.А. Латыпова* |
|  | (подпись руководителя, дата) |  | (И.О.Ф. руководителя) |
| ***С заданием ознакомлен(а)*** |  |  | *П.А. Маймин* |
|  | (подпись студента, дата) |  | (И.О.Ф. студента) |

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 5](#_Toc105064516)

[ГЛАВА 1. ТЕОРЕТИЧЕСКИЕ АСПЕКТЫ РАЗРАБОТКИ ИНФОРМАЦИОННОЙ СИСТЕМЫ 7](#_Toc105064517)

[1.1 Характеристика Клуб-бара «Лайт» 7](#_Toc105064518)

[1.2 Компоненты информационной системы 8](#_Toc105064519)

[1.3 Сравнительный анализ СУБД 10](#_Toc105064520)

[1.4 Обоснование выбора языка для разработки 15](#_Toc105064521)

[ГЛАВА 2. РАЗРАБОТКА АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ УЧЕТА ОПТОВЫХ ПРОДАЖ СТРОЙМАТЕРИАЛОВ ДЛЯ КОМПАНИИ ООО «ОСНОВА» 19](#_Toc105064522)

[2.1 Постановка задачи 19](#_Toc105064523)

[2.2 Проектирование функциональных возможностей системы(таблицы по типам полей) 12](#_Toc105064524)

[2.3 Проектирование и разработка базы данных(курсовик) 14](#_Toc105064525)

[2.4 Разработка пользовательского приложения(все окта последовательно) 19](#_Toc105064526)

[2.5 Тестирование разработанной автоматизированной системы 29](#_Toc105064527)

[ЗАКЛЮЧЕНИЕ 31](#_Toc105064528)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ И ЛИТЕРАТУРЫ 32](#_Toc105064529)

[ПРИЛОЖЕНИЯ 34](#_Toc105064530)

# ВВЕДЕНИЕ

В настоящее время информационные технологии стали неотъемлемой частью жизни людей и бизнеса. Современные технологии позволяют создавать эффективные и удобные системы для автоматизации бизнес-процессов. В данной работе будет рассмотрена разработка информационной системы для загородного клуб-бара «Лайт».

Загородные клуб-бары пользуются большой популярностью среди любителей активного отдыха и качественного сервиса. Однако, управление бизнесом такого рода предприятий требует большого количества ресурсов и времени, что может привести к снижению эффективности работы и ухудшению качества обслуживания.

Разработка информационной системы позволит автоматизировать процессы управления бизнесом, упростить работу персонала и повысить качество обслуживания гостей. Система будет включать в себя модули управления заказами, складского учета, учета посещений и финансового учета.

В ходе работы будут рассмотрены современные методы и подходы к разработке информационных систем, а также специфика работы загородного клуб-бара. Результатом работы будет полнофункциональная информационная система, которая поможет оптимизировать работу и повысить качество обслуживания гостей.

Внедрение информационной системы на предприятии необходимо, так как сотрудники могут испытывать трудности при большом количестве заказов и допускать большое количество ошибок из-за человеческого фактора.

Целью данной работы является разработка клиентского и административного модуля системы, для удобства клиентам просматривать и делать заказы, а операторам для лёгкой регистрации блюд и заведений для клуб-бара «Лайт».

Для реализации цели необходимо решить ряд задач:

* проанализировать предметную область;
* выбрать СУБД для размещения базы данных;
* выбрать язык программирования для реализации программного продукта;
* разработать программный продукт.

# ГЛАВА 1. ТЕОРЕТИЧЕСКИЕ АСПЕКТЫ РАЗРАБОТКИ ИНФОРМАЦИОННОЙ СИСТЕМЫ.

## 1.1 Характеристика Клуб-бара «Лайт»

Клуб-бар «Лайт» – это одновременно клуб активного отдыха и бар. В него можно прийти, чтобы хорошо провести время с друзьями или одному. Клуб-бар предоставляет широкий выбор, как игр, так и разнообразного меню на любой вкус.

На данный момент существует много разных баров, кафе и клубов, в которые можно прийти и хорошо отдохнуть. Но клуб-бар «Лайт» – это уникальное место, где можно не только насладиться вкусной едой и авторскими коктейлями, но и провести время, в высококачественных игровых зонах, за увлекательными играми, такими как: бильярд, мини-гольф, которые являются одними из самых популярных игр, а также множеством других, что делает его идеальным местом для людей, которые любят активный отдых. Каждое блюдо готовится из свежих и качественных продуктов, что делает его вкус неповторимым.

Клуб-бар «Лайт» идеально подходит для проведения корпоративных мероприятий, дней рождения и других особенных событий, являясь отличным местом для людей, которые хотят провести время в хорошей компании за игрой и вкусной едой.

Также имеется несколько заведений в различных местах, штат которых состоит из девяти позиций. В штат входят: директор, администратор ИС, руководитель, заведующий игровыми помещениями, заведующий кухней, повара, кассиры, менеджер по закупкам и заведующий складом (Рисунок 1).

Рисунок 1 – Организация предприятия

## 1.2 Компоненты информационной системы

Компоненты информационной системы могут включать в себя несколько модулей, каждый из которых выполняет свои функции. Рассмотрим основные компоненты информационной системы для загородного клуб-бара:

1. Модуль управления заказами - позволяет оформлять заказы на блюда и напитки, редактировать их, отслеживать статус выполнения заказов и контролировать их оплату. Модуль также может включать функции управления меню, добавления новых блюд и напитков, а также их удаления или изменения.

2. Модуль складского учета - отвечает за учет и контроль запасов продуктов и напитков на складе. Модуль позволяет контролировать количество продуктов, отслеживать сроки годности и контролировать поставки.

3. Модуль учета посещений - позволяет отслеживать посещаемость загородного клуб-бара и контролировать доступ гостей. Модуль может включать функции управления абонементами, контроля доступа в зоны клуба, учета посещений и контроля допуска гостей.

4. Модуль финансового учета - отвечает за учет финансовых операций, связанных с работой загородного клуб-бара «Лайт». Модуль позволяет контролировать доходы и расходы, анализировать финансовые показатели и планировать бюджет.

Все модули информационной системы должны быть интегрированы между собой, чтобы обеспечить эффективное управление бизнесом. Кроме того, информационная система должна быть легко масштабируемой и готовой к внедрению новых функций и модулей в будущем.

В этом проекте на данном этапе будет реализован первый модуль(Модуль управления заказами), как основной способ удовлетворения потребностей гостей и как пример информационной системы клуб-бара.

Данный модуль информационной системы обеспечивает лёгкие инструменты для содержания целостности меню бара и возможность регистрировать и изменять его блюда, также добавлять информацию о новых заведениях и имеющихся в них помещений для игр. Клиентское же приложение позволяет легко делать заказы на доставку и бронировать обеденные столы или игровые помещения.

## 1.3 Сравнительный анализ СУБД

Современный рынок информационных технологий предоставляет широкий выбор СУБД (систем управления базами данных), каждая из которых имеет свои преимущества и недостатки. Поэтому, при разработке информационной системы, важно провести сравнительный анализ СУБД, чтобы выбрать наиболее подходящую для данного проекта. В данной главе будет рассмотрен сравнительный анализ нескольких популярных СУБД, таких как MSSQL, MySQL и PostgreSQL, по таким критериям, как производительность, надежность, функциональность, масштабируемость, стоимость и совместимость. Основываясь на результатах анализа, будет выбрана наиболее подходящая СУБД для разработки информационной системы.

1. MSSQL

Microsoft SQL Server — система управления реляционными базами данных (РСУБД), разработанная корпорацией Microsoft. Основной используемый язык запросов — Transact-SQL, создан совместно Microsoft и Sybase. Transact-SQL является реализацией стандарта ANSI/ISO по структурированному языку запросов (SQL) с расширениями. Используется для работы с базами данных размером от персональных до крупных баз данных масштаба предприятия.

Производительность:

MSSQL – это одна из самых быстрых СУБД на рынке. Она способна обрабатывать огромные объемы данных, поддерживая многопоточность и распределенную обработку.

Надежность:

MSSQL является очень надежной СУБД и имеет хорошую репутацию в этой области. Она обеспечивает высокий уровень защиты данных и предоставляет механизмы резервного копирования и восстановления.

Функциональность:

MSSQL предоставляет широкий спектр функциональных возможностей, таких как хранимые процедуры, триггеры, представления и т.д. Она также поддерживает многие стандарты SQL и обеспечивает высокую совместимость с другими СУБД.

Масштабируемость:

MSSQL обеспечивает хорошую масштабируемость, что позволяет ей работать с большими объемами данных и высокой нагрузкой. Она также поддерживает кластеризацию и репликацию, что позволяет распределить нагрузку между несколькими серверами.

Стоимость:

MSSQL является коммерческой СУБД и имеет определенную стоимость, в отличии от MySQL и PostgreSQL, которые являются бесплатными и открытыми для использования.

Совместимость:

MSSQL обеспечивает высокую совместимость с другими СУБД и поддерживает стандарты SQL. Она также поддерживает различные ОС, такие как Windows и Linux.

MSSQL является одной из лучших СУБД на рынке, обеспечивая высокую производительность, надежность, функциональность и масштабируемость, обеспечивая хранение огромного количества данных.

1. MySQL

MySQL является решением для малых и средних приложений. Обычно MySQL используется в качестве сервера, к которому обращаются локальные или удалённые клиенты, однако в дистрибутив входит библиотека внутреннего сервера, позволяющая включать MySQL в автономные программы.

Производительность:

MySQL обеспечивает высокую производительность и быстродействие. Она способна работать с большими объемами данных и поддерживает многопоточность и распределенную обработку.

Надежность:

MySQL является надежной СУБД и обеспечивает высокий уровень защиты данных. Она предоставляет механизмы резервного копирования и восстановления, а также поддерживает транзакции ACID.

Функциональность:

MySQL предоставляет широкий спектр функциональных возможностей, таких как хранимые процедуры, триггеры, представления и т.д. Она также поддерживает многие стандарты SQL и обеспечивает высокую совместимость с другими СУБД.

Масштабируемость:

MySQL обеспечивает хорошую масштабируемость и может работать с большими объемами данных. Она также поддерживает кластеризацию и репликацию, что позволяет распределить нагрузку между несколькими серверами.

Стоимость:

MySQL является бесплатной и открытой для использования СУБД. Она имеет открытый исходный код и доступна для загрузки с Интернета без каких-либо ограничений.

Совместимость:

MySQL обеспечивает высокую совместимость с другими СУБД и поддерживает стандарты SQL. Она также поддерживает различные ОС, такие как Windows, Linux и Mac OS.

В целом, MySQL является хорошей СУБД с высокой производительностью, надежностью и функциональностью, которая также обладает хорошей масштабируемостью и высокой совместимостью с другими СУБД. Более того, она бесплатна и открыта для использования.

1. PostgreSQL

PostgreSQL — свободная объектно-реляционная система управления базами данных. Существует в реализациях для множества UNIX-подобных платформ, включая AIX, различные BSD-системы, HP-UX, IRIX, Linux, macOS, Solaris/OpenSolaris, Tru64, QNX, а также для Microsoft Windows. PostgreSQL базируется на языке SQL и поддерживает многие из возможностей стандарта SQL:2011

Производительность:

PostgreSQL обладает высокой производительностью, поддерживая многопоточность и распределенную обработку, что позволяет обрабатывать большие объемы данных. Он также имеет хорошую производительность при работе с OLTP- и OLAP-задачами. Кроме того, PostgreSQL позволяет настраивать параметры производительности в соответствии с требованиями проекта.

Надежность:

PostgreSQL является очень надежной СУБД, которая обеспечивает механизмы резервного копирования и восстановления данных, а также поддерживает транзакционную целостность данных. Он также обеспечивает высокую отказоустойчивость и устойчивость к сбоям.

Функциональность:

PostgreSQL предоставляет широкий спектр функциональных возможностей, таких как поддержка многих стандартов SQL, поддержка геоинформационных данных, хранение JSON-данных и многие другие. Он также поддерживает многие расширения, которые могут быть использованы для расширения функциональности.

Масштабируемость:

PostgreSQL обеспечивает высокую масштабируемость, позволяя работать с большими объемами данных и поддерживая репликацию данных на несколько серверов. Он также обеспечивает возможность горизонтального масштабирования с помощью шардинга.

Стоимость:

PostgreSQL является бесплатной и открытой СУБД, которая не требует оплаты за лицензию. Однако, при использовании PostgreSQL в коммерческих проектах может потребоваться оплата за поддержку и консультации.

Совместимость:

PostgreSQL является очень совместимой СУБД, которая может работать на различных операционных системах, в том числе на Windows, Linux и macOS. Он также может быть интегрирован с другими СУБД, такими как MySQL и Oracle.

В целом, PostgreSQL является мощной и надежной СУБД, которая обладает широким спектром функциональных возможностей, высокой производительностью и масштабируемостью. Он также является бесплатным и открытым решением, что делает его привлекательным для использования в различных проектах.

Для приведенных СУБД мною была построена таблица с сравнение основных качеств программ по пятибалльной шкале (Таблица 1).

Таблица 1 – Сравнительный анализ СУБД

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Производительность | Надежность | Функциональность | Масштабируемость | Стоимость | Совместимость |
| MSSQL | 5 | 4 | 5 | 5 | 3 | 5 |
| MySQL | 5 | 4 | 4 | 5 | 5 | 3 |
| PostgreSQL | 4 | 5 | 4 | 4 | 4 | 5 |

В итоге для реализации была выбрана СУБД MSSQL, имеющая отличную способность к обработке большого количества данных и возможность лёгкого масштабирования самой базы данных, вмещающая огромное количество данных.

## 1.4 Обоснование выбора платформы и языка для разработки первого модуля информационной системы.

## Инструментом разработки является Microsoft Visual Studio – линейка продуктов компании Microsoft, включающих интегрированную среду разработки программного обеспечения и ряд других инструментальных средств.

Интегрированная среда разработки Visual Studio — это стартовая площадка для написания, отладки и сборки кода, а также последующей публикации приложений. Интегрированная среда разработки (IDE) представляет собой многофункциональную программу, которую можно использовать для различных аспектов разработки программного обеспечения. Помимо стандартного редактора и отладчика, которые существуют в большинстве сред IDE, Visual Studio включает в себя компиляторы, средства автозавершения кода, графические конструкторы и многие другие функции для упрощения процесса разработки. В качестве языка программирования был выбранный C#, интегрированный в IDE Visual Studio.

C # (C-Sharp) – это язык программирования, разработанный Microsoft и работающий на .NET Framework.

C# используется для разработки веб-приложений, настольных приложений, мобильных приложений, игр и многого другого.

Приложения Microsoft .NET работают в среде Microsoft .NET Framework в рамках Системы исполнения программ Common Language Runtime (CLR). Примененная в Microsoft .NET Framework концепция управляемого кода обеспечит надежное и безопасное выполнение программ, а также значительно уменьшит вероятность допущения ошибок в процессе программирования. Этому же способствует система обработки исключений и система автоматического освобождения неиспользуемой оперативной памяти, называемой системой сборки мусора (garbage collection).

Встроенные в язык C# и рассчитанные на среду Microsoft .NET Framework средства документирования, такие, как атрибуты и операторы комментариев специального вида, позволят существенно упростить создание конструкторской документации на программный код. Это особенно ценно при разработке больших проектов, когда из-за сложности и объемности задачи сопровождение разработки превращается в непосильную задачу и становится настоящим кошмаром для менеджера проекта.

В сочетании с мощным средством ускоренной разработки приложений Microsoft Visual Studio .NET набор языков платформы Microsoft .NET послужит отличным подспорьем при создании программ самого разного типа, как автономных, так и рассчитанных на использование в Интернете.

В качестве графической библиотеки было использовано WPF. Графическая библиотека, разработанная Microsoft, которая используется для создания интерактивных приложений с использованием .NET Framework. WPF предоставляет разработчикам мощный инструментарий для создания графических пользовательских интерфейсов (GUI), позволяя создавать более сложные и динамичные приложения, чем это было возможно ранее.

WPF использует XAML (eXtensible Application Markup Language) для определения пользовательского интерфейса и обработки событий. XAML - это декларативный язык разметки, который позволяет разработчикам определять элементы пользовательского интерфейса и их свойства, а также связывать их с логикой приложения. Это упрощает и ускоряет процесс разработки, так как разработчик может работать с элементами интерфейса, не пиша при этом много кода.

WPF также предоставляет различные визуальные элементы, такие как кнопки, текстовые поля, изображения, списки и т.д., которые могут быть легко настроены и скомпонованы в пользовательский интерфейс. Он также поддерживает анимацию, трансформацию и другие визуальные эффекты, что позволяет создавать более интерактивные и привлекательные пользовательские интерфейсы.

WPF также обеспечивает поддержку многозадачности и многопоточности, что позволяет создавать приложения, которые могут обрабатывать несколько задач одновременно. Он также поддерживает масштабирование, что делает его идеальным для создания приложений, которые могут работать на разных устройствах и разных разрешениях экрана.

Windows Presentation Foundation – это мощная и гибкая библиотека, которая позволяет разработчикам создавать более сложные и интерактивные пользовательские интерфейсы, чем это было возможно ранее. Он предоставляет различные инструменты и функции, которые упрощают процесс разработки и позволяют создавать приложения, которые работают быстро и эффективно.

Для связи и работы с базой данных была выбрана библиотека Entity Framework, позволяющая легко соединятся с СУБД и редактировать её модель.

Entity Framework (EF) – это библиотека связи с СУБД, которая позволяет разработчикам использовать объектно-ориентированный подход к работе с данными в .NET Framework. EF позволяет работать с данными в виде объектов и свойств, что делает процесс работы с данными более интуитивно понятным и удобным для разработчиков.

EF поддерживает многие СУБД, включая Microsoft SQL Server, Oracle, MySQL и другие. Он также поддерживает многие функции, такие как LINQ (Language Integrated Query), которая позволяет разработчикам использовать язык запросов C# для извлечения данных из СУБД.

EF позволяет разработчикам создавать модели данных, которые могут быть использованы для работы с данными в приложении. Модели данных могут быть созданы автоматически на основе схемы данных в СУБД, или же могут быть созданы вручную с помощью инструментов моделирования. Модели данных могут быть настроены для работы с различными типами связей между таблицами в СУБД, такими как один-ко-многим, многие-ко-многим и т.д.

EF также обеспечивает механизмы миграции, которые позволяют разработчикам изменять схему данных в СУБД, не затрагивая существующие данные. Механизмы миграции позволяют автоматически обновлять схему данных в СУБД при обновлении приложения.

EF предоставляет механизмы кэширования данных, что позволяет ускорить производительность приложения. Кэширование данных позволяет избежать повторных запросов к СУБД при работе с данными, что снижает нагрузку на сервер и ускоряет работу приложения.

Entity Framework позволяет разработчикам упростить процесс работы с данными в приложении. Он предоставляет множество функций и инструментов, которые позволяют работать с данными в виде объектов и свойств, что делает процесс работы с данными более интуитивно понятным и удобным для разработчиков.

Также для осуществление более глубокого функционала, по типу задания формата поля, используется сторонняя NuGet библиотека Extended.Wpf.Toolkit.

# ГЛАВА 2. РАЗРАБОТКА МОДУЛЯ УПРАВЛЕНИЯ ЗАКАЗАМИ ИНФОРМАЦИОННОЙ СИСТЕМЫ ЗАГОРОДНОГО КЛУБ-БАРА «ЛАЙТ»

## 2.1 Постановка задачи

Требуется разработать модуль учёта заказов информационной системы клуб-бара.

Назначение и цели создания программы: система регистрации заказов и их реализации, бронирования обеденных столов и помещений для игр.

Информационная система должна содержать следующие функции:

* Авторизация и регистрация пользователей;
* Просмотр всех заказов администратором и просмотр своих заказов клиентом;
* Создание нового заказа;
* Редактирование заказа;
* Добавление новых блюд в меню;
* Просмотр статуса заказа.
* Бронирование столов и помещений для игр.

## 2.2 Проектирование функциональных возможностей системы

В соответствии с постановкой задачи были определены пользователи и функциональный состав ИС представлен в виде диаграммы:

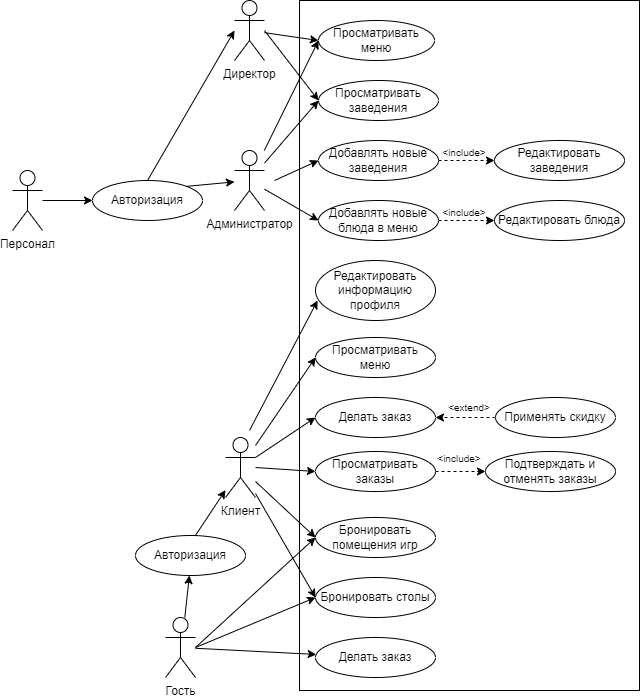


Рисунок 1 - Функциональная схема приложения

## 2.3 Проектирование и разработка базы данных

Основной задачей проекта является создание клиентского и административного приложения. Клиентское приложение позволяет заказывать еду для доставки и бронировать столы, также как и бронировать помещения для игр. Административное приложение позволяет изменять и добавлять новые блюда и заведения, доступные для бронирования столов и игровых помещений.

Входные данные:

* Информация о наличии товаров на складе;
* Информация о пользователях приложения.

Выходные данные:

* Информация о заказах;
* Информация об остатках товаров на складе;
* Отчеты о продажах.

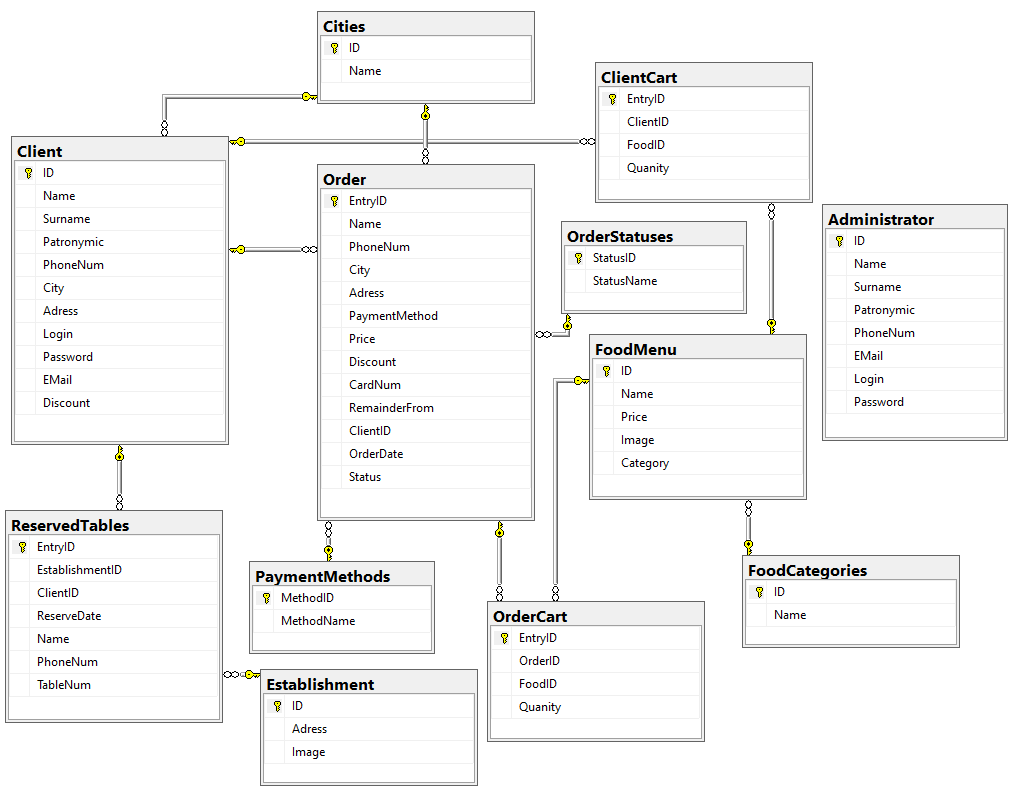
На основе данных предметной области была спроектирована логическая модель хранения данных, представленная на Рисунке 3. 

Рисунок 3 - Диаграмма базы данных

Для реализации базы данных была использована СУБД Microsoft SQL Server 2019

Всего создано 12 таблиц: Client, ClientCart, Cities, Order, OrderStatuses, OrderCart, PaymentMethods, FoodMenu, FoodCategories, ReserverdTables, Establishments, Administrator.

Для однозначного определения записей в каждой из таблиц выделен первичный ключ.

Структура полей данных таблиц базы данных для таблицы «Client» представлена в Рисунке 4.

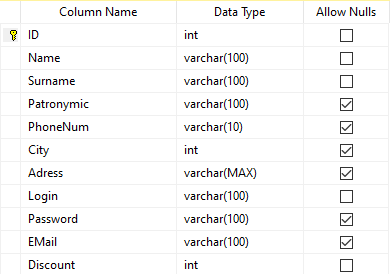


Рисунок 4 – Таблица «Client»

Структура полей данных таблиц базы данных для таблицы «Order» представлена в Рисунке 5.

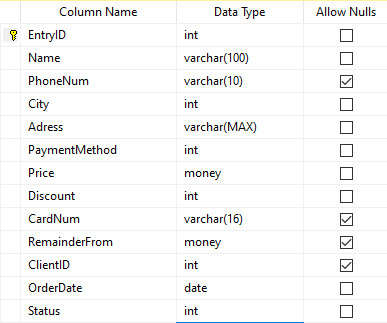


Рисунок 5 - Таблица «Order»

Структура полей данных таблиц базы данных для таблицы «FoodMenu» представлена в Рисунке 6.

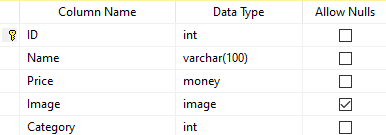


Рисунок 6 - Таблица «FoodMenu»

Структура полей данных таблиц базы данных для таблицы «ClientCart» представлена в Рисунке 7.



Рисунок 7 - Таблица «ClientCart»

Структура полей данных таблиц базы данных для таблицы «OrderCart» представлена в Рисунке 7.

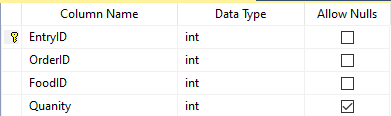


Рисунок 7 - Таблица «OrderCart»

Структура полей данных таблиц базы данных для таблицы «Establishments» представлена в Рисунке 8.

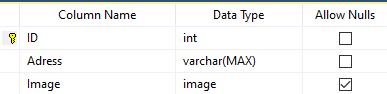


Рисунок 8 - Таблица «Establishments»

Структура полей данных таблиц базы данных для таблицы «ReservedTables» представлена в Рисунке 9.

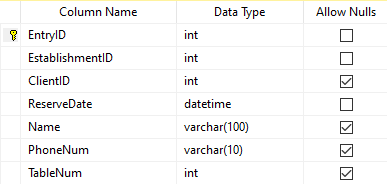


Рисунок 9 - Таблица «ReservedTables»

Структура полей данных таблиц базы данных для таблицы «Cities» представлена в Рисунке 10.

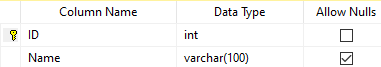


Рисунок 10 - Таблица «Cities»

Структура полей данных таблиц базы данных для таблицы «FoodCategories» представлена в Рисунке 11.

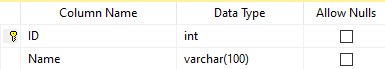


Рисунок 11 - Таблица «FoodCategories»

Структура полей данных таблиц базы данных для таблицы «OrderStatuses» представлена в Рисунке 12.

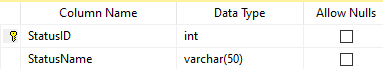


Рисунок 12 - Таблица «OrderStatuses»

Структура полей данных таблиц базы данных для таблицы «PaymentMethods» представлена в Рисунке 13.

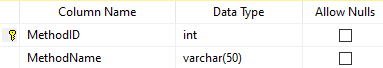


Рисунок 13 - Таблица «PaymentMethods»

Структура полей данных таблиц базы данных для таблицы «Administrator» представлена в Рисунке 14.

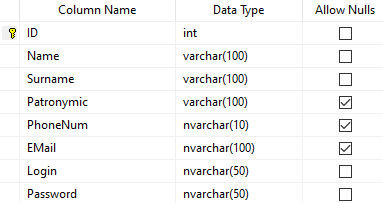


Рисунок 14 - Таблица «Administrator»

## 2.4 Разработка пользовательского приложения

Структура клиентского приложения состоит из двух окон и 10 страниц, сформированные средствами технологии пользовательского интерфейса (Рисунок 15):

* Формы
  + Главная форма
  + Форма заполнения информации банковской карты
* Страницы
  + Страница меню еды
  + Страница просмотра корзины
  + Страница оформления заказа
  + Страница бронирования столов
  + Страница бронирования игровых помещений
  + Страница входа в аккаунт
  + Страница регистрации аккаунта
  + Страница редактирования профиля
  + Страница просмотра сделанных заказов
  + Страница просмотра выбранного заказа

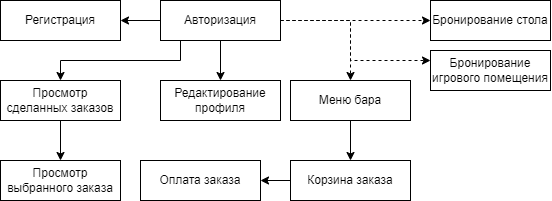


Рисунок 15 – Структура автоматизированной системы учета продаж

----------------------------------------------------

Запуск клиентского приложения осуществляется путем нажатия ярлыка, выведенного на рабочий стол компьютера пользователя двойным нажатием мыши по исполняемому файлу «BarLight.exe», после чего приложение откроется на странице меню. В приложении возможно делать заказы и бронировать в режиме гостя. Авторизованный режим предоставляет удобное автозаполнение полей и применение скидки при оплате заказа.

Для оперирования приложением используется вспомогательный класс Manager, содержащий различные свойства и методы для облегчения работы с приложением:

* Для смен страниц используются свойства-контейнеры, содержащие ссылку на контейнеры для страниц;
* Операции с базой данных осуществляются с помощью свойства Context, содержащее экземпляр базы данных;
* Различные манипуляции с названиями элементами приложения осуществлены с помощью методов внутри класса;

Главное окно приложения и страница меню бара (Рисунок 16). Данная страница предназначена для просмотра меню бара и оформления заказа для доставки на дом. Меню оформлено с помощью ListView, позволяющий создавать своё оформление содержимого. Каждый элемент таблицы содержит: изображение блюда/напитка, его наименование, цена за единицу и кнопку для добавления его в корзину заказа.



Рисунок 16 – Страница меню

Загрузка изображений выполнена с помощью конвертации загруженных в базу данных массивов байтов обратно в изображения, которые после отображаются в приложении. Дабы изображения не подкачивались каждый раз из базы данных, внутри страницы создана статическая переменная \_foodMenu типа FoodMenu имеющая единоразово загруженную таблицу содержащую меню бара, что сильно сокращает время загрузки страницы. Страница меню также содержит 6 поддерживаемых фильтров имеющихся категорий блюд: закуски, салаты, супы, горячее, напитки и всё, отображающее меню полностью. Для меню бара была создана таблица «FoodMenu». Для категорий блюд была создана таблица «FoodCatrgories». Изображения блюд взяты из интернета для демонстрации примера отображения изображений.



Рисунок 17 – Демонстрация отображения блюд категории «Горячее»

При нажатии на кнопку с изображением корзины, на элементе таблицы, выбранное блюдо/напиток добавляется в корзину (Рисунок 18). В корзину можно перейти нажав на кнопку в правой нижней части интерфейса. Также слева от иконки отображается общее количество выбранных блюд и напитков. В корзине можно просмотреть общую стоимость нескольких блюд одного типа, общую стоимость заказа, и по нажатию кнопки +/- увеличить или уменьшить количество выбранного блюда/напитка. Для хранения данных корзины во вспомогательном классе Manager был создан список типа ClientCart, имеющий структуру таблицы из БД, содержащий информацию и количество блюда/напитка. Если заказ делает авторизованный пользователь, одновременно производится запись в базу данных, содержащую ID клиента, количество и тип блюда. При изменении количества блюда нажатиями на кнопки +/- количество также перезаписывается в списке и базе данных. Если нажать на “-“, когда количество блюда равно единице, блюдо пропадёт из списка, а запись в БД также удалится.



Рисунок 18 – Страница корзины с выбранными блюдами

После окончания формирования заказа можно перейти к его оплате, нажатием на кнопку «К оплате» в правой верхней части интерфейса страницы. Если корзина оказалась пуста, высветится оповещение и запретит перейти на страницу оплаты (Рисунок 19).

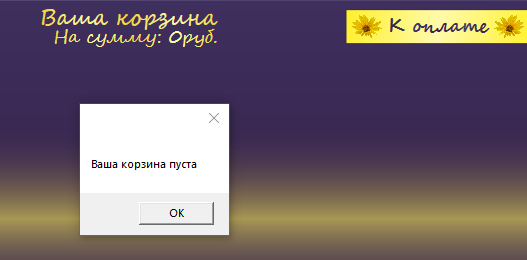


Рисунок 19 – Оповещение о пустой корзине

Страница оформления заказа (Рисунок 20) содержит поля обязательные для заполнения, содержащие информацию для успешной доставки заказа. В поле имени возможно вписать только буквы, также как в поле ввода телефона можно использовать только цифры и имеет ограничение в 10 символов. Выбор города представлен в виде выпадающего списка содержащий возможные города для доставки. Заполнение выпадающего списка создано с помощью привязки к классу listCities, наследующий тип List<Cities>. При вызове класса происходит самозаполнение списка городами из таблицы Cities.



Рисунок 20 – Страница оплаты заказа



Рисунок 21 – Пример заполнения полей информации с выпадающим списком

На странице оплаты также отображается итоговая сумма заказа в левой нижней части интерфейса страницы. Если заказчик произвёл вход в аккаунт, вместе с этим будет применена скидка (Рисунок 22), увеличивающаяся с определённой потраченной суммой.



Рисунок 22 – Пример применения скидки и выполненного автозаполнения в поля информации

Присутствуют три различных вида оплаты заказа: онлайн оплата банковской картой, картой при получении и наличными при получении.

При оплате картой открывается отдельное окно заполнения реквизитов (Рисунок 23). Его возможно закрыть, нажав на крестик сверху справа, отменив оплату онлайн. Для подтверждения необходимо заполнить поля: номера карты, CVV кода и даты окончания срока карты. Все поля имеют ограничения на написание цифр и свои ограничения длины поля. Если не все поля будут заполнены, появится сообщение о необходимости заполнить все поля (Рисунок 24). Если же дата окончания карты отмечена раньше, чем сегодняшний месяц, появится сообщения об истечении срока карты (Рисунок 25).

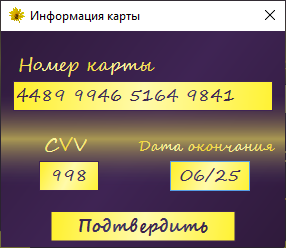


Рисунок 23 – Пример заполнения информации карты



Рисунок 24 – Сообщение о незаполненных полях

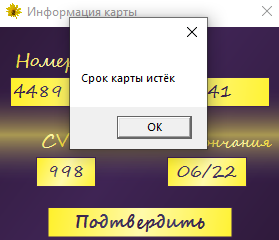


Рисунок 25 – Сообщение об истечении срока карты

На данный момент не реализован функционал настоящей проверки карты, поэтому продемонстрирован пример реализации оплаты через карту.

При выборе способа оплаты наличными при получении активируется поле ввода для заполнения какими наличными оплатит заказчик (Рисунок 26), чтобы доставщик мог подготовить сдачу. Это поле можно оставить пустым.

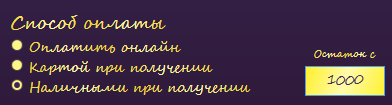


Рисунок 26 – Пример ввода в поле «Остаток с»

При нажатии на кнопку оплатить внизу страницы заказ записывается в базу данных, в таблицу Order, имея уникальный идентификатор, как номер заказа. Если пользователь до этого вошёл в аккаунт, сумма заказа зачтётся в счётчик общей скидки на все дальнейшие заказы, а информация его корзины копируется в таблицу OrderCart, после позволяя просматривать содержимое сделанных заказов. По нажатии кнопки «Ок», страница оплаты закроется и откроется страница меню.



Рисунок 27 – Окно после подтверждения заказа

Если при нажатии на кнопку «Оплатить» было не заполнено хотя бы одно поле, высветится предупреждающее сообщении о необходимости заполнить все поля (Рисунок 28). Если же все поля были заполнены, телефон написан не верно(состоит менее, чем из 10 символов), появится сообщение о неправильности ввода поля телефона (Рисунок 29).



Рисунок 28 – Сообщение о необходимости заполнить поля

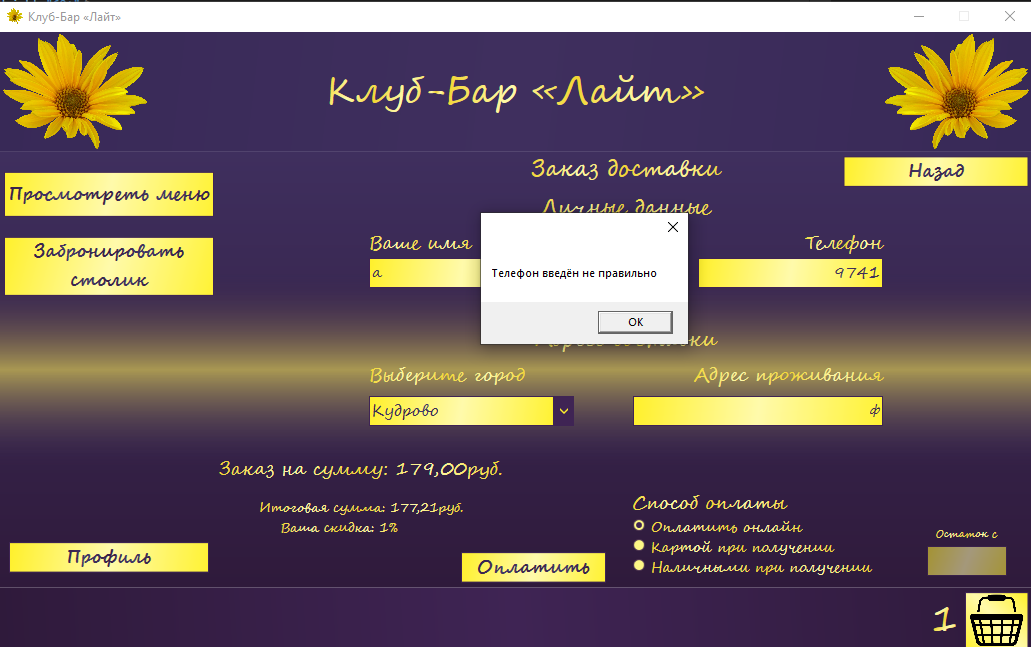


Рисунок 29 – Сообщение о неверном вводе телефона

При нажатии на кнопку «Войти в профиль», слева снизу интерфейса главного окна, открывается страница авторизации пользователя (Рисунок 30).



Рисунок 30 – Страница авторизации

На странице присутствуют поля для ввода логина и пароля. Если же у пользователя нет аккаунта, снизу поля для пароля находится ссылочный текст «Зарегистрироваться», переправляющий на страницу регистрации (Рисунок 31).

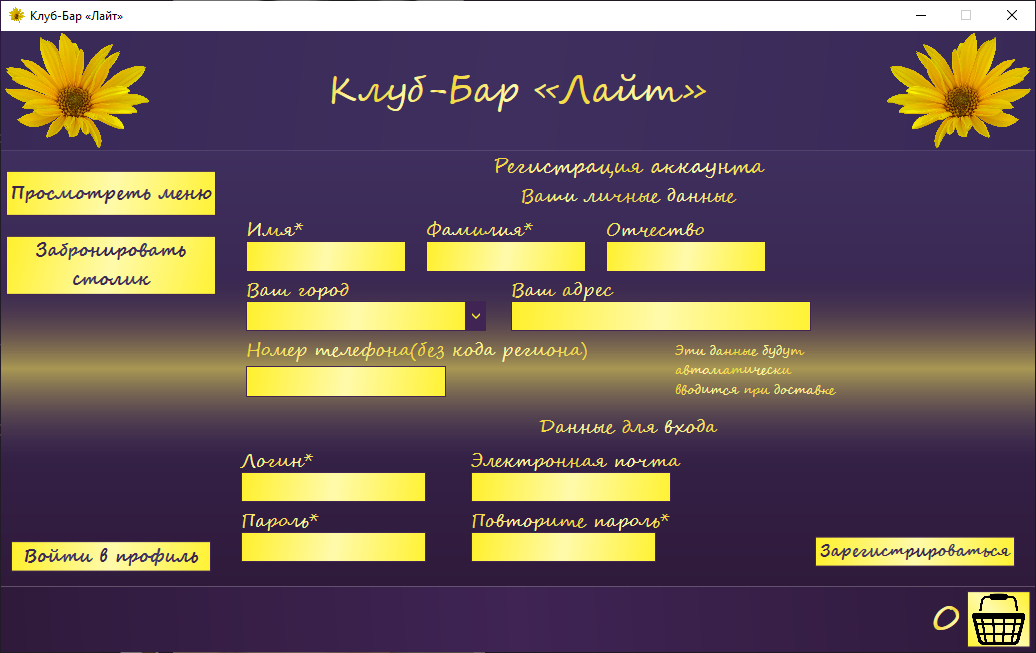


Рисунок 31 – Страница регистрации

Для регистрации необходимо заполнить обязательные для заполнения поля, они отмечены знаком «\*»:

* Имя
* Фамилия
* Логин
* Пароль, а также повторить его

Остальные поля можно дополнить или изменить позже при редактировании профиля.

Также при заполнении полей полностью поддерживается табуляция для перехода на другие поля.

Выпадающий список выбора города также заполнен с помощью класса-списка listCities.

При нажатии на кнопку «Зарегистрироваться» появляется оповещение об успешной регистрации (Рисунок 32) и переход на страницу авторизации. Для содержания информации о зарегистрированных клиентах используется таблица Client.



Рисунок 31 – Успешная регистрация

Для входа в профиль, на странице авторизации нужно ввести логин и пароль. Для примера будем использовать только что созданный аккаунт. После ввода данных и нажатии на кнопку «Войти» открывается страница профиля, содержащая заполненные данные и действующую скидку на заказы (Рисунок 32). Если же введённые данные не окажутся в базе данных, то внизу кнопки «Войти» отобразится красный текст оповещения (Рисунок 33). Кнопка входа в аккаунт меняется на кнопку, перенаправляющая на авторизованный профиль. ID авторизованного пользователя записывается в вспомогательную переменную ClientID в классе Manager, позже использующаяся во многих транзакциях. Если у пользователя были ранее выбраны некоторые блюда, когда он вышел из своего профиля, при повторном входе они обратно добавятся в корзину.

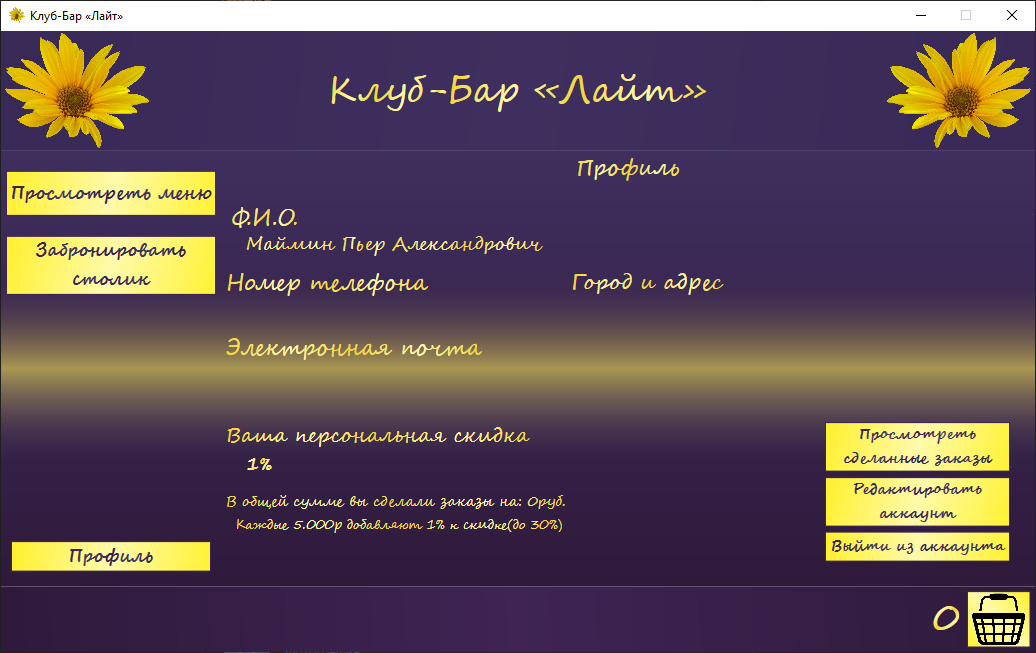


Рисунок 32 – Страница аккаунта с минимально заполненными данными



Рисунок 33 – Ошибка авторизации

Нажав на странице профиля кнопку «Редактировать аккаунт» открывается страница редактирования данных профиля (Рисунок 34). Здесь можно добавить новую или изменить уже существующую информацию.

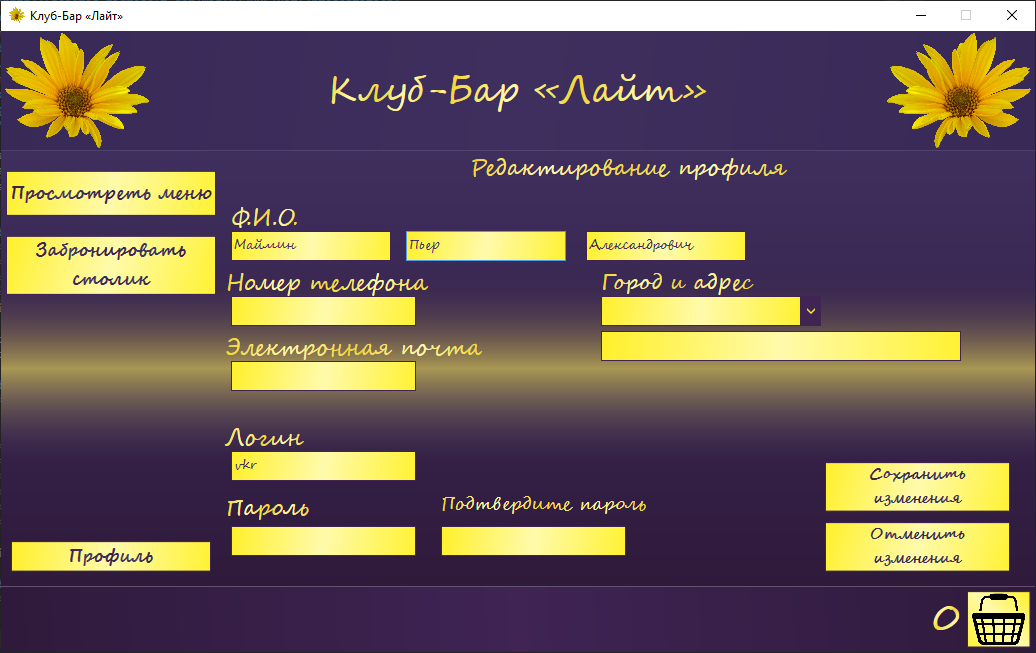


Рисунок 34 – Страница редактирования профиля

Для подтверждения изменений в профиле нужно нажать кнопку «Сохранить изменения», предварительно заново вписав и повторив пароль (Рисунок 35). На странице также поддерживается табуляция по полям ввода.

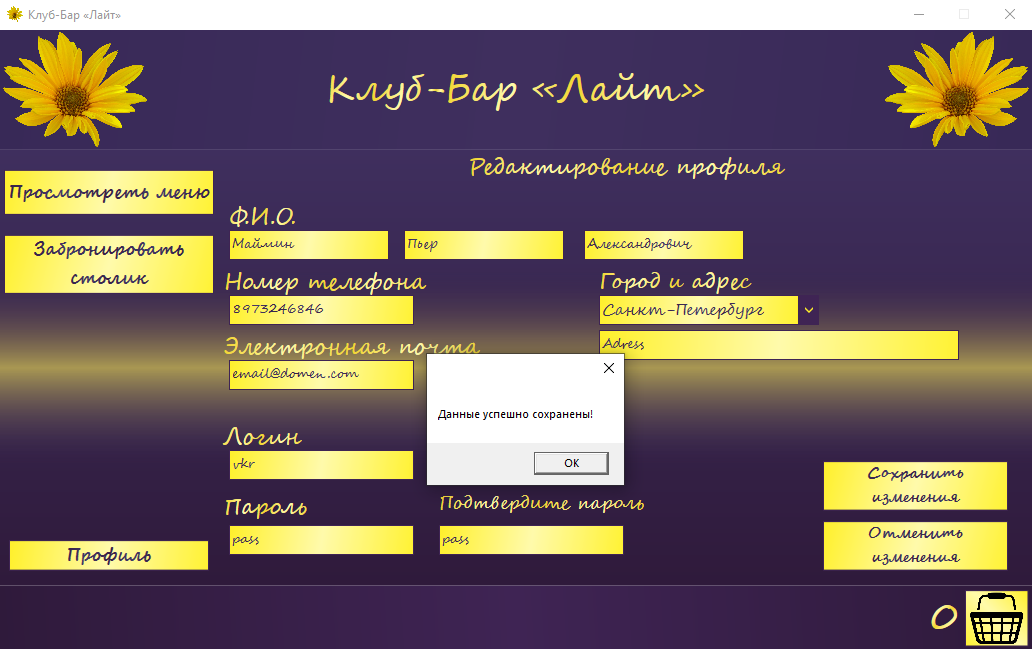


Рисунок 35 – Успешное сохранение данных



Рисунок 35 – Отображение данных во вкладке профиля

На странице редактирования присутствует обработка разных ошибок:

* Обязательные поля не заполнены (Рисунок 36);
* Не верно введён номер телефона (Рисунок 37);
* В адресе указан либо город, либо адрес без города (Рисунок 38);
* Для подтверждения не указан пароль или логин (Рисунок 39);

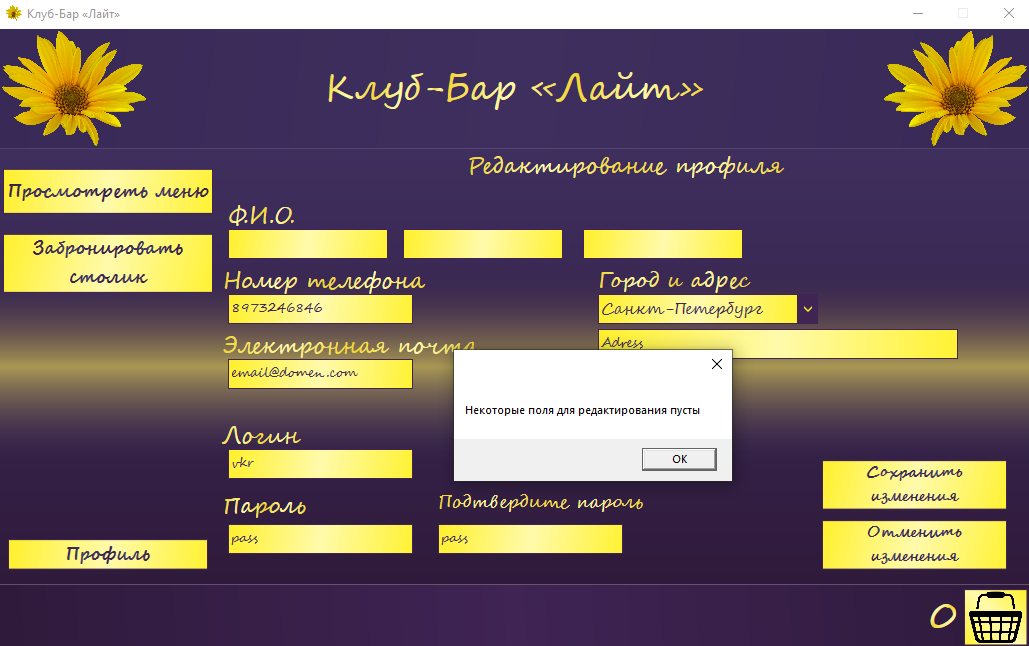


Рисунок 36 – Не заполнены обязательные поля

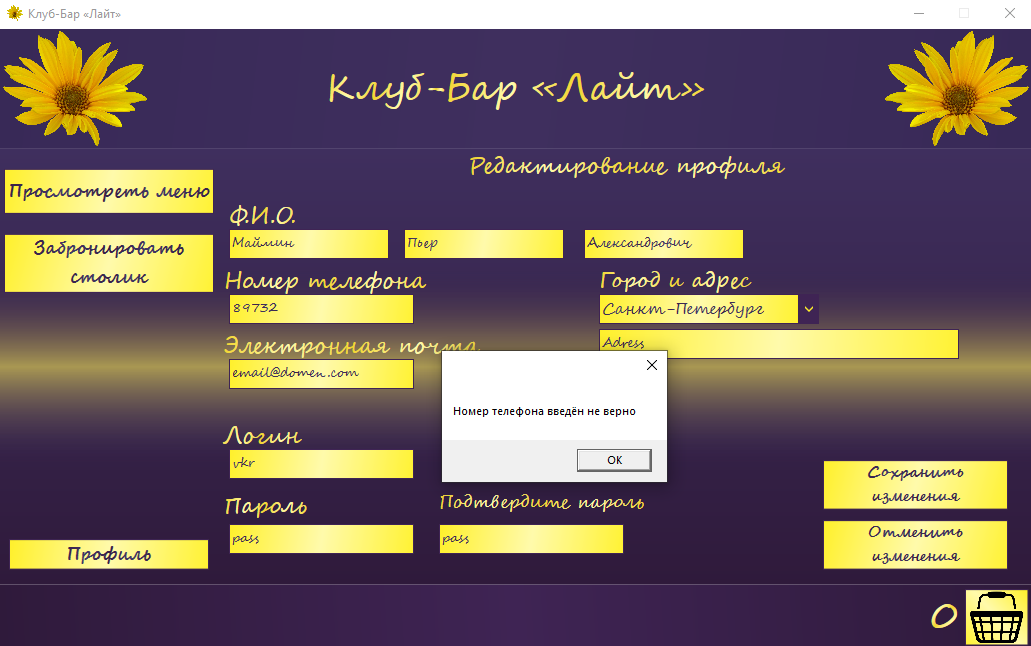


Рисунок 37 – Не верно введён номер телефона

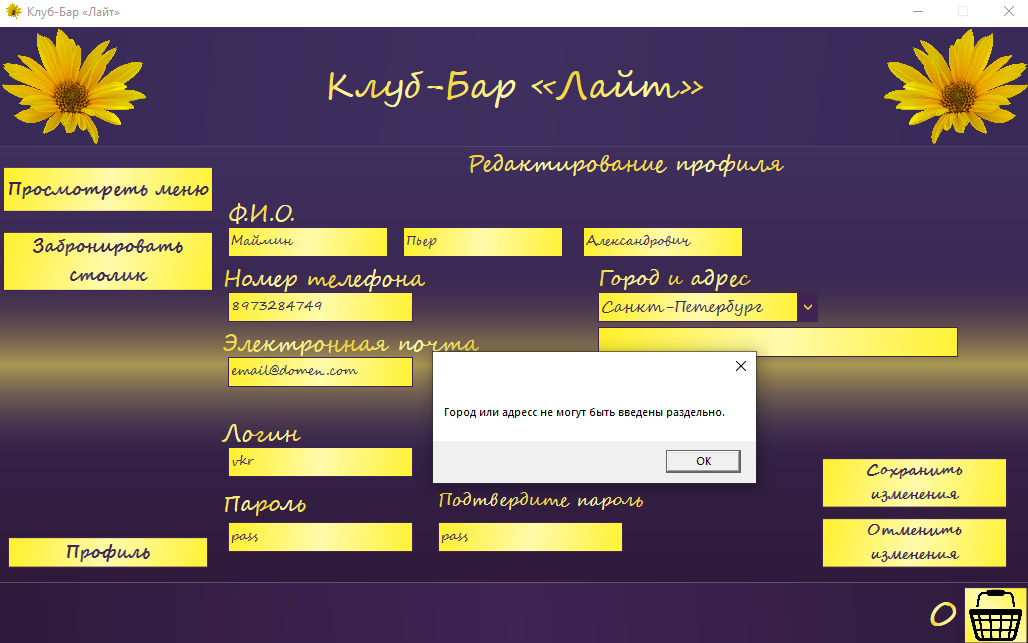


Рисунок 38 – Указан либо город, либо адрес без города

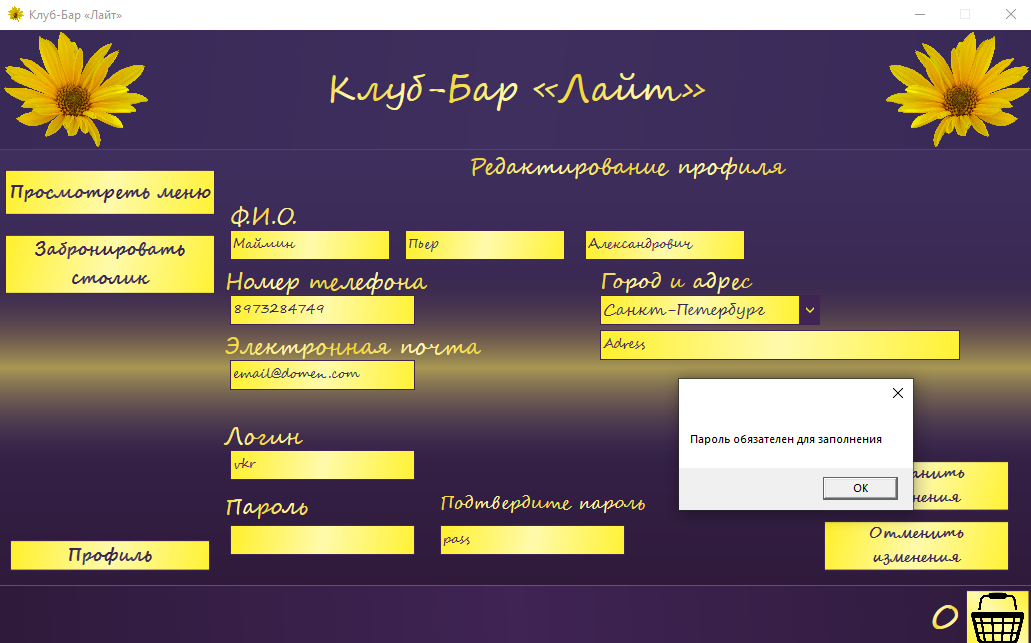


Рисунок 39 – Не введены логин и/или пароль

При нажатии на кнопку «Отменить изменения» страница редактирования закроется, вернув страницу профиля, без внесения изменений.

При нажатии на кнопку «Просмотреть сделанные заказы», на странице профиля, открывается страница ранее сделанных заказах (Рисунок 40). На странице отображаются все сделанные заказы. Отображается:

* Дата заказа;
* Номер заказа;
* Итоговая цена заказа;
* Адрес доставки заказа;
* Способ оплаты;
* Статус заказа(Выполняется, Получен, Отменён);
* Кнопка для просмотра содержимого заказа;

Нажав на кнопку просмотра заказа, открывается страница с отображением содержимого заказа (Рисунок 41). Для отображения заказов и их информации используется таблица Order. Для отображения содержимого используется таблица OrderCart, связывающаяся по ID заказа.



Рисунок 40 – Страница заказов

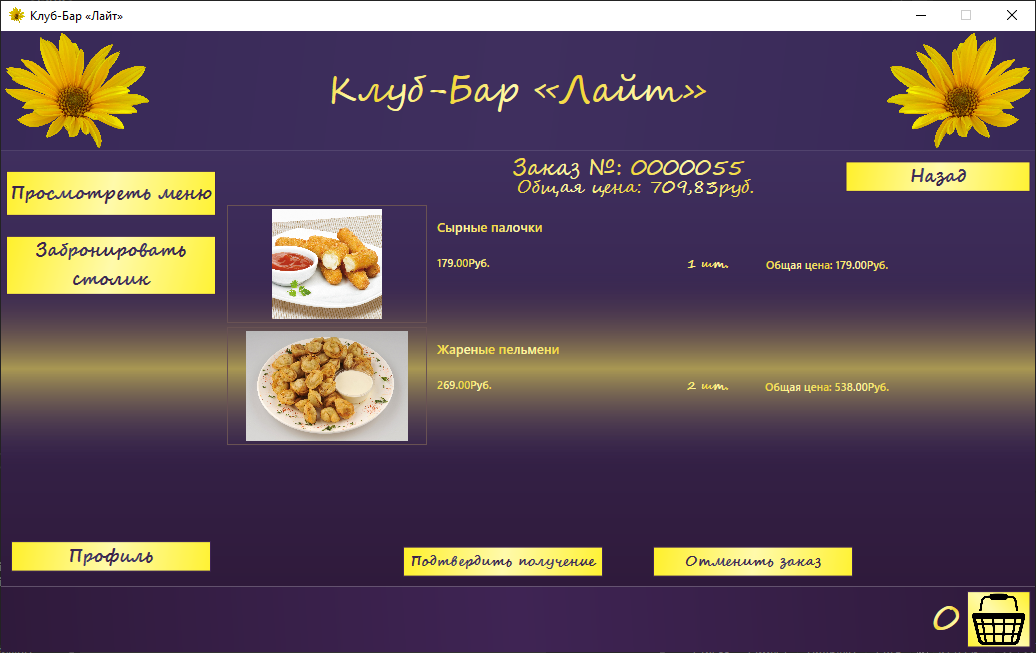


Рисунок 41 – Страница просмотра заказа

На странице просмотра заказа также отображается его номер и общая цена вверху страницы, вместе с кнопкой возвращения к списку заказов. Содержимое заказа показывается также, как и в корзине, но без возможности изменять его содержимое. Также при просмотре заказа можно подтвердить его получение или отменить, нажав на кнопки внизу страницы. В таком случае изменяется статус заказа на Получен/Отменён при подтверждении данного действия (Рисунок 42). Для хранения видов статусов используется вспомогательная таблица OrderStatuses. Планируется добавление возможности заполнения электронной формы возврата средств, если заказ был оплачен на онлайн основе.

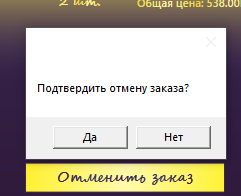
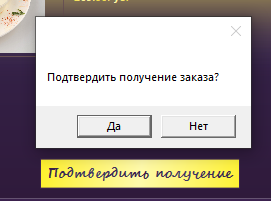


Рисунок 42 – Окна подтверждения/отмены заказа

Для примера отображения статуса в списке был выбран вариант подтверждения (Рисунок 43).



Рисунок 43 – Статус второго заказа изменён на «Получен»

Нажав на кнопку «Назад» справа вверху страницы, открывается страница профиля.

При нажатии на кнопку «Выйти из аккаунта», на странице профиля, производится выход из него, перенаправляя на страницу авторизации, также выписывая из ClientID значение пользователя, выставляя 0, далее используя гостевой режим.

По нажатии на кнопку «Забронировать столик», на главном окне приложения, открывается страница с имеющимися заведениями сети (Рисунок 44). На данный момент реализован простой вариант бронирования стола: можно выбрать заведение нажав на кнопку «Забронировать столик» снизу элемента списка. Кнопка изменит свой текст на «Выбрано», после чего, при введении данных, даты и времени можно, нажатием на кнопку «Подтвердить», забронировать свободное место на выбранное время (Рисунок 45).

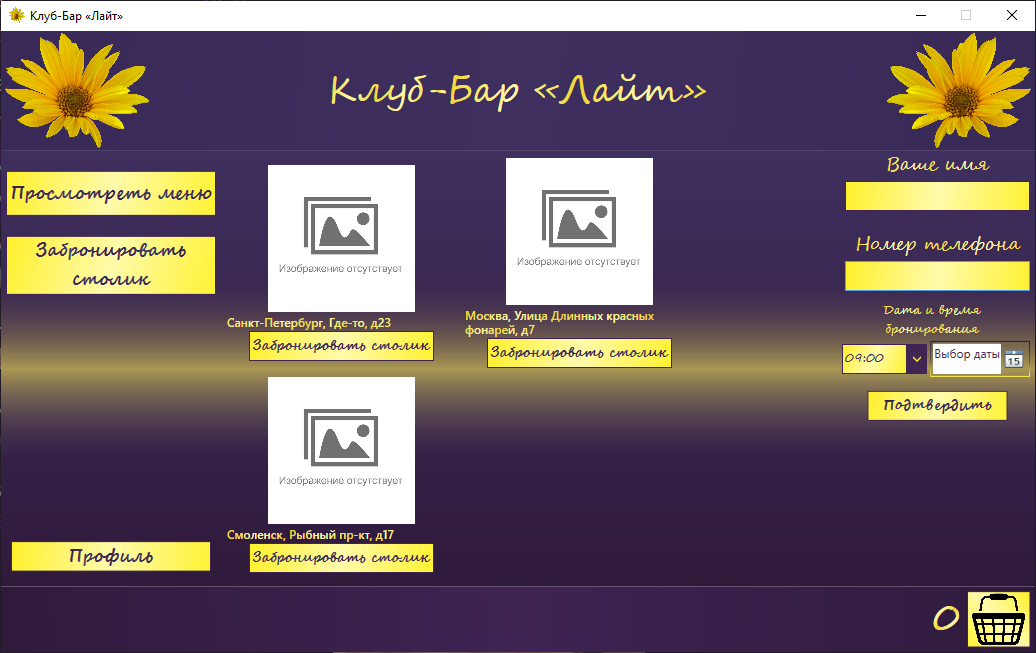


Рисунок 44 – Страница заведений

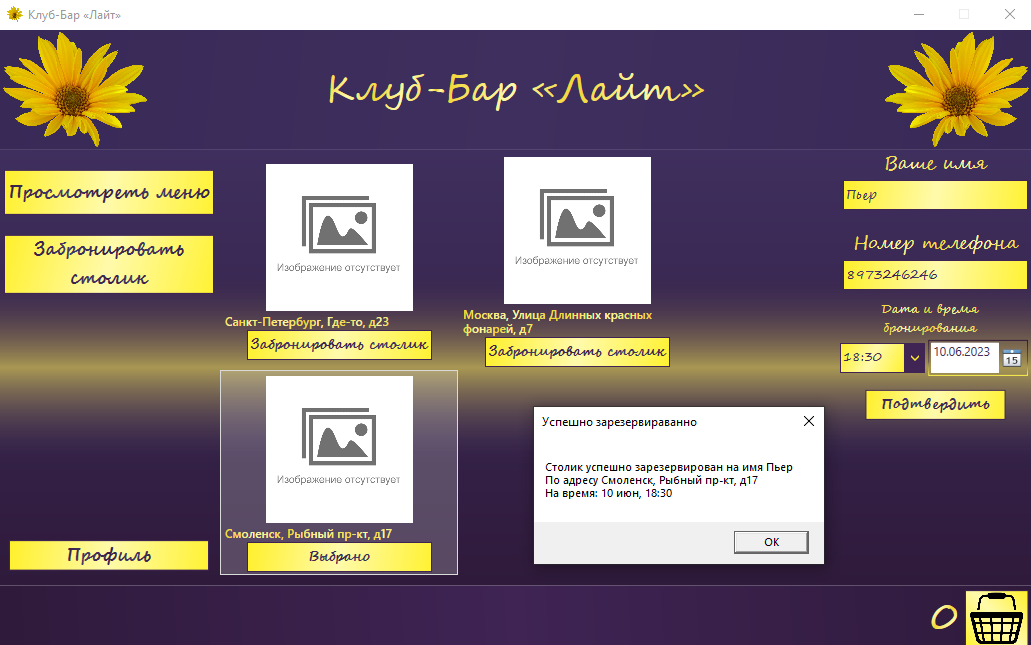


Рисунок 45 – Выбор заведения и заполнение данных для бронирования

Если ранее был произведён вход в аккаунт, поля имени и телефона будут автоматически заполнены. Выбор времени произведён с помощью выпадающего списка, содержащий список доступного времени от 9 утра до 23:45. Каждая запись сделана промежутком в 15 минут. Выбор даты выполнен с помощью DatePicker. Информация о заведениях берётся из таблицы Establishments. Данные о бронировании стола регистрируются в таблицу TableReservation. В дальнейшем планируется переработать систему бронирования столов, реализовав выбор места с помощью окна, отображающим расположение самих столов и добавить функцию бронировки игровых помещений.

На странице бронирования реализована обработка разных ошибок:

* Не выбрано заведение для брони; (Рисунок 46)
* Не введено имя клиента; (Рисунок 47)
* Не введен или введен не верно номер телефона; (Рисунок 47)
* Не выбрана дата бронирования; (Рисунок 47)
* Выбрана сегодняшняя или вчерашняя дата; (Рисунок 49)

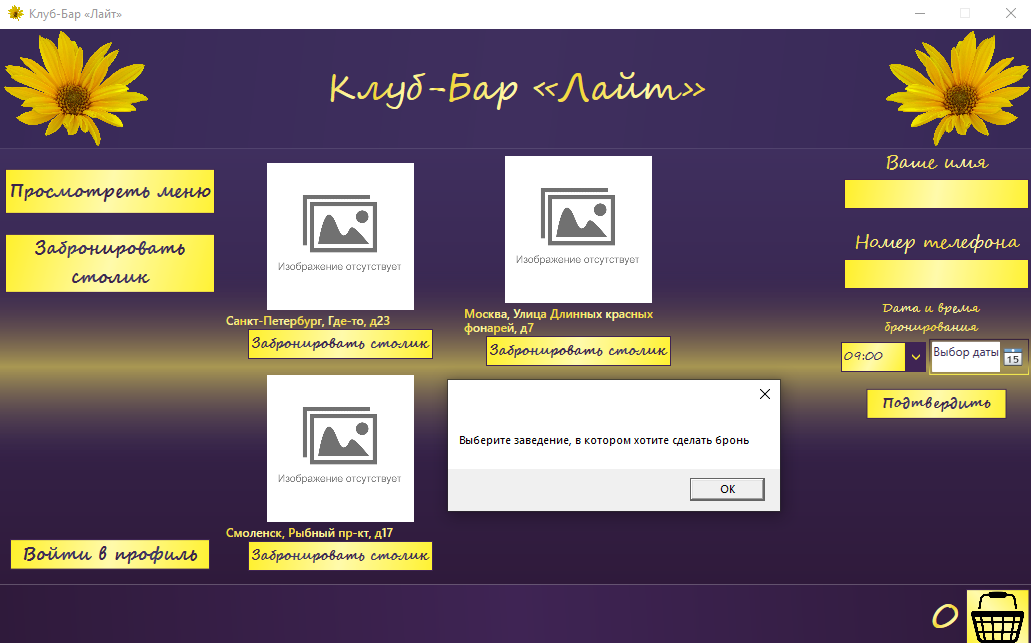


Рисунок 46 – Не выбрано заведение для бронирования

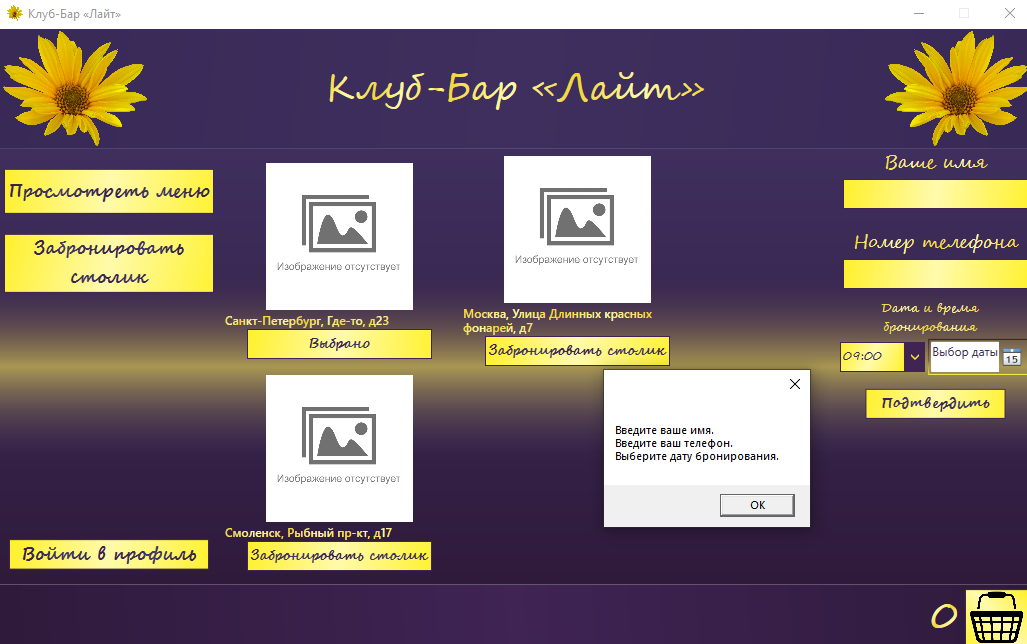


Рисунок 47 – Не введено имя клиента, телефона и не выбрана дата

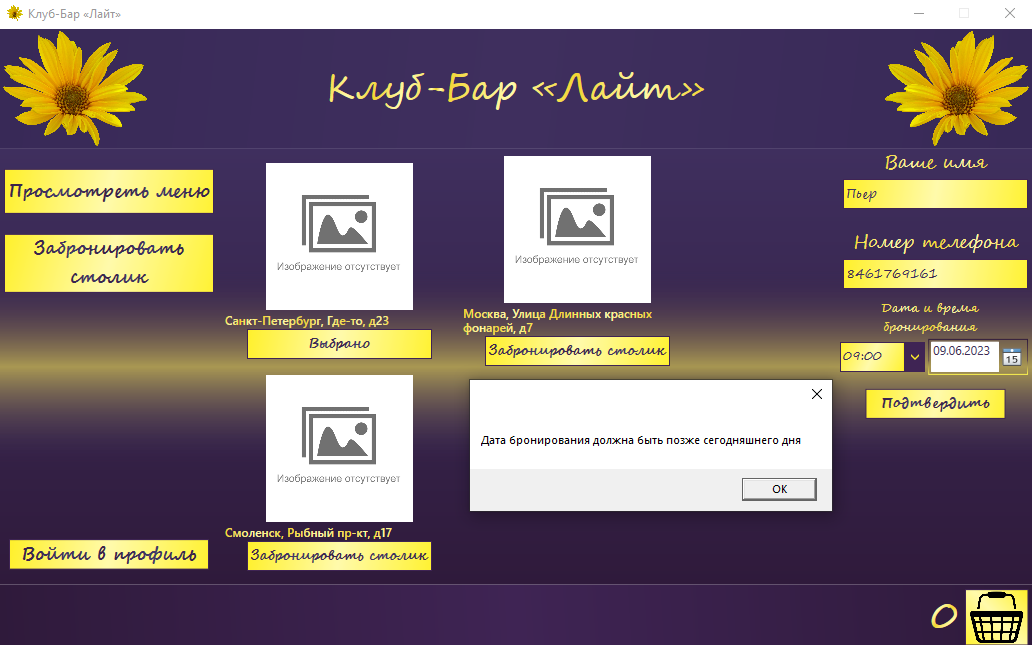


Рисунок 48 – Выбрана сегодняшняя или вчерашняя дата

# ЗАКЛЮЧЕНИЕ

Автоматизация учета продаж способствует сокращению человеческих затрат и усилий, а главное оперативно выдает результат необходимый работнику, а также заменяет большие архивы на структурированное хранение в электронном виде.

В ходе выполнения выпускной квалификационной работы были рассмотрены актуальные вопросы разработки информационных систем.

В первой главе было рассмотрено предназначение автоматизированного учета, проанализированы приложения аналоги и выбрана платформа и язык для реализации автоматизированной системы учета оптовых продаж.

Во второй главе поставлены задачи по разработке конфигурации, спроектирована структура проекта, разработана и протестирована система клиентского приложения.

В итоге, в ходе выполнения дипломной работы была разработана автоматизированная система учета оптовых продаж для организации ООО «Основа». Внедрение данной системы позволит просматривать существующие заказы и отслеживать количество материалов на складе. Заменит большое количество бумажных носителей на структурированное хранение в электронном виде.

Таким образом, все поставленные в работе задачи решены, а цель достигнута.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ И ЛИТЕРАТУРЫ

1. Абрамян, Михаил Visual C# на примерах (+ CD-ROM) / Михаил Абрамян. - М.: БХВ-Петербург, 2012. - 496 c.
2. Биллиг, В. А. Основы объектного программирования на C# (C# 3.0, Visual Studio 2008) / В.А. Биллиг. - М.: Интернет-университет информационных технологий, Бином. Лаборатория знаний, 2017. - 584 c
3. Брайан, Джонсон Основы Microsoft Visual Studio .NET 2003 / Джонсон Брайан. - М.: Русская Редакция, 2020. - 362 c.
4. Васильев, Алексей C#. Объектно-ориентированное программирование / Алексей Васильев. - М.: Питер, 2012. - 320 c.
5. Дунаев, В. В. Базы данных. Язык SQL для студента / В.В. Дунаев. - М.: БХВ-Петербург, 2017. - 288 c.
6. Карвин, Билл Программирование баз данных SQL. Типичные ошибки и их устранение / Билл Карвин. - М.: Рид Групп, 2018. - 336 c
7. Культин Н. Microsoft Visual C# в задачах и примерах; БХВ-Петербург М., 2015. - 320 c
8. Маркин, А. В. Построение запросов и программирование на SQL. Учебное пособие / А.В. Маркин. - М.: Диалог-Мифи, 2014. - 384 c.
9. Постолит Visual Studio .NET: разработка приложений баз данных / Постолит, Анатолий. - М.: СПб: БХВ, 2019. - 544 c.
10. Савин, В. И. Организация складской деятельности / В.И. Савин. - М.: Дело и сервис, 2016. - 544 c.
11. Таран, С. А. Как организовать склад. Практические рекомендации профессионала / С.А. Таран. - М.: Альфа-пресс, 2015. - 240 c.
12. Справочник по Transact-SQL (компонент Database Engine) [Электронный ресурс] // Режим доступа: <https://docs.microsoft.com/ru-ru/sql/t-sql/language-reference?view=sql-server-ver15>
13. Общие сведения о Visual Studio | Microsoft Docs [Электронный ресурс] // Режим доступа: <https://docs.microsoft.com/ru-ru/visualstudio/get-started/visual-studio-ide?view=vs-2019>
14. Обзор языка C# - руководство по C# | Microsoft Docs [Электронный ресурс] // Режим доступа: <https://docs.microsoft.com/ru-ru/dotnet/csharp/tour-of-csharp/>
15. Автоматизация продаж. Системы автоматизации учета и процесса продаж [Электронный ресурс] // Режим доступа: <https://efsol.ru/solutions/sales-automation.html>

# ПРИЛОЖЕНИЯ

Приложение А. Листинг кода формы Авторизации Auth.cs

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.SqlClient;

namespace AutoUchet

{

public partial class Auth : Form

{

public Auth()

{

InitializeComponent();

}

private void button1\_Click(object sender, EventArgs e)

{

this.Close();

}

private void userRole()

{

string Login = "";

string Perm = "";

string UserName = textBox1.Text;

string sql = "SELECT login, password, perm FROM [Users] WHERE login = @un";

SqlConnection conn = new SqlConnection(Model.con);

conn.Open();

SqlCommand command = new SqlCommand(sql, conn);

command.Parameters.AddWithValue("@un", textBox1.Text);

SqlDataReader reader = command.ExecuteReader();

if (reader.HasRows)

{

reader.Read();

if ((string)reader[1] == textBox2.Text)

{

Login = textBox1.Text;

Perm = (string)reader[2];

switch((string)reader[2])

{

case "Admin": this.Hide(); Zakaz f2 = new Zakaz(Login, Perm); f2.Show(); break;

case "Klad": this.Hide(); Sklad f4 = new Sklad(); f4.Show(); break;

default: this.Hide(); Zakaz f3 = new Zakaz(Login, Perm); f3.Show(); break;

}

}

Приложение Б. Листинг кода формы Авторизации Auth.cs

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace AutoUchet

{

public partial class Reg : Form

{

public Reg()

{

InitializeComponent();

}

private void button1\_Click(object sender, EventArgs e)

{

this.Hide();

Auth auth = new Auth();

auth.Show();

}

private void button2\_Click(object sender, EventArgs e)

{

string log = textBox3.Text;

if (textBox4.Text != textBox5.Text)

{

MessageBox.Show("Пароли не совпадают!", "Ошибка!");

return;

}

string perm = "User";

using (SqlConnection connection = new SqlConnection(Model.con))

{

connection.Open();

string sql = "INSERT INTO [dbo].[Users] (login, password, LastN, FirstN, perm) " + "VALUES (@login, @pass, @LN, @FN, @perm)";

SqlCommand command = new SqlCommand(sql, connection);

SqlParameter LNP = new SqlParameter("@LN", textBox1.Text);

SqlParameter FNP = new SqlParameter("@FN", textBox2.Text);

SqlParameter LogP = new SqlParameter("@login", log);

SqlParameter PassP = new SqlParameter("@pass", textBox4.Text);

SqlParameter PermP = new SqlParameter("@perm", perm);

command.Parameters.Add(LNP);

command.Parameters.Add(FNP);

command.Parameters.Add(LogP);

command.Parameters.Add(PassP);

command.Parameters.Add(PermP);

if (textBox3.Text == log)

{

MessageBox.Show("Пользователь с таким логином существует!", "Ошибка!");

return;