Частное учреждение образования

«Колледж бизнеса и права»

ПРОГРАММНОЕ СРЕДСТВО ДЛЯ УЧЕТА ЗАКАЗОВ В ПИЦЦЕРИИ

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту по дисциплине

«Конструирование программ и языки программирования»

КП Т.x

Руководитель проекта (x)

Учащаяся (x)

2022

Содержание

Изм.

Изм.

Изм.

Изм.

Изм.

Изм.

Изм.

Изм.

Лист

Лист

Лист

Лист

Лист

Лист

Лист

Лист

№ докум.

№ докум.

№ докум.

№ докум.

№ докум.

№ докум.

№ докум.

№ докум.

Подпись

Подпись

Подпись

Подпись

Подпись

Подпись

Подпись

Подпись

Дата

Дата

Дата

Дата

Дата

Дата

Дата

Дата

Лист

Лист

Лист

Лист

Лист

Лист

Лист

Лист

3

3

3

3

3

3

3

3

КП Т.993020 ПЗ

КП Т.716007. ПЗ

КП Т.716009. ПЗ

КП Т.716007. ПЗ

КП Т.995020. ПЗ

КП Т.716007. ПЗ

КП Т.716009. ПЗ

КП Т.716007. ПЗ

Разраб.

Разраб.

Разраб.

Разраб.

Разраб.

Разраб.

Разраб.

Разраб.

*x*

*Голубев И.В.*

*Гринь А.М.*

*Голубев И.В.*

*Юськович В.В.*

*Голубев И.В.*

*Гринь А.М.*

*Голубев И.В.*

Провер.

Провер.

Провер.

Провер.

Провер.

Провер.

Провер.

Провер.

x

Т. контр.

Т. контр.

Т. контр.

Т. контр.

Т. контр.

Т. контр.

Т. контр.

Т. контр.

Н. контр.

Н. контр.

Н. контр.

Н. контр.

Н. контр.

Н. контр.

Н. контр.

Н. контр.

Утверд.

Утверд.

Утверд.

Утверд.

Утверд.

Утверд.

Утверд.

Утверд.

*Программное средство для учета заказов в пиццерии*

Программное средство для учета заказов автомобильных комплектующих «Cars»

Лит.

У

Лит.

У

Лит.

У

Лит.

У

Лит.

У

Лит.

У

Лит.

У

Лит.

У

Листов

Листов

Листов

Листов

Листов

Листов

Листов

Листов

62

50

55

50

81

50

55

50

КБП

КБП

КБП

КБП

КБП

КБП

КБП

КБП

[Введение 4](#_Toc41922226)

[1 Объектно-ориентированный анализ и проектирование систем 5](#_Toc41922227)

[1.1 Сущность задачи 5](#_Toc41922228)

[1.2 Проектирование модели 6](#_Toc41922229)

[2 Вычислительная система 9](#_Toc41922230)

[2.1 Требования к аппаратным и операционным ресурсам 9](#_Toc41922231)

[2.2 Инструменты разработки 9](#_Toc41922232)

[3 Проектирование задачи 11](#_Toc41922233)

[3.1 Требования к приложению 11](#_Toc41922234)

[3.2 Концептуальный прототип 11](#_Toc41922235)

[3.3 Организация данных 12](#_Toc41922236)

[3.4 Функции: логическая и физическая организация 15](#_Toc41922237)

[3.5 Проектирование справочной системы приложения 16](#_Toc41922238)

[4 Описание программного средства 17](#_Toc41922239)

[4.1 Общие сведения 17](#_Toc41922240)

[4.2 Функциональное назначение 17](#_Toc41922241)

[4.3 Входные и выходные данные 18](#_Toc41922242)

[5 Методика испытаний 19](#_Toc41922243)

[5.1 Технические требования 19](#_Toc41922244)

[5.2 Функциональное тестирование 19](#_Toc41922245)

[6 Применение 30](#_Toc41922246)

[6.1 Назначение программы 30](#_Toc41922247)

[6.2 Условия применение 30](#_Toc41922248)

[6.3 Справочная система 30](#_Toc41922249)

[Заключение 33](#_Toc41922266)

[Список информационных источников 34](#_Toc41922267)

[Приложение А Текст программных модулей 35](#_Toc41922268)

[Приложение Б Формы входных и выходных документов 54](#_Toc41922269)

[Приложение В Результаты работы программы 55](#_Toc41922270)

Введение

Пицца (итал. Pizza) — традиционное итальянское блюдо в виде круглой дрожжевой лепёшки, выпекаемой с уложенной сверху начинкой из томатного соуса, сыра и зачастую других ингредиентов, таких как мясо, овощи, грибы и других продуктов. Небольшую пиццу иногда называют пиццеттой. Повар, специализирующийся на приготовлении пиццы, — пиццайоло. В мире различных пицц и пиццерий, подбор любого блюда является непростой задачей, на вкус блюда влияют такие факторы как: ингредиенты, квалифицированные повара, скорость доставки, влияющая на температуру, и качественное оборудование

Для быстрого и удобного заказа любимой пиццы для покупателей необходима продуманная система выборки и заказа интересующего его набора пицц. Всё это должно быть реализовано в одном месте. Ведение базы данных для хранения списка пицц и работа с ней является наилучшим из возможных решений.

Целью курсового проекта на тему Программное средство для учета заказов в «Доминос» является создания программного средства “SuperPizzeria.exe”, предназначенной для автоматизации учёта, поиска и заказа всевозможных пицц, получение всей необходимой информации для пользователя для составления своего заказа, работа со скидками.

Решение поставленных задач отражено в пояснительной записке, которая состоит из шести разделов и содержит необходимую и достаточную информацию по использованию данного программного средства.

В первом разделе «Объектно-ориентированный анализ и проектирование системы» рассматривается сущность и актуальность поставленной задачи, описание существующих аналогов, проектирование модели, отображающей функциональную структуру объекта.

Второй раздел «Вычислительная система» содержит описание вычислительной системы: технические характеристики персонального компьютера, требования, которые будут предъявляться к персональному компьютеру, описание операционной системы, языка реализации и языка моделирования.

Третий раздел «Проектирование задачи» включает требования к программному средству, концептуальный прототип, логическую и физическую организацию данных в контексте среды разработки, описание функций пользователя в рамках среды разрабатываемого программного средства, проектирование справочной системы программного средства.

Четвертый раздел «Описание программного средства» отражает общие сведения о программе, функциональное назначение, структуру входных и выходных данных.

В пятом разделе «Методика испытаний» рассматриваются требования к техническим средствам для проведения испытания, требования к характеристикам программы применительно к условиям эксплуатации, требования к информационной и программной совместимости. Также описывается порядок проведения функционального тестирования.

Шестой раздел «Применение» содержит информацию, необходимую в процессе эксплуатации программного средства: его назначение, условия применения, а также приводится описание справочной системы.

В заключении описывается выполнение поставленной задачи, степень соответствия проектных решений задания, причины несоответствия, если таковые имеются.

В приложении А представлен текст программных модулей и в приложении Б – формы выходных документов.

Графическая часть представлена диаграммами вариантов использования, классов, последовательности, деятельности и компонентов. Диаграмма последовательности сделана для функции добавления нового арендатора. Диаграмма деятельности сделана для функции добавления и редактирования информации по арендаторам.

1. Объектно-ориентированный анализ и проектирование систем
   1. Сущность задачи

Предметной областью решаемой задачи является пиццерия. Объектом решаемой задачи является автоматизация рабочего места сотрудника и администрации, упрощение поиска пицц и составления заказа для покупателей пиццерия.

Для налаженной работы составления заказов, его сотрудников и покупателей необходимо участие администратора этой пиццерии. Для четкой координации работы, ориентирования среди пицц, времени доставки, пополнения склада ингредиентов, нужен специалист, владеющий всей общей информацией.

Специалисту необходимо знать точный список пицц, их названия, состав, цену ингредиентов.

Для этого у администратора ведется список пицц. В нем указывается вся необходимая информация: номер, название, цена, состав, включение в меню. Данный список представлен на рисунке 1.



Рисунок 1

Исходя из анализа предметной области можно выделить следующие задачи, подлежащие автоматизации:

* ведение базы данных, содержащей информацию о пиццах, а именно: название, состав, цена всех ингредиентов, стоимость самой пиццы, включение в меню;
* учет заказов;
* сортировка и поиск данных по стоимости;
* добавление новых пицц;
* удаление, редактирование уже существующих пицц;
* формирование заказа для доставки;
* формирование корзины пицц.

Аналоги данного приложения широко распространены в интернете, их единственным преимуществом является нахождение в интернете, в то время как разрабатываемое приложение имеет лишь десктоп версию, потому обязательно необходимо иметь доступ к компьютеру.

* 1. Проектирование модели

Цель моделирования данных состоит в обеспечении разработчика информационной системы концептуальной схемой базы данных в форме одной или нескольких локальных моделей, которые относительно легко могут быть отображены в любую систему без данных [3].

Наиболее распространенным средством моделирования данных является диаграмма «Сущность-связи» (ERD). С их помощью определяется важные для предметной области объекты (сущности), их свойства (атрибуты) и отношения друг с другом (связи). ERD непосредственно используются для проектирования реляционных баз данных. Нотация ERD была впервые введена П. Ченном и получила дальнейшее развитие в работах Баркера.

Диаграмма «Сущность-связи» представлена на рисунке 2.

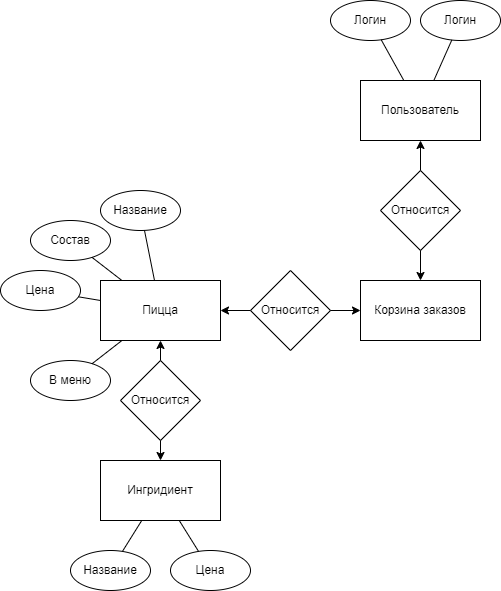


Рисунок 2

Исходя из предметной области можно выделить следующие сущности разработки: «Пользователи», «Пиццы», «Ингредиенты», «Заказы», «Единицы корзины», «Скидки».

Для сущности «Пользователи» атрибутами будут являться:

* персональный номер;
* логин;
* пароль;
* статус.

Для сущности «Пиццы» атрибутами будут являться:

* персональный номер;
* название;
* состав;
* включение в меню;
* список ингредиентов.

Для сущности «Единицы корзины» атрибутами будут являться:

* персональный номер;
* персональный номер пользователя;
* персональный номер пиццы;
* количество.
* персональный номер заказа.

Для сущности «Ингредиенты» атрибутами будут являться:

* персональный номер;
* название;
* цена.

Для сущности «Заказы» атрибутами будут являться:

* персональный номер;
* дата заказа;
* персональный номер скидки;
* статус.

Для сущности «Скидки» атрибутами будут являться:

* персональный номер;
* название;
* множитель скидки.

Суть диаграммы вариантов использования заключается в том, что проектируемая система представляется в виде множества сущностей или актеров, взаимодействующих с системой с помощью, так называемых вариантов использования.

К «extend» относится входная информация:

* о количестве пицц;
* о пиццах;

К «include» относится:

* формирование отчета заказов;
* доставка автомобильных пицц;
* контроль заказов покупателя;
* ведение и работа с базой данных;

Диаграмма вариантов использования представлена на листе 1 графической части.

Диаграмма классов служит для представления статической структуры модели системы в терминологии классов объектно-ориентированного программирования. Диаграмма может отражать различные взаимосвязи между отдельными сущностями предметной области, такими как объекты и подсистемы, а также описывать их внутреннюю структуру и типы отношений. Для организации программного средства для учета заказов пицц «Доминос» в программном средстве реализованы классы: «MainWindow», «Profiler», «Registration», «Administration», «Ingridients», «Authorization», «OrdersAccounting», «AuthReg», «Functions». Описанные классы представлены на листе 2 графической части.

При моделировании поведения проектируемой или анализируемой системы возникает необходимость детализировать особенности алгоритмической и логической реализации выполняемых системой операций. Для моделирования процесса выполнения операций в языке UML используются так называемые диаграммы деятельности.

Диаграммы деятельности – частный случай диаграмм состояний. Основная цель использования таких диаграмм – визуализация особенностей реализации операций классов, когда необходимо представить алгоритмы их выполнения. Диаграмма деятельности представлена на листе 3 графической части.

Временной аспект поведения имеет существенное значение при моделировании синхронных процессов, описывающих взаимодействия объектов. Именно для этой цели и используются диаграммы последовательности, в которых ключевым моментом является динамика взаимодействия объектов во времени. Например, сотрудник переходит с вкладки на вкладку и добавляет данные, что является временным аспектом поведения сотрудника. Он включает программу, вводит данные, сохраняет данные – последовательность действий. Диаграмма последовательности представлена на листе 4 графической части.

Диаграмма компонентов описывает объекты реального мира – компоненты программного обеспечения. Эта диаграмма позволяет определить архитектуру разрабатываемой системы, установив зависимости между программными компонентами, в роли которых может выступать исходный, бинарный и исполняемый код.

Определим компоненты для данного программного средства и построим диаграмму компонентов. Главным исполняемым файлом является файл «SuperPizzeria.exe». Этот файл является результатом компиляции проекта «SuperPizzeria.sln». Компонент «SuperPizzeria.sln» реализован в среде разработки Visual Studio 2019. В состав компонента «SuperPizzeria.sln» входят следующие файлы:

* файлы проекта с расширениями \*.sln, \*.cs, \*.resx, \*.xaml, \*.cd;
* файлы отчетов \*.docx, реализованные в текстовом процессоре MS Word;
* файлы базы данных «Pizzeria.mdf» и «Pizzeria\_log.ldf», созданные в MS SQL Server.

Вид диаграммы компонентов для данной проектируемой системы представлен в графической части на листе 5.

1. Вычислительная система
   1. Требования к аппаратным и операционным ресурсам

Минимальными требованиями к аппаратному и программному обеспечению, необходимыми для корректной работы программы являются:

* операционная система Windows 7 и выше;
* процессор Intel Celeron 1800 МГц и выше;
* объем оперативной памяти 1024 МБ и более;
* мышь;
* монитор;
* принтер;
* клавиатура.

Наиболее удобной операционной системой для проведения испытаний является Windows 11, так как она ориентирована на максимальное использование всех возможностей персонального компьютера, сетевых ресурсов и обеспечение комфортных условий работы.

На компьютере должны быть установлены программы Microsoft SQL Server 2019 и Microsoft Office Word 2021.

* 1. Инструменты разработки

Ниже описаны инструменты разработки, которые будут использоваться для написания программного средства.

Персональный компьютер со следующей аппаратной конфигурацией:

* процессор Intel Core i5-1035G1 2600 МГц;
* оперативная память DDR4 16 ГБ;
* встроенная видеокарта Intel UHD Graphics 630;
* винчестер SSD 512 ГБ;
* клавиатура и мышь.

Программное обеспечение:

* операционная система Windows 11;
* среда программирования Visual Studio 2019 c использованием интерфейса программирования приложений API Windows Presentation Foundation;
* язык программирования C#;
* система управления базами данных (СУБД) Microsoft SQL Server 2019;
* интегрированная среда SQL Server Management Studio (SSMS) 2019;
* программная платформа Microsoft .NET Framework 6.0;
* офисный пакет приложений Microsoft Office (MS Office) 2021;
* программа для построения диаграмм Drawio.io;

Операционная система – это набор управляющих программ, предназначенных для управления ресурсами вычислительной системы как единого комплекса, другими словами, операционная система – это набор программного обеспечения, который обеспечивает работу компьютера.

При разработке программного средства использовалась операционная система Windows 11, так как на данный момент эта операционная система является самой распространённой операционной системой. В Windows 11 были исправлены практически все недостатки предыдущих операционных систем. Аппаратные требования Windows 11 скромнее, она способна работать даже на маломощных компьютерах и ещё добавлено множество функций, существенно облегчающих работу за компьютером.

Visual Studio 2019 – это лучший инструмент для разработки любого приложения под любую платформу. Система управления версиями в этом выпуске делает разработку гибкой, а совместную работу – эффективной [11].

Windows Presentation Foundation – это фреймворк пользовательского интерфейса, который создает компьютерные клиентские приложения. Платформа разработки WPF поддерживает широкий набор функций разработки приложений, включая модель приложения, ресурсы, элементы управления, графику, макет, привязку данных, документы и безопасность. [12].

Язык программирования С# – это простой, современный и объектно-ориентированный язык, который предоставляет современным разработчикам гибкость и возможности для создания программного обеспечения, которое будет работать не только сегодня, но и будет применяться в течение многих лет в будущем. [7].

СУБД Microsoft SQL Server 2019 – основной продукт Microsoft по обработке данных. Версия 2018 – это наиболее значительный прорыв в истории платформ данных Microsoft: более быстрые транзакции и запросы, ценные сведения на любом устройстве, расширенная аналитика, новые технологии безопасности и новые варианты использования при применении гибридного облака. SQL Server 2019 предоставляет передовые функции для решения задач высокой важности за счет выполнения операций в памяти и встроенной операционной аналитики [10].

SQL Server – это хорошо масштабируемый, полностью реляционный, быстродействующий многопользовательский сервер баз данных масштаба предприятия, способный обрабатывать большие объемы данных для клиент-серверных приложений [10].

SQL Server Management Studio (SSMS) – это интегрированная среда для управления любой инфраструктурой SQL: от SQL Server до базы данных SQL Azure. SSMS предоставляет средства для настройки, наблюдения и администрирования экземпляров SQL. С помощью SSMS можно развертывать, отслеживать и обновлять компоненты уровня данных, используемые вашими приложениями, а также создавать запросы и скрипты [10].

.NET Framework – [программная платформа](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%B0%D1%8F_%D0%BF%D0%BB%D0%B0%D1%82%D1%84%D0%BE%D1%80%D0%BC%D0%B0#%D0%BA%D1%80%D0%BE%D1%81%D1%81%D0%BF%D0%BB%D0%B0%D1%82%D1%84%D0%BE%D1%80%D0%BC%D0%B5%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5), выпущенная компанией [Microsoft](https://ru.wikipedia.org/wiki/Microsoft) в [2002 году](https://ru.wikipedia.org/wiki/2002_%D0%B3%D0%BE%D0%B4). Основой платформы является общеязыковая среда исполнения [Common Language Runtime (CLR)](https://ru.wikipedia.org/wiki/Common_Language_Runtime), которая подходит для разных языков программирования. Функциональные возможности CLR доступны в любых языках программирования, использующих эту среду [6].

Microsoft Office 2021 – офисный пакет приложений, созданных корпорацией Microsoft для операционных систем Microsoft Windows [9].

Draw.io - используется для создания таких типов диаграмм, как блок-схемы, Организационные диаграммы, Поэтажные планы, Сетевые диаграммы, Диаграммы UML, Интеллектуальные карты и многое другое. Он также обычно используется для таких сценариев, как Отображение процессов и Визуальная совместная работа [8].

1. Проектирование задачи
   1. Требования к приложению

Разрабатываемое приложение должно иметь понятный и удобный в использовании интерфейс, чтобы взаимодействие между программой и пользователем было максимально упрощено.

Согласно общим требованиям, графический интерфейс разрабатываемого приложения должен:

* ориентироваться на пользователя, который общается с программой на внешнем уровне взаимодействия;
* сохранять стандартизированное назначение и местоположение на экране графических объектов, работающих в среде Windows.

Интерфейс программного приложения будет разрабатываться с учетом общих требований к пользовательскому интерфейсу.

Существуют общие требования, которые предъявляются к программным приложениям:

* соответствие стандартам организации интерфейса: использование многооконного подхода, реализация управления работой программного приложения с помощью элементов управления;
* требования к выбранной цветовой схеме рабочего экрана;
* выполнение одной функции с помощью разных элементов управления.

Пользователю необходимо установить пакеты SQL Server 2019 и Microsoft Office 2021 для работы с базами данных и переноса отчетов в документ Microsoft Office Word 2021.

Для обучения пользователя необходимо разработать справочную систему, в которой должны быть раскрыты все аспекты работы с программой, возможные трудности, возникшие во время работы и пути их решения.

Для удобства работы пользователя с программным средством необходимо при разработке форм придерживаться единого стиля оформления. Формы не должны быть перегружены излишней информацией или содержать информацию, не относящуюся к данной форме. Также необходимо предусмотреть защиту данных от удаления и изменения, а также от ввода некорректных данных. В случае ввода некорректных данных или попытке совершить запрещенные действия, пользователь должен быть проинформирован о своих действиях с помощью диалоговых окон.

* 1. Концептуальный прототип

Концептуальный прототип состоит из описания внешнего пользовательского интерфейса, а именно, элементов управления.

При создании данного приложения важную роль играют формы, так как они являются основным средством работы с пользователем. Разрабатываемое приложение будет содержать несколько форм.

Все формы будут содержать стандартные пользовательские элементы управления.

В рабочем режиме программы, пользователю, для удобной навигации, будет предоставлено меню.

На главной странице расположен весь необходимый для пользователя функционал.

Посередине располагается меню пиццерии, где находятся все доступные для заказа пиццы.

Пользователь может добавить пиццу в свой заказ путём нажатия на кнопку “Добавить в корзину”. Выбор количества пицц поможет пользователю в кратчайшие сроки добавить как можно большее количество пицц.

Кнопка “Конструктор пиццы” позволяет пользователю создать свою собственную пиццу из имеющихся в пиццерии ингредиентов.

Кнопка “Скидки” предоставит пользователю всю информацию о актуальных скидках в пиццерии.

Кнопка “Справка” покажет пользователю всю информацию, необходимую ему для пользования приложением с комфортом.

Кнопка “Профиль” переведёт пользователя на страницу с его профилем, где он увидит список всех совершённых им заказов и сможет взаимодействовать с сотрудниками пиццерии путём отклонения и принятия предложенных ими скидок.

Все страницы главной формы будут иметь схожий интерфейс. Макет главной формы для вкладки «Магазин» представлен на рисунке 3.

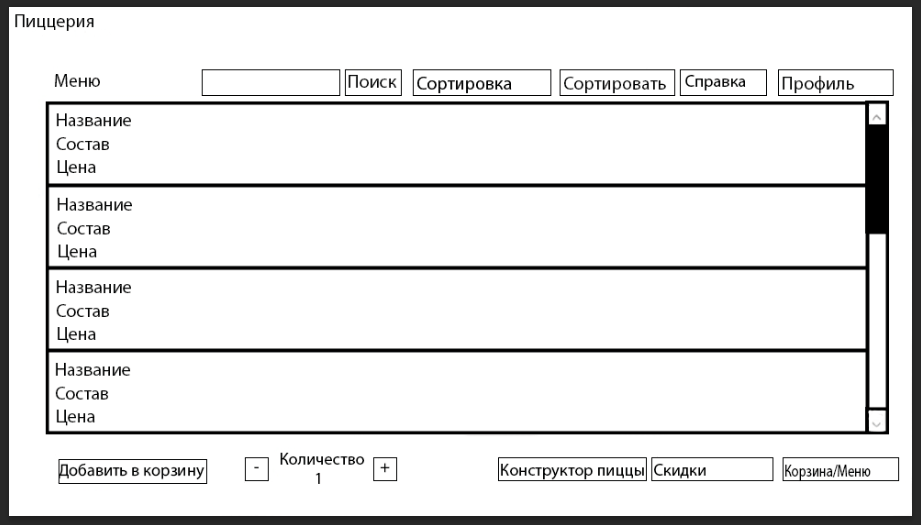


Рисунок 3

* 1. Организация данных

Организация данных подразумевает создание модели данных, главными элементами которой являются сущности и их связи.

Реляционная модель основана на математическом понятии отношения, представлением которого является таблица. В реляционной модели отношения используются для хранения информации об объектах, представленных в базе данных. Отношение имеет вид двухмерной таблицы, в которой строки соответствуют записям, а столбцы – атрибутам. Каждая запись должна однозначно характеризоваться в таблице. Для этого используют первичные и внешние ключи. Достоинством реляционной модели является простота и удобство физической реализации.

Структура базы данных разрабатываемого программного средства включает девять таблиц. Описание таблиц приводится в таблицах 1-5.

Таблица «Единицы корзины» хранит информацию о корзинах покупателей, структура которой представлена в таблице 1.

Таблица 1 – Структура таблицы «Единицы корзины»

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Тип  поля | Размер поля,  байт | Описание поля |
| Id | int | 4 | Идентификатор Единицы корзины |
| Pizza | int | 4 | Идентификатор таблицы «Пиццы» |
| Amount | int | 4 | Количество пицц |
| UserId | int | 4 | Идентификатор таблицы «Пользователи» |
| Order | int | 4 | Идентификатор таблицы «Заказы» |

Таблица «Ингредиенты» хранит названия ингредиенты, структура которой представлена в таблице 2.

Таблица 2 – Структура таблицы «Ингредиенты»

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Тип  поля | Размер поля,  байт | Описание поля |
| ID | int | 4 | Идентификатор ингредиента |
| Name | varchar | 50 | Название ингредиента |
| Cost | int | 4 | Стоимость ингредиента |

Таблица «Заказы» хранит информацию о деталях, которые были добавлены в корзину, структура которой представлена в таблице 3.

Таблица 3 – Структура таблицы «Заказы»

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Тип  поля | Размер поля,  байт | Описание поля |
| ID | int | 4 | Идентификатор заказа |
| DateOfOrder | datetime | 8 | Дата и время заказа |
| Sale | int | 4 | Идентификатор скидки |
| UserId | int | 4 | Идентификатор пользователя |
| Status | Int | 4 | Статус заказа |

Таблица «Пиццы» хранит информацию обо всех пиццах, структура которой представлена в таблице 4.

Таблица 4.1 – Структура таблицы «Пиццы»

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Тип  поля | Размер поля,  байт | Описание поля |
| Id | int | 4 | Идентификатор пиццы |

Таблица 4.2 – Продолжение таблицы 4.1

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Тип  поля | Размер поля,  байт | Описание поля |
| Name | varchar | 50 | Название пиццы |
| Ingridientslist | varchar | 50 | Список ингредиентов |
| IsOnMenu | bit | 1 | Включение в меню |

Таблица «Скидки» хранит информацию скидок, структура которой представлена в таблице 5.

Таблица 5 – Структура таблицы «Скидки»

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Тип  поля | Размер поля,  байт | Описание поля |
| Id | int | 4 | Идентификатор скидки |
| Summary | varchar | 50 | Описание скидки |
| Multiplier | varchar | 50 | Множитель скидки |

Таблица «Пользователи» хранит информацию пользователей для входа в программу, структура которой представлена в таблице 5.

Таблица 5 – Структура таблицы «Пользователи»

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Тип  поля | Размер поля,  байт | Описание поля |
| Id | int | 4 | Идентификатор пользователя |
| Login | varchar | 50 | Логин |
| Password | varchar | 50 | Пароль |
| Status | int | 4 | Статус пользователя |
| Cart | Int | 4 | Идентификатор корзины |

Физическая схема базы данных представлена на рисунке 4.

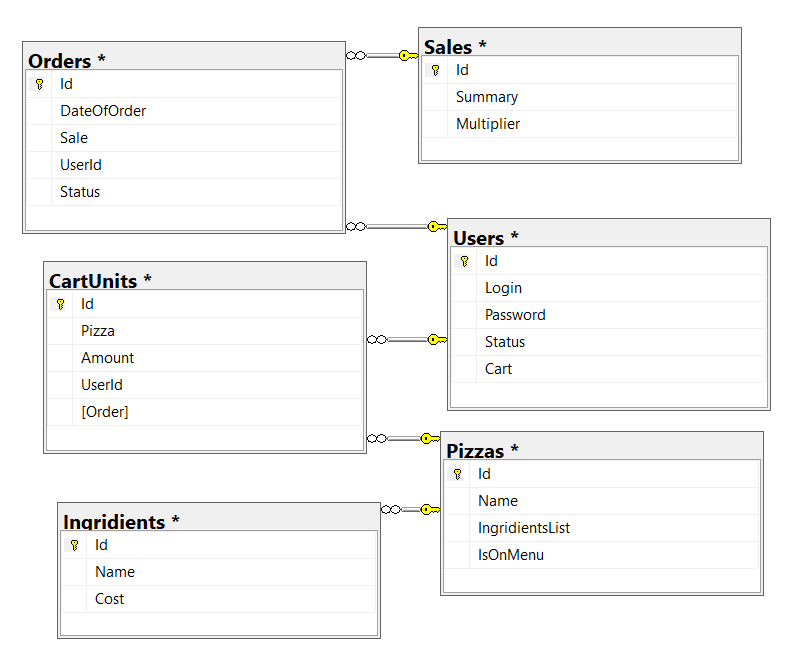


Рисунок 4

* 1. Функции: логическая и физическая организация

На основании диаграммы вариантов использования в программном средстве «SuperPizzaria.exe» реализована функция поиска пицц по заданным покупателем алгоритмам поиска.

Рассмотрим данные функции программы.

Для того, чтобы выполнить поиск по комплектам запчастей необходимо перейти в главное окно «Пиццерия» и заполнить поля данными. После того, как покупатель нажмет на кнопку «Поиск» будет автоматически выполнен поиск товаров по заданным фильтрам. Код функции представлен ниже.

private void Search\_Click(object sender, RoutedEventArgs e)

{

string text = SearchQuery.Text;

if (text.Length == 0)

{

db.Pizzas.Where(i => i.IsOnMenu).Load();

pizzasList = db.Pizzas.Local.ToList();

}

else

{

Regex nameReg = new Regex(@"[а-я]+");

string pizzaName = "";

foreach (Match match in nameReg.Matches(text.ToLower()))

{

pizzaName += match + " ";

}

List<Pizza> newPizzas = new List<Pizza>();

foreach (Pizza p in db.Pizzas.Where(p => p.Name.ToLower().IndexOf(pizzaName.ToLower()) != -1))

{

newPizzas.Add(p);

}

pizzasList = newPizzas;

}

Menu.ItemsSource = null;

Menu.ItemsSource = pizzasList;

Menu.Items.Refresh();

}

Для того, чтобы пользователь мог добавить пиццу при помощь конструктора пицц, ему необходимо нажать на кнопку “Конструктор пицц”, код реализации данной функции представлен ниже.

private void AddPizzaUser\_Click(object sender, RoutedEventArgs e)

{

if (IngridientsList.SelectedItems.Count < 2 || PizzaName.Text.Length < 5)

{

MessageBox.Show("Некорректные данные для создания пиццы", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

return;

}

var ings = IngridientsList.SelectedItems;

foreach (Ingridient ing in ings)

{

ingsList.Add(ing.Id);

}

Pizza pizza = new Pizza() { Name = Convert.ToString(PizzaName.Text), IngridientsList = Functions.ArrayListToString(ingsList), IsOnMenu = true };

PizzaName.Text = "";

db.Pizzas.Add(pizza);

SaveChanges\_Click(sender, e);

BackUser\_Click(sender, e);

}

Полный текст программы представлен в приложении А.

* 1. Проектирование справочной системы приложения

Для эффективной работы пользователя с приложением необходимо обеспечить его качественной справочной системой, в которой должны быть приведены методы и приемы работы с приложением, включающие данные о том, что произойдет после нажатия на определенную кнопку или при выборе пункта меню, сведения о том, какую информацию и в каком виде следует вводить в соответствующие поля, каким образом можно работать.

Справочная система будет представлена в виде «Окно-уведомление» в приложении, содержащая информацию по эксплуатации программного средства.

В подразделе «Инструкция использования» будут описаны все функции главного окна приложения, как правильно выполнять поиск деталей, как добавить(удалить) существующую пиццу, как изменить ранее введенные данные.

В подразделе «Экспорт отчетов» будет описано как сделать отчет и его содержание.

1. Описание программного средства
   1. Общие сведения

Программное средство «Cars» разрабатывалось с целью автоматизация рабочего места сотрудника и упрощение поиска запчастей для покупателей «Cars» и увеличения количества годовых продаж предприятия за счет уменьшения времени поиска нужных запчастей, как во время покупки, так и во время продажи благодаря простому, лаконичному интерфейсу и функциональным возможностям приложения. Установка данного программного обеспечения позволяет проделывать сложные операции без ошибок. Данная программа была разработана на персональном компьютере со следующей конфигурацией:

* процессор Intel Core i5-1035G1 2600 МГц;
* оперативная память DDR4 16 ГБ;
* встроенная видеокарта Intel UHD Graphics 630;
* винчестер SSD 512 ГБ;
* клавиатура и мышь.

Программа создана в средстве разработки Microsoft Visual Studio 2019 на языке программирования C# в операционной системе Windows 11. Она может работать в средах операционных систем семейства Microsoft Windows, начиная с Windows 7.

В программе присутствует база данных «Pizzeria.mdf». Программа не требовательна к системным ресурсам, также проста в использовании и не требует специальных навыков при работе. Для работы данного программного средства необходима предварительная установка и настройка следующих программных продуктов:

* интегрированная среда SQL Management Studio 2019;
* среда разработки Microsoft Visual Studio 2019;
* платформа Microsoft Net Framework версии 6.0.

В состав программных входят:

* программа «SuperPizzeria.exe»;
* база данных «Pizzeria.mdf»;
* журнал транзакций «Pizzeria\_log.ldf»;

Объем установленного приложения не превышает 200 МБ, файлов – 100 МБ.

Программное средство требует логина и пароля.

* 1. Функциональное назначение

Программное средство создано для упрощения работы администрации и сотрудников пиццерии, также оно облегчает использование пользователем приложения пиццерии, предлагаю ему удобный и лаконичный интерфейс.

Программное средство позволяет вести учет заказов, а также управлять меню при помощи лаконичного интерфейса.

Программа использует стандартные элементы управления, такие как кнопки, меню, списка, поля ввода, что обеспечивает единство интерфейса системы и программного средства.

В работе программного средства предусмотрены некоторые ситуации, которые должны предупреждать пользователя, чтобы он выполнял все необходимые требования по эксплуатации программы. Для этого существуют сообщения системы.

Таким образом, программа может применяться в реальных условиях, представляя собой достаточно удобный помощник.

* 1. Входные и выходные данные

Входными являются данные, введённые сотрудником в таблицы базы данных: «Пиццы», «Пользователи», «Заказы», «Единицы корзины», «Скидки», «Ингредиенты».

Входными данным при добавлении данных о новом покупателе будут являться:

* логин;
* пароль;
* статус.

Входными данными при добавлении данных о пиццы являются:

* название;
* список ингредиентов;
* включение в меню.

Входными данными при добавлении данных о новом заказе являются:

* дата заказа;
* скидка;
* пользователь;
* статус.

Входными данными при добавлении данных о новом элементе корзины являются:

* пицца;
* количество;
* пользователь;
* заказ.

Входными данными при добавлении данных о новом ингредиенте являются:

* название;
* цена.

Вся вносимая в программное средство информация хранится в локальной базе данных.

Выходные данные – это сформированные списки, которые необходимы для формирования операций по магазину.

Отчет обо всех заказах представлен в приложении Б на рисунке Б.1.

1. Методика испытаний
   1. Технические требования

Основными минимальными требованиями, выдвигаемыми к аппаратному обеспечению персонального компьютера, являются:

* процессор Intel Core i5-1035G1 2600 МГц;
* оперативная память DDR4 16 ГБ;
* встроенная видеокарта Intel UHD Graphics 630;
* винчестер SSD 512 ГБ;
* клавиатура и мышь.

Для работы с программой необходимо наличие монитора, клавиатуры и мыши.

Компьютер должен работать под управлением операционной системы, начиная с Windows 7 и выше. Необходимо наличие Microsoft Office 2019, SQL Server Management Studio 2019, Microsoft SQL Server 2019 и Microsoft .NET Framework 6.0.

* 1. Функциональное тестирование

Функциональное тестирование – это тестирование функций приложения на соответствие требованиям и проводится для выявления неполадок и недочетов программы на этапе ее сдачи в эксплуатацию.

Проведем тестирование проверки каждого пункта меню с целью проверки всех функций.

Тестирование программы будет производиться последовательно, переходя из одной части программы в другую. Во время теста будут проверяться все действия с программой, навигация пунктам меню, которые может произвести пользователь. После чего, все собранные и найденные ошибки будут исправлены [5].

В таблицах 6 представлены тест-кейсы, подготовленные для проведения функционального тестирования.

Таблица 6.1 – Функция тестирования добавления новой пиццы и включение её в меню.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Модуль / Функция | Шаги выполнения | Ожидаемый результат | Фактический результат |
| 1 | Добавление новой пиццы в меню. | 1. Авторизоваться как администратор.  2. Нажать на кнопку «Добавление пиццы».  3. Заполняем поле «Название пиццы» данными «Мамамия».  4. Выбрать ингредиенты «Трюфели курица картофель».  5. Нажать на кнопку «Добавить пиццу».  6. Нажить галочку «В меню».  7. Нажать на кнопку «Сохранить изменения». | Добавление в базу данных новой пиццы и включение её в меню. | Добавление в базу данных новой пиццы и включение её в меню. Результат отображен на рисунках 6.1 и 6.2. |

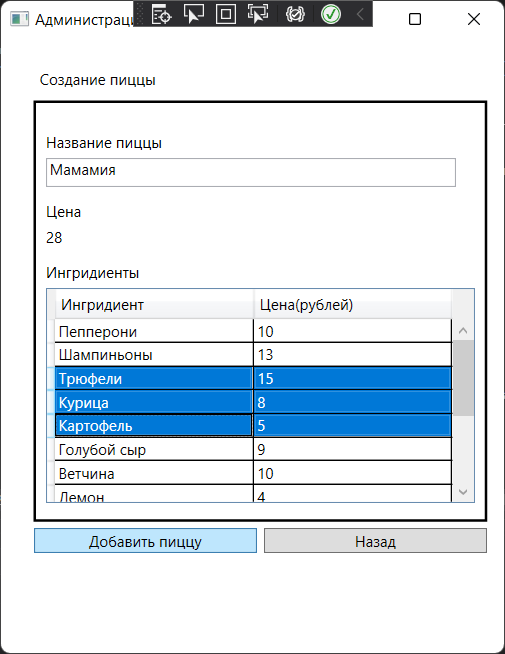


Рисунок 6.1

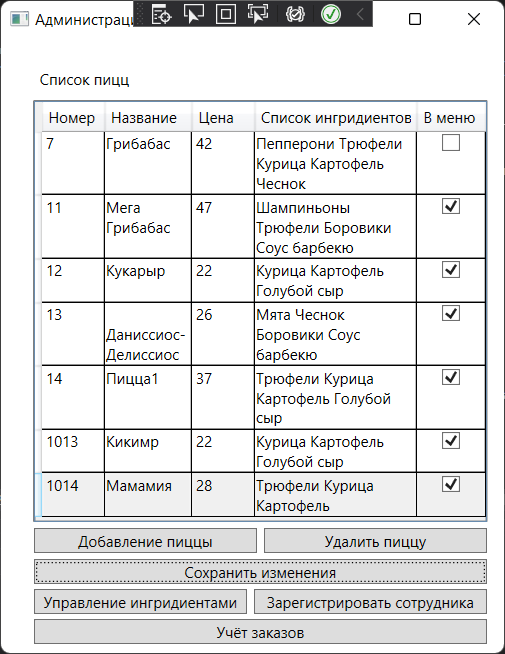


Рисунок 6.2

Таблица 6.2 – Функция удаления пиццы из базы данных.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Модуль / Функция | Шаги выполнения | Ожидаемый результат | Фактический результат |
| 2 | Проверка возможности удаления пицц базы данных. | 1. Авторизоваться как администратор.  2. Выбрать пиццу в меню «Грибабас»  3. Нажать на кнопку «Удалить пиццу».  4. Нажать на кнопку «Сохранить изменения». | Удаление пиццы из базы данных. | После нажатия на кнопку «Сохранить изменения» пицца пропала из списка и была удалена из базы данных. Результат представлен на рисунке 6.3 |

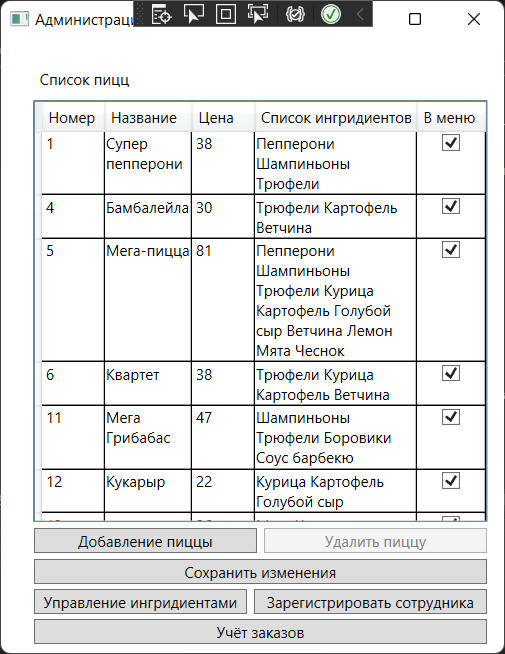


Рисунок 6.3

Таблица 6.3 – Функция тестирования добавления нового ингредиента в базу.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Модуль / Функция | Шаги выполнения | Ожидаемый результат | Фактический результат |
| 3 | Проверка на добавление нового ингредиента в базу приложения. | 1. Авторизоваться как администратор.  2. Нажать на кнопку «Управление ингредиентами».  3. Нажать на кнопку «Добавить».  4. Внести необходимые данные: Название: «Ананасы», Цена: «5»  5. Нажать на кнопку «Сохранить» | Добавление нового ингредиента в базу для дальнейшего использования. | После нажатия на кнопку «Сохранить» были внесены изменения в список ингредиентов. Результаты представлены на рисунке 6.4. |

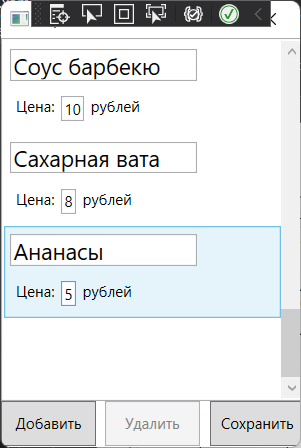


Рисунок 6.4

Таблица 6.4 – Функция тестирования удаления ингредиента из базы данных.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Модуль / Функция | Шаги выполнения | Ожидаемый результат | Фактический результат |
| 4 | Проверка на удаление ингредиента из базы данных. | 1. Авторизоваться как администратор.  2. Нажать на кнопку «Управление ингредиентами».  3. Выбрать ингредиент: «Сахарная вата».  4. Нажать на кнопку «Удалить».  5. Нажать на кнопку «Сохранить». | Удаление выбранного ингредиента из базы данных. | После нажатия на кнопку ингредиент был удалён из базы данных. Результаты работы представлены на рисунке 6.5. |

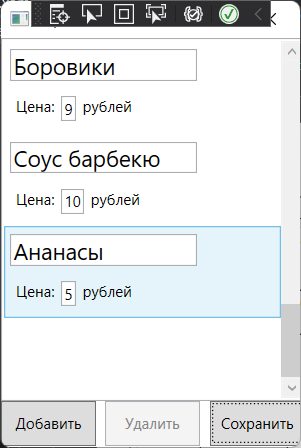


Рисунок 6.5

Таблица 6.5 – Функция тестирования добавления нового сотрудника в базу данных.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Модуль / Функция | Шаги выполнения | Ожидаемый результат | Фактический результат |
| 4 | Проверка на удаление ингредиента из базы данных. | 1. Авторизоваться как администратор.  2. Нажать на кнопку «Зарегистрировать сотрудника».  3. Ввод данных в поля: Логин = Emp1, Пароль = 1  4. Нажать на кнопку «Зарегистрироваться». | Добавление нового сотрудника в базу дынных приложения. | После нажатия на кнопку в базу данных добавляется новый пользователь со статусом сотрудника. Результат работы представлен на рисунке 6.6 |

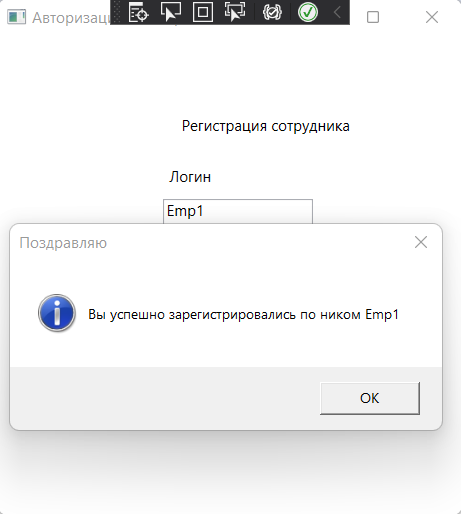


Рисунок 6.6

Таблица 6.6 – Функция тестирования добавления нового сотрудника в базу данных с некорректными данными.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Модуль / Функция | Шаги выполнения | Ожидаемый результат | Фактический результат |
| 6 | Проверка добавления нового сотрудника с некорректными данными. | 1. Авторизоваться как администратор.  2. Нажать на кнопку «Зарегистрировать сотрудника».  3. Ввод данных в поля: Логин = Emp1, Пароль = 1  4. Нажать на кнопку «Зарегистрироваться». | Сотрудник не был добавлен, т.к. такой уже существует | После нажатия на кнопку возникло оповещение о том, что пользователь с ником «Emp1» уже существует. Результат работы представлен на рисунке 6.7 |

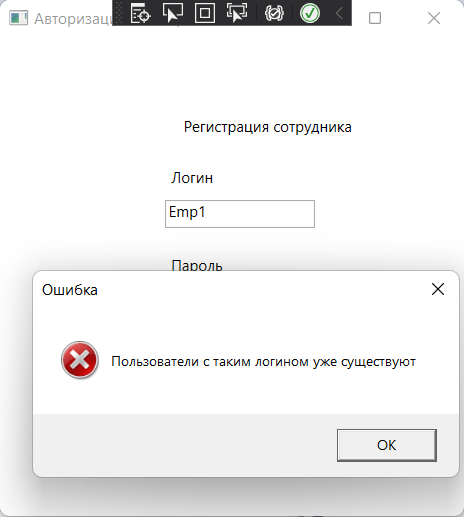


Рисунок 6.7

Таблица 6.7 – Функция тестирования добавления пиццы в корзину.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Модуль / Функция | Шаги выполнения | Ожидаемый результат | Фактический результат |
| 7 | Проверка добавления пиццы в корзину пользователя. | 1. На главной форме выбираем пиццу «Супер пепперони».  2. Нажать на кнопку «Добавить в корзину».  3. Нажать на кнопку «Корзина/Меню». | В корзину была добавлена пицца «Супер пепперони». | После нажатия на кнопку в корзине пользователя появилась пицца «Супер пепперони». Результат работы представлен на рисунках 6.8, 6.9 |

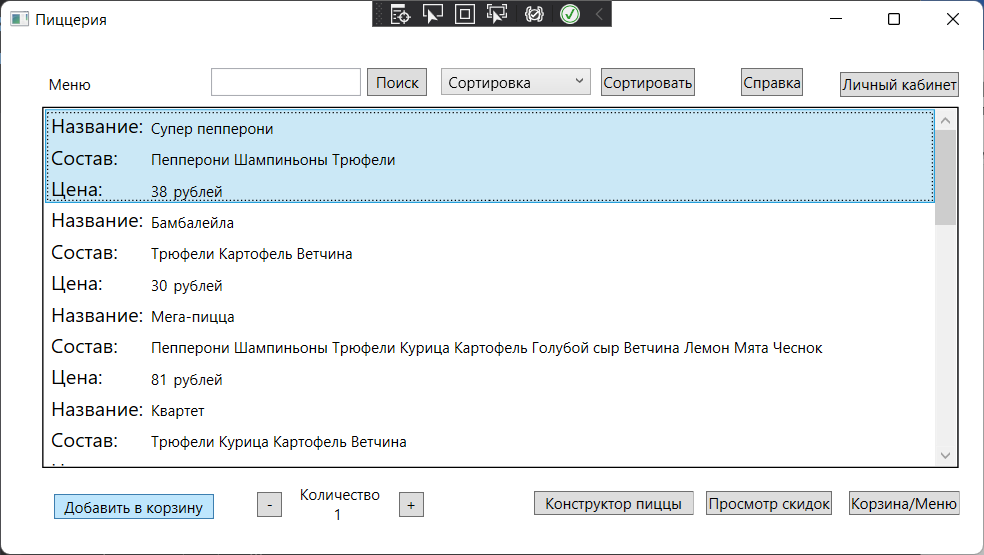


Рисунок 6.8

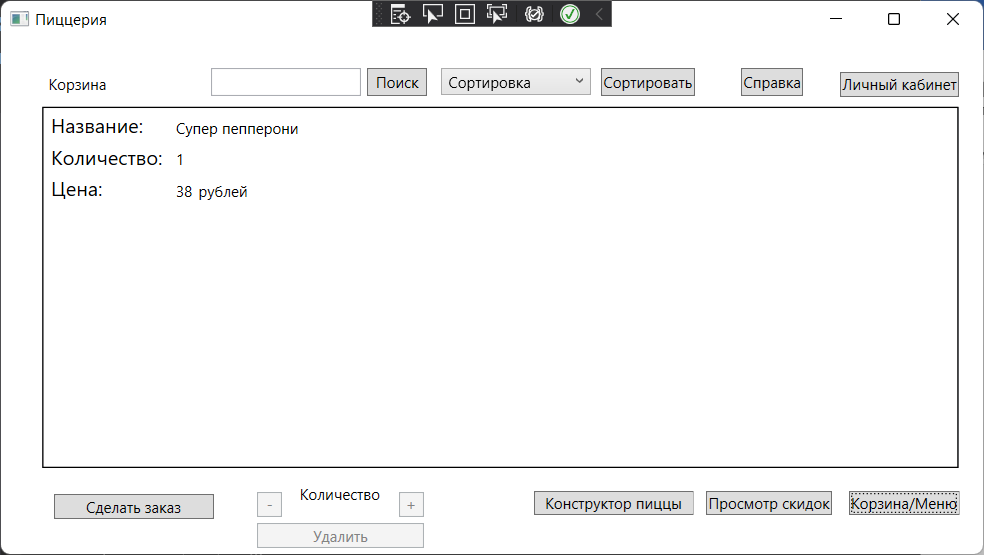


Рисунок 6.9

Таблица 6.8 – Функция тестирования удаления товара из корзины.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Модуль / Функция | Шаги выполнения | Ожидаемый результат | Фактический результат |
| 8 | Проверка удаления товара из корзины. | 1. Зайти на главную страницу приложения.  2. Нажать на кнопку «Корзина/Меню».  3. Выбрать пиццу «Супер пепперони»  4. Нажать на кнопку «Удалить». | Удаление «Супер пепперони» из списка пицц в корзине пользователя. | После нажатия на кнопку «Удалить», «Супер пепперони» была удалена из корзины пользователя. Результат работы отображен на рисунках 6.9, 6.10. |

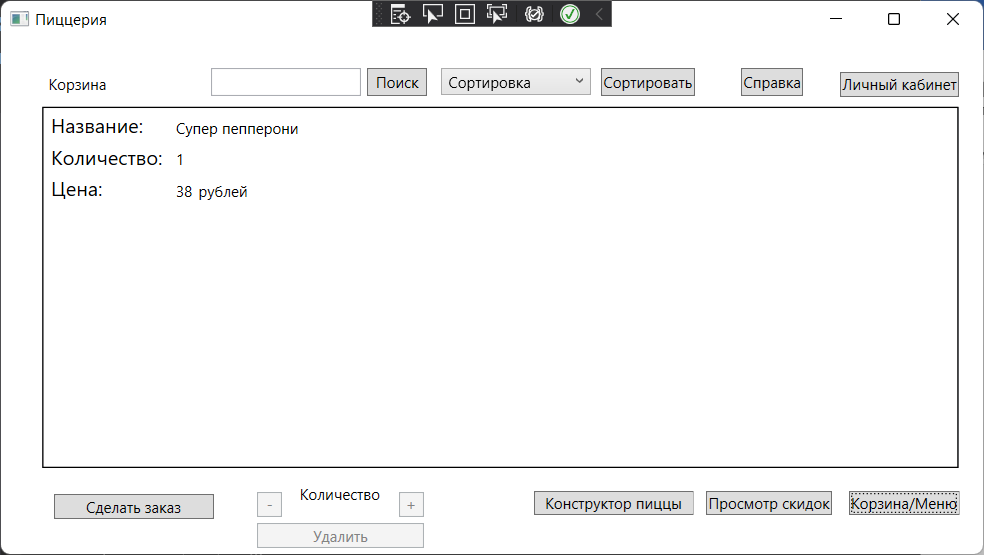


Рисунок 6.9

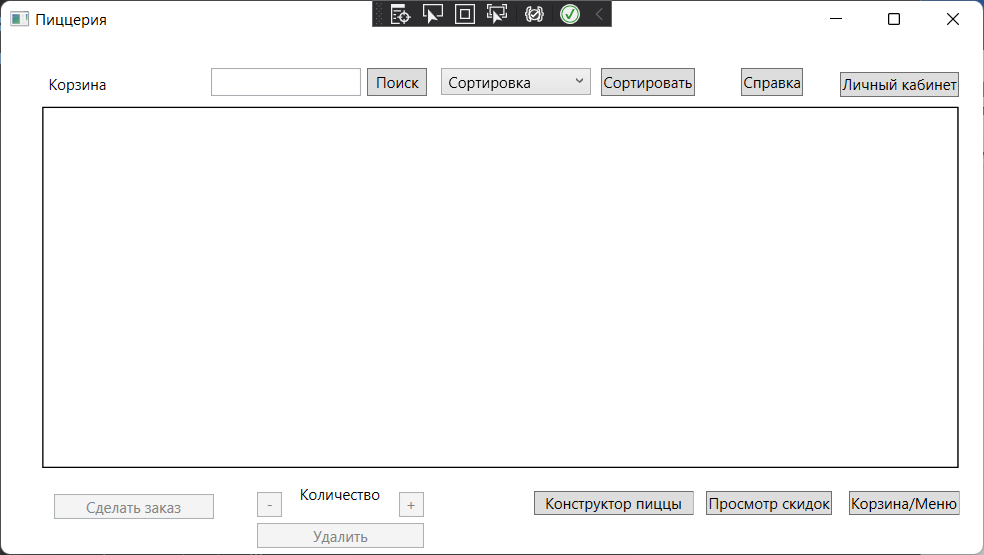


Рисунок 6.9

Таблица 6.9 – Функция тестирования оформления заказа.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Модуль / Функция | Шаги выполнения | Ожидаемый результат | Фактический результат |
| 9 | Проверка оформления заказа пользователем. | 1. Перейти на главную страницу формы.  2. Добавить в корзину 3 пиццы «Бамбалейла», «Супер пепперони», «Квартет».  3. Нажать на кнопку «Корзина/Меню»  4. Нажать на кнопку «Сделать заказ». | Оформления заказа пользователя с выбранными пиццами. | После нажатия на кнопку « Сделать заказ » Был оформлен заказ пользователя с 3 пиццами по его выбору. Результат работы представлен на рисунке 6.10. |

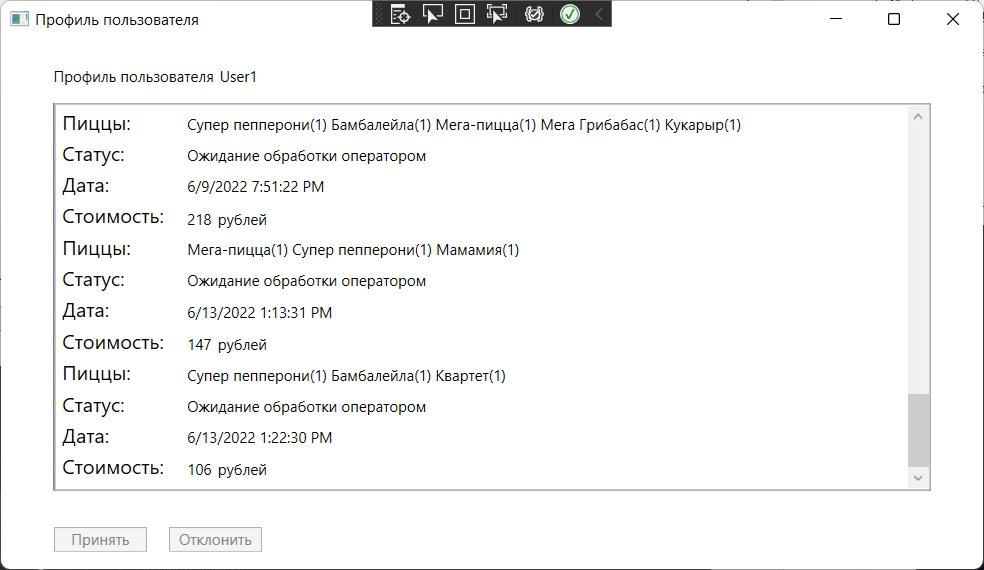


Рисунок 6.10

Таблица 6.10 – Функция тестирования экспорта данных заказов.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Модуль / Функция | Шаги выполнения | Ожидаемый результат | Фактический результат |
| 10 | Проверка экспорта данных. | 1. Авторизоваться как администратор.  2. Нажать на кнопку «Учёт заказов».  2. Нажать на кнопку «экспорт». | Отображение документа MS Office Word с представленной таблицей заказов. | После нажатия на кнопку экспорт отобразился документ MS Office Word с таблицей. Результат работы представлен на рисунке 6.11. |

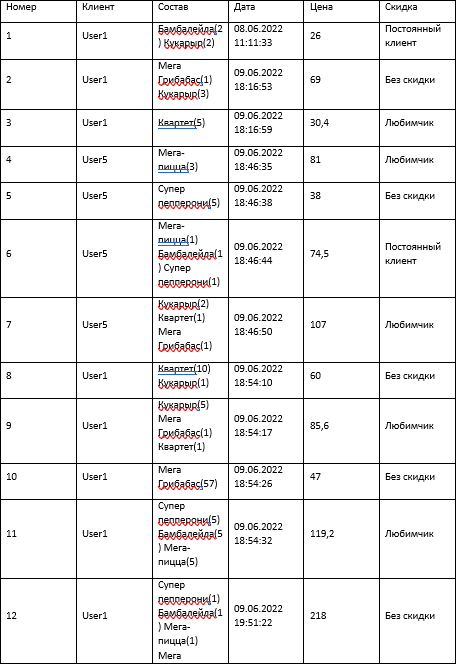


Рисунок 6.11

1. Применение
   1. Назначение программы

Приложение «SuperPizzaria.exe» предназначено для автоматизации рабочего места администратора, сотрудника и для автоматизации составления оформления заказов пользователями. Программное средство предоставляет пользователю возможности для поиска, отображения и редактирования имеющихся в базе пицц, предоставляет инструменты для создания и добавления пицц, управления меню, интерфейс для работы со скидами и экспортом отчётов.

* 1. Условия применение

Для работы с программой необходимо наличие программного обеспечения:

* операционная система, начиная с Windows 7;
* система управления базами данных Microsoft SQL Server Express 2019;
* библиотека Microsoft .NET Framework 6.0 с пакетом обновления 1 (SP1);
* офисный пакет приложений Microsoft Office (MS Office) 2021;

На случай редактирования проекта программы необходимо наличие программного обеспечения:

* интегрированная среда Microsoft SQL Server Management Studio 2010;
* среда разработки приложений Microsoft Visual Studio 2019.
  1. Справочная система

Справочная система приложения представлена с помощью нескольких расположенных в разных местах приложения справочных кнопок. Всего в приложении представлено 3 справочника.

Первый справочник находится в главном окне, он необходим для понимания пользователем всего доступного ему функционала. Там описаны все доступные ему разделы и кнопки с подробным описанием функционала. Первый справочник представлен на рисунке 7.

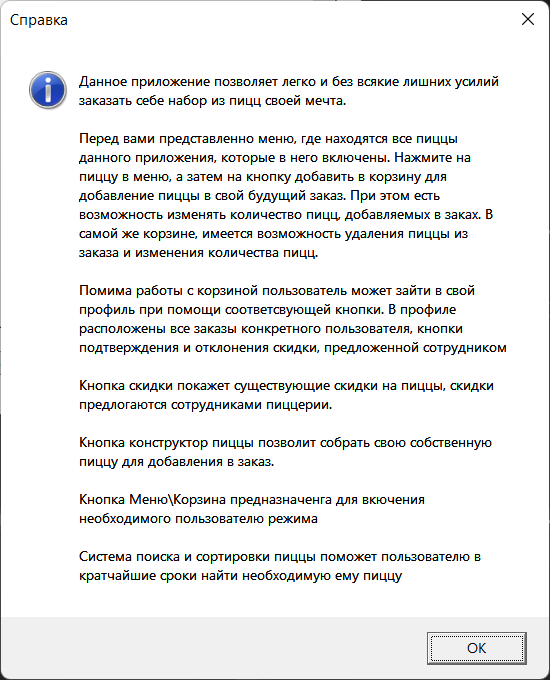


Рисунок 7

Второй справочник создан для администрации. Он предоставляет всю необходимую информацию о функционировании части приложения, предназначенной для администратора пиццерии. Второй справочник представлен на рисунке 8.

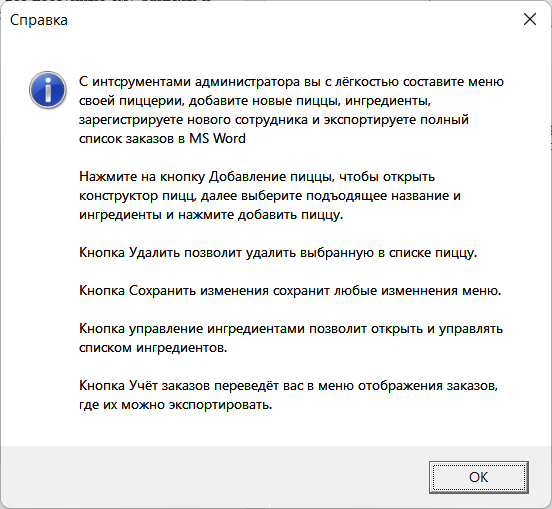


Рисунок 8

Третий справочник создан для сотрудника. Он предоставляет всю необходимую информацию о функционировании части приложения, предназначенной для сотрудника пиццерии. Третий справочник представлен на рисунке 9.

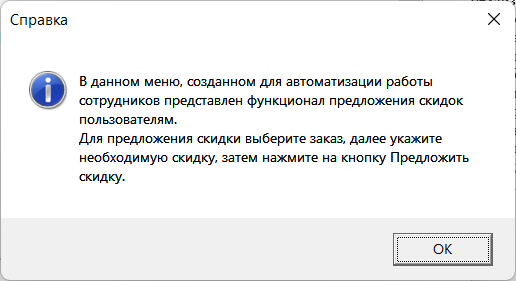
.

Рисунок 9

Заключение

В рамках курсового проектирования на тему «Программное средство для учета заказов в пиццерии было разработано программное средство «SuperPizzeria.exe», автоматизирующее часть работы администратора, сотрудника и пользователя.

Для достижения целей курсового проектирования были решены следующие задачи:

* определена вычислительная система, необходимая для создания программного средства;
* разработана физическая и логическая модель данных;
* по модели выполнено проектирование задачи;
* разработано программное средство;
* описано созданное программное средство;
* выбрана методика испытаний;
* описан процесс тестирования;
* приведены примеры области применения.

Разработка имеет интуитивно понятный графический интерфейс, позволяющий даже с минимальным знанием компьютера использовать данное программного средство.

Программа реализована в полном объеме и в соответствии с заданными требованиями, полностью отлажена и протестирована. Поставленные задачи выполнены.

К достоинствам данного программного средства можно отнести: интуитивно-понятный интерфейс, простоту в использовании.

В процессе разработки данного программного средства были применены и закреплены знания по уже изученному материалу, были отработаны навыки владения методами надежного программирования и эффективности разработки программного обеспечения в Microsoft Visual Studio 2019 с использованием языка программирования C#, разработана база данных средствами системы управления базами данных Microsoft SQL Server 2019.

Программное средство готово к практическому использованию на предприятии «SuperPizzaria».

Данная программа может быть дополнена и модернизирована.

Список информационных источников

1. Багласова, Т.Г. Методические указания по выполнению курсового проекта для учащихся по специальности 2-40 01 01 «Программное обеспечение технологий» / Т.Г. Багласова. – Минск : КБП, 2022.
2. Багласова, Т.Г. Методические указания по оформлению курсовых и дипломных проектов / Т.Г. Багласова, К.О. Якимович. – Минск : КБП, 2013. – 29 c.
3. В.Ю. Методические указания к курсовому проектированию / В.Ю. Михалевич. – Минск: КБП, 2022.
4. Общие требования к тестовым документам : ГОСТ 2.105-95. – Введ. 01.01.1996 – Минск : Межгос. совет по стандартизации, метрологии и сертификации, 1995. – 84 с.
5. Программа и методика испытаний. Требования к содержанию, оформлению и контролю качества: ГОСТ 19.301-2000. – Введ. 01.09.2001. – Минск: Межгос. совет по стандартизации, метрологии и сертификации, 2000. – 14 с.
6. .NET Framework [Электронный ресурс]. – Microsoft, 2022. – Режим доступа : https://docs.microsoft.com/en-us/dotnet/framework/. – Дата доступа : 10.06.2022.
7. C# [Электронный ресурс]. – Microsoft, 2022. – Режим доступа : https://docs.microsoft.com/en-us/dotnet/cs/. – Дата доступа : 10.06.2022.
8. Draw.io [Электронный ресурс]. – Drawio, 2022. – Режим доступа : <https://drawiopid.com/en/what-is-drawio/>. – Дата доступа : 10.06.2022.
9. MS Office [Электронный ресурс]. – Microsoft, 2021. – Режим доступа : https://officeapplications.net/microsoft-word/. – Дата доступа : 10.06.2022.
10. SQL Server 2019 [Электронный ресурс]. – Microsoft, 2019. – Режим доступа : https://docs.microsoft.com/ru-ru/sql/tutorials/tutorials-for-sql-server-2019?view=sql-server-2019/. – Дата доступа : 10.06.2022.
11. Visual Studio [Электронный ресурс]. – Microsoft, 2022. – Режим доступа : https://visualstudio.microsoft.com/vs/. – Дата доступа : 10.06.2022.
12. WPF [Электронный ресурс]. – Microsoft, 2022 – Режим доступа : <https://docs.microsoft.com/en-us/vs/designers/getting-started-with-wpf/>. – Дата доступа: 10.06.2022.

Приложение А

(обязательное)

Текст программных модулей

/// Код программы MainWindow

namespace SuperPizzeria

{

/// <summary>

/// Логика взаимодействия для MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

ApplicationContext db;

private bool switcher = false;

private int UserId;

List<Pizza> pizzasList;

public MainWindow(int userId=2)

{

InitializeComponent();

SortOptions.SelectedIndex = 0;

UserId = userId;

db = new ApplicationContext();

db.Pizzas.Where(i=>i.IsOnMenu).Load();

db.CartUnits.Where(i => i.UserId == UserId && i.Order == 0).Load();

pizzasList = db.Pizzas.Local.ToList();

Menu.ItemsSource = pizzasList;

Cart.ItemsSource = db.CartUnits.Local.ToBindingList();

}

private void Sort\_Click(object sender, RoutedEventArgs e)

{

switch (SortOptions.SelectedIndex)

{

case 0:

{

pizzasList = pizzasList.OrderBy(i => i.Id).ToList();

break;

}

case 1:

{

pizzasList = pizzasList.OrderBy(i => i.Cost).ToList();

break;

}

case 2:

{

pizzasList = pizzasList.OrderByDescending(i => i.Cost).ToList();

break;

}

}

Menu.ItemsSource = null;

Menu.ItemsSource = pizzasList;

Menu.Items.Refresh();

}

public void ReloadMenu(object sender, System.ComponentModel.CancelEventArgs e)

{

db.Dispose();

db = new ApplicationContext();

Menu.ItemsSource = null;

db.Pizzas.Where(i => i.IsOnMenu).Load();

Menu.ItemsSource = db.Pizzas.Local.ToBindingList();

}

private void MainWindow\_Closing(object sender, System.ComponentModel.CancelEventArgs e)

{

db.Dispose();

AuthReg ar = new AuthReg();

ar.Show();

}

private void PlusButton\_Click(object sender, RoutedEventArgs e)

{

AmountOfPizzas.Text = Convert.ToString(Convert.ToInt32(AmountOfPizzas.Text) + 1);

}

private void MinusButton\_Click(object sender, RoutedEventArgs e)

{

if(Convert.ToInt32(AmountOfPizzas.Text)>1)

AmountOfPizzas.Text = Convert.ToString(Convert.ToInt32(AmountOfPizzas.Text) - 1);

}

private void Switch\_Click(object sender, RoutedEventArgs e)

{

Visibility cartVisibility = switcher ? Visibility.Hidden : Visibility.Visible;

Visibility menuVisibility = switcher ? Visibility.Visible : Visibility.Hidden;

switcher = !switcher;

GeneralLabel.Content = switcher ? "Корзина" : "Меню";

CartGrid.Visibility = cartVisibility;

MenuGrid.Visibility = menuVisibility;

MenuItems.Visibility = menuVisibility;

CartItems.Visibility = cartVisibility;

if (db.CartUnits.Where(i => i.UserId == UserId && i.Order == 0).Count() == 0)

OrderButton.IsEnabled = false;

else

OrderButton.IsEnabled = true;

db.CartUnits.Where(i => i.UserId == UserId && i.Order == 0).Load();

Cart.ItemsSource = db.CartUnits.Local.ToBindingList();

Cart.Items.Refresh();

}

private void Cart\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

AmountOfPizzasCart.DataContext = Cart.SelectedItem;

MinusButtonCart.IsEnabled = true;

PlusButtonCart.IsEnabled = true;

Delete.IsEnabled = true;

}

private void Button\_Click(object sender, RoutedEventArgs e)

{

int amount = Convert.ToInt32(AmountOfPizzas.Text);

int pizzaId = ((Pizza)Menu.SelectedItem).Id;

if (db.CartUnits.Where(i => i.Pizza == pizzaId && i.Order == 0).Count() > 0)

{

CartUnit cu = db.CartUnits.Where(i => i.Pizza == pizzaId && i.Order == 0).ToArray()[0];

cu.Amount += amount;

}

else

{

db.CartUnits.Add(new CartUnit { Amount = amount, Pizza = pizzaId, UserId = UserId, Order = 0 });

}

db.SaveChanges();

Cart.Items.Refresh();

}

private void PlusButtonCart\_Click(object sender, RoutedEventArgs e)

{

CartUnit cu = (CartUnit)Cart.SelectedItem;

cu.Amount = cu.Amount + 1;

db.SaveChanges();

Cart.Items.Refresh();

AmountOfPizzasCart.DataContext = null;

AmountOfPizzasCart.DataContext = Cart.SelectedItem;

}

private void MinusButtonCart\_Click(object sender, RoutedEventArgs e)

{

CartUnit cu = (CartUnit)Cart.SelectedItem;

if (cu.Amount > 1)

cu.Amount--;

db.SaveChanges();

Cart.Items.Refresh();

AmountOfPizzasCart.DataContext = null;

AmountOfPizzasCart.DataContext = Cart.SelectedItem;

}

private void Delete\_Click(object sender, RoutedEventArgs e)

{

CartUnit cu = (CartUnit)Cart.SelectedItem;

db.CartUnits.Remove(cu);

db.SaveChanges();

Cart.Items.Refresh();

MinusButtonCart.IsEnabled = false;

PlusButtonCart.IsEnabled = false;

Delete.IsEnabled = false;

if (db.CartUnits.Where(i => i.UserId == UserId && i.Order == 0).Count() == 0)

OrderButton.IsEnabled = false;

else

OrderButton.IsEnabled = true;

}

private void OrderButton\_Click(object sender, RoutedEventArgs e)

{

var orderUnits = db.CartUnits.Where(i => i.UserId == UserId && i.Order == 0);

Cart.Items.Refresh();

db.Orders.Add(new Order() { DateOfOrder = DateTime.Now, UserId = UserId, Sale = 0, Status = 0 });

db.SaveChanges();

Cart.Items.Refresh();

Order lastOrder = db.Orders.ToList().Last();

foreach (CartUnit orderUnit in orderUnits)

{

orderUnit.Order = lastOrder.Id;

}

db.SaveChanges();

Cart.Items.Refresh();

Cart.ItemsSource = null;

Cart.Items.Clear();

db.Dispose();

db = new ApplicationContext();

Cart.Items.Refresh();

}

private void UserProfile\_Click(object sender, RoutedEventArgs e)

{

Profiler profile= new Profiler(UserId);

profile.ShowDialog();

}

private void CreatePizza\_Click(object sender, RoutedEventArgs e)

{

Administration ad = new Administration(1);

ad.Closing += ReloadMenu;

ad.ShowDialog();

}

private void Search\_Click(object sender, RoutedEventArgs e)

{

string text = SearchQuery.Text;

if (text.Length == 0)

{

db.Pizzas.Where(i => i.IsOnMenu).Load();

pizzasList = db.Pizzas.Local.ToList();

}

else

{

Regex nameReg = new Regex(@"[а-я]+");

string pizzaName = "";

foreach (Match match in nameReg.Matches(text.ToLower()))

{

pizzaName += match + " ";

}

List<Pizza> newPizzas = new List<Pizza>();

foreach (Pizza p in db.Pizzas.Where(p => p.Name.ToLower().IndexOf(pizzaName.ToLower()) != -1))

{

newPizzas.Add(p);

}

pizzasList = newPizzas;

}

Menu.ItemsSource = null;

Menu.ItemsSource = pizzasList;

Menu.Items.Refresh();

}

private void Sales\_Click(object sender, RoutedEventArgs e)

{

string salesInfo = "";

foreach (var sale in db.Sales.ToList())

{

salesInfo += sale.Summary + " " + sale.Multiplier + "\n";

}

MessageBox.Show(salesInfo, "Скидки");

}

private void Spravka\_Click(object sender, RoutedEventArgs e)

{

string inforamtion = "Данное приложение позволяет легко и без всякие лишних усилий заказать себе набор из пицц своей мечта." +

"\n\nПеред вами представленно меню, где находятся все пиццы данного приложения, которые в него включены. " +

"Нажмите на пиццу в меню, а затем на кнопку добавить в корзину для добавление пиццы в свой будущий заказ. При этом есть возможность " +

"изменять количество пицц, добавляемых в заках. В самой же корзине, имеется возможность удаления пиццы из заказа и изменения количества пицц." +

"\n\nПомима работы с корзиной пользователь может зайти в свой профиль при помощи соответсвующей кнопки. В профиле расположены все заказы кон" +

"кретного пользователя, кнопки подтверждения и отклонения скидки, предложенной сотрудником" +

"\n\nКнопка скидки покажет существующие скидки на пиццы, скидки предлогаются сотрудниками пиццерии." +

"\n\nКнопка конструктор пиццы позволит собрать свою собственную пиццу для добавления в заказ." +

"\n\nКнопка Меню\\Корзина предназначенга для вкючения необходимого пользователю режима\n\n" +

"Система поиска и сортировки пиццы поможет пользователю в кратчайшие сроки найти необходимую ему пиццу";

MessageBox.Show(inforamtion, "Справка", MessageBoxButton.OK, MessageBoxImage.Information);

}

}

}

/// Код программы Administration

namespace SuperPizzeria

{

/// <summary>

/// Логика взаимодействия для Administration.xaml

/// </summary>

public partial class Administration : Window

{

ApplicationContext db;

ArrayList ingsList;

public Administration(int mode)

{

InitializeComponent();

db = new ApplicationContext();

db.Pizzas.Load();

if (mode == 0)

{

AddPizza.Click += AddPizza\_Click;

Back.Click += Back\_Click;

}

else

{

PizzaMode();

ingsList = new ArrayList();

AddPizza.Click += AddPizzaUser\_Click;

Back.Click += BackUser\_Click;

}

MenuGrid.ItemsSource = db.Pizzas.Local.ToBindingList();

IngridientsList.ItemsSource = db.Ingridients.Local.ToBindingList();

this.Closing += Administration\_Closing;

}

private void Administration\_Closing(object sender, System.ComponentModel.CancelEventArgs e)

{

db.Dispose();

AuthReg ar = new AuthReg();

ar.Show();

}

private void EmployeeRegButton\_Click(object sender, RoutedEventArgs e)

{

AuthReg ar = new AuthReg(true);

ar.Show();

this.Closing -= Administration\_Closing;

db.Dispose();

this.Close();

}

private void BackUser\_Click(object sender, RoutedEventArgs e)

{

this.Closing -= Administration\_Closing;

db.Dispose();

this.Close();

}

private void IngridientsControl\_Click(object sender, RoutedEventArgs e)

{

Ingridients ingr = new Ingridients();

ingr.ShowDialog();

}

private void SaveChanges\_Click(object sender, RoutedEventArgs e)

{

try

{

db.SaveChanges();

db.Pizzas.Load();

MenuGrid.ItemsSource = db.Pizzas.Local.ToBindingList();

}

catch (DbUpdateException ex)

{

MessageBox.Show("Сначала заполните данные у созданных ингридиентов!", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void AddPizzaMenu\_Click(object sender, RoutedEventArgs e)

{

PizzaMode();

ingsList = new ArrayList();

}

private void AddPizza\_Click(object sender, RoutedEventArgs e)

{

if(IngridientsList.SelectedItems.Count < 2 || PizzaName.Text.Length < 5)

{

MessageBox.Show("Некорректные данные для создания пиццы", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

return;

}

var ings = IngridientsList.SelectedItems;

foreach (Ingridient ing in ings)

{

ingsList.Add(ing.Id);

}

Pizza pizza = new Pizza() { Name = Convert.ToString(PizzaName.Text), IngridientsList = Functions.ArrayListToString(ingsList) };

PizzaName.Text = "";

db.Pizzas.Add(pizza);

PizzasMode();

}

private void AddPizzaUser\_Click(object sender, RoutedEventArgs e)

{

if (IngridientsList.SelectedItems.Count < 2 || PizzaName.Text.Length < 5)

{

MessageBox.Show("Некорректные данные для создания пиццы", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

return;

}

var ings = IngridientsList.SelectedItems;

foreach (Ingridient ing in ings)

{

ingsList.Add(ing.Id);

}

Pizza pizza = new Pizza() { Name = Convert.ToString(PizzaName.Text), IngridientsList = Functions.ArrayListToString(ingsList), IsOnMenu = true };

PizzaName.Text = "";

db.Pizzas.Add(pizza);

SaveChanges\_Click(sender, e);

BackUser\_Click(sender, e);

}

private void PizzasMode()

{

PizzasGrid.Visibility = Visibility.Visible;

PizzaGrid.Visibility = Visibility.Hidden;

AddPizza.Visibility = Visibility.Hidden;

Back.Visibility = Visibility.Hidden;

DeletePizza.Visibility = Visibility.Visible;

//EditPizza.Visibility = Visibility.Visible;

EmployeeRegButton.Visibility = Visibility.Visible;

IngridientsControl.Visibility = Visibility.Visible;

OrderAccounting.Visibility = Visibility.Visible;

GridLabel.Content = "Список пицц";

Spravka.Visibility = Visibility.Visible;

SaveChanges.Visibility = Visibility.Visible;

}

private void PizzaMode()

{

PizzasGrid.Visibility = Visibility.Hidden;

PizzaGrid.Visibility = Visibility.Visible;

AddPizza.Visibility = Visibility.Visible;

Back.Visibility = Visibility.Visible;

DeletePizza.Visibility = Visibility.Hidden;

EmployeeRegButton.Visibility = Visibility.Hidden;

IngridientsControl.Visibility = Visibility.Hidden;

OrderAccounting.Visibility = Visibility.Hidden;

Spravka.Visibility = Visibility.Hidden;

//EditPizza.Visibility = Visibility.Hidden;

GridLabel.Content = "Создание пиццы";

SaveChanges.Visibility = Visibility.Hidden;

db.Ingridients.Load();

IngridientsList.ItemsSource = db.Ingridients.Local.ToBindingList();

IngridientsList.UnselectAll();

}

private void EditPizza\_Click(object sender, RoutedEventArgs e)

{

Binding bind = new Binding();

bind.ElementName = "Pizza";

bind.Path = new PropertyPath("Name");

IngridientsInGrid.Binding = bind;

}

private void DeletePizza\_Click(object sender, RoutedEventArgs e)

{

Pizza pizza = (Pizza)MenuGrid.SelectedItem;

db.Pizzas.Remove(pizza);

DeletePizza.IsEnabled = false;

}

private void Back\_Click(object sender, RoutedEventArgs e)

{

PizzasMode();

}

private void IngridientsList\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

IngridientCost.Text = "0";

var ings = IngridientsList.SelectedItems;

foreach (Ingridient ing in ings)

{

IngridientCost.Text = Convert.ToString(Convert.ToInt32(IngridientCost.Text) + ing.Cost);

}

}

private void OrderAccounting\_Click(object sender, RoutedEventArgs e)

{

OrdersAccounting mw = new OrdersAccounting(1);

mw.ShowDialog();

}

private void MenuGrid\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

DeletePizza.IsEnabled = true;

}

private void Spravka\_Click(object sender, RoutedEventArgs e)

{

string inforamtion = "С интсрументами администратора вы с лёгкостью составите меню своей пиццерии, добавите новые пиццы, ингредиенты, зарегистрируете нового сотрудника и экспортируете полный список заказов в MS Word" +

"\n\nНажмите на кнопку Добавление пиццы, чтобы открыть конструктор пицц, далее выберите подъодящее название и ингредиенты и нажмите добавить пиццу." +

"\n\nКнопка Удалить позволит удалить выбранную в списке пиццу." +

"\n\nКнопка Сохранить изменения сохранит любые изменнения меню." +

"\n\nКнопка управление ингредиентами позволит открыть и управлять списком ингредиентов." +

"\n\nКнопка Учёт заказов переведёт вас в меню отображения заказов, где их можно экспортировать.";

MessageBox.Show(inforamtion, "Справка", MessageBoxButton.OK, MessageBoxImage.Information);

}

}

}

/// Код программы OrdersAccounting

namespace SuperPizzeria

{

/// <summary>

/// Логика взаимодействия для EmployeeMode.xaml

/// </summary>

public partial class OrdersAccounting : System.Windows.Window

{

ApplicationContext db;

Order selectedOrder;

bool firstKey = false;

bool secondKey = false;

int Status;

public OrdersAccounting(int status)

{

InitializeComponent();

db = new ApplicationContext();

this.Status = status;

if (Status == 0)

{

db.Orders.Where(i => i.Status == 0).Load();

}

else

{

db.Orders.Load();

SuggestSale.Visibility = Visibility.Hidden;

Sales.Visibility = Visibility.Hidden;

AccountOrders.Visibility = Visibility.Visible;

SalesLabel.Visibility = Visibility.Hidden;

}

db.Sales.Load();

Orders.ItemsSource = db.Orders.Local.ToBindingList();

Sales.ItemsSource = db.Sales.Local.ToBindingList();

this.Closing += EmployeeMode\_Closing;

}

private void EmployeeMode\_Closing(object sender, System.ComponentModel.CancelEventArgs e)

{

db.Dispose();

AuthReg ar = new AuthReg();

ar.Show();

}

private void Sales\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

secondKey = true;

if (secondKey && firstKey)

{

SuggestSale.IsEnabled = true;

}

}

private void SuggestSale\_Click(object sender, RoutedEventArgs e)

{

selectedOrder.Sale = ((Sale)Sales.SelectedItem).Id;

selectedOrder.Status = 1;

SuggestSale.IsEnabled = false;

firstKey = false;

db.SaveChanges();

Orders.Items.Refresh();

db.Dispose();

Orders.SelectionChanged -= Orders\_SelectionChanged;

Orders.ItemsSource = null;

db = new ApplicationContext();

if (Status == 0)

{

db.Orders.Where(i => i.Status == 0).Load();

}

else

{

db.Orders.Load();

}

Orders.ItemsSource = db.Orders.Local.ToBindingList();

Orders.SelectionChanged += Orders\_SelectionChanged;

}

private void Orders\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

selectedOrder = (Order)Orders.SelectedItem;

if(selectedOrder.Status == 0 )

firstKey = true;

if(secondKey && firstKey)

{

SuggestSale.IsEnabled = true;

}

}

private void AccountOrders\_Click(object sender, RoutedEventArgs e)

{

var application = new Microsoft.Office.Interop.Word.Application();

Document document = application.Documents.Add();

Microsoft.Office.Interop.Word.Paragraph actorParagrap = document.Paragraphs.Add();

Microsoft.Office.Interop.Word.Range actorRange = actorParagrap.Range;

actorRange.Text = "Заказы пиццерии";

actorParagrap.set\_Style("Обычный");

actorRange.InsertParagraphAfter();

List<Order> orders = db.Orders.ToList();

Microsoft.Office.Interop.Word.Paragraph tableParagrap = document.Paragraphs.Add();

Microsoft.Office.Interop.Word.Range tableRange = tableParagrap.Range;

Microsoft.Office.Interop.Word.Table priceTable = document.Tables.Add(tableRange, orders.Count() + 1, 6);

priceTable.Borders.InsideLineStyle = priceTable.Borders.OutsideLineStyle = Microsoft.Office.Interop.Word.WdLineStyle.wdLineStyleSingle;

priceTable.Range.Cells.VerticalAlignment = Microsoft.Office.Interop.Word.WdCellVerticalAlignment.wdCellAlignVerticalCenter;

priceTable.Borders.InsideLineStyle = priceTable.Borders.OutsideLineStyle = Microsoft.Office.Interop.Word.WdLineStyle.wdLineStyleSingle;

priceTable.Range.Cells.VerticalAlignment = Microsoft.Office.Interop.Word.WdCellVerticalAlignment.wdCellAlignVerticalCenter;

Microsoft.Office.Interop.Word.Range cellRange;

cellRange = priceTable.Cell(1, 1).Range;

cellRange.Text = "Номер";

cellRange = priceTable.Cell(1, 2).Range;

cellRange.Text = "Клиент";

cellRange = priceTable.Cell(1, 3).Range;

cellRange.Text = "Состав";//Название пицц

cellRange = priceTable.Cell(1, 4).Range;

cellRange.Text = "Дата";

cellRange = priceTable.Cell(1, 5).Range;

cellRange.Text = "Цена";

cellRange = priceTable.Cell(1, 6).Range;

cellRange.Text = "Скидка";

for (int i = 0; i < orders.Count(); ++i)

{

Order order = orders[i];

cellRange = priceTable.Cell(i + 2, 1).Range;

cellRange.Text = order.Id.ToString();

cellRange = priceTable.Cell(i + 2, 2).Range;

cellRange.Text = order.GetUserName;

cellRange = priceTable.Cell(i + 2, 3).Range;

cellRange.Text = order.GetPizzasNames;

cellRange = priceTable.Cell(i + 2, 4).Range;

cellRange.Text = order.GetOrderDate;

cellRange = priceTable.Cell(i + 2, 5).Range;

cellRange.Text = order.GetOrderCost.ToString();

cellRange = priceTable.Cell(i + 2, 6).Range;

cellRange.Text = order.GetSaleName;

}

application.Visible = true;

}

private void Spravka\_Click(object sender, RoutedEventArgs e)

{

string inforamtion = "В данном меню, созданном для автоматизации работы сотрудников представлен функционал предложения скидок пользователям." +

"\nДля предложения скидки выберите заказ, далее укажите необходимую скидку, затем нажмите на кнопку Предложить скидку.";

MessageBox.Show(inforamtion, "Справка", MessageBoxButton.OK, MessageBoxImage.Information);

}

}

}

/// Код программы Authoziration

namespace SuperPizzeria

{

/// <summary>

/// Логика взаимодействия для Authorization.xaml

/// </summary>

public partial class Authorization : Page

{

Window wind;

public Authorization(Window win)

{

InitializeComponent();

wind = win;

}

private void Button\_Click(object sender, RoutedEventArgs e)

{

((AuthReg)wind).AuthRegFrame.Content = new Registration(wind);

}

private void Button\_Click\_1(object sender, RoutedEventArgs e)

{

using (ApplicationContext db = new ApplicationContext())

{

var users = db.Users.Where(i => i.Login == LoginField.Text && i.Password == PasswordField.Text);

if (users.Count() > 0)

{

switch (users.ToArray()[0].Status)

{

case 1:

{

MainWindow mw = new MainWindow(users.ToArray()[0].Id);

mw.Show();

break;

}

case 2:

{

OrdersAccounting mw = new OrdersAccounting(0);

mw.Show();

break;

}

case 3:

{

Administration mw = new Administration(0);

mw.Show();

break;

}

}

wind.Close();

}

else

{

MessageBox.Show("Пользователя с такими данными не существует", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

}

}

}

/// Код программы Ingridients

namespace SuperPizzeria

{

/// <summary>

/// Логика взаимодействия для Ingrdients.xaml

/// </summary>

public partial class Ingridients : Window

{

ApplicationContext db;

private List<Ingridient> ingList;

public Ingridients()

{

InitializeComponent();

db = new ApplicationContext();

db.Ingridients.Load();

ingList = db.Ingridients.Local.ToList();

IngridientsList.ItemsSource = ingList;

this.Closing += Ingridients\_Closing;

}

private void Ingridients\_Closing(object sender, System.ComponentModel.CancelEventArgs e)

{

db.Dispose();

}

private void Create\_Click(object sender, RoutedEventArgs e)

{

Ingridient ing = new Ingridient();

db.Ingridients.Add(ing);

}

private void Edit\_Click(object sender, RoutedEventArgs e)

{

Console.WriteLine(sender);

}

private bool IsAllFull()

{

foreach(Ingridient ing in ingList)

{

if (ing.Name is null || ing.Cost<=0)

return false;

}

return true;

}

private void SaveChanges\_Click(object sender, RoutedEventArgs e)

{

try

{

if(IsAllFull())

db.SaveChanges();

else

MessageBox.Show("Прежде чем сохранять список, заполните все ингредиенты корректнымми данными", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

catch (DbUpdateException ex)

{

MessageBox.Show("Сначала заполните данные у созданных ингридиентов!", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void Delete\_Click(object sender, RoutedEventArgs e)

{

Ingridient ing = (Ingridient)IngridientsList.SelectedItem;

db.Ingridients.Remove(ing);

Delete.IsEnabled = false;

}

private void IngridientsList\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

Delete.IsEnabled = true;

}

}

}

/// Код программы Functions

namespace SuperPizzeria.Classes

{

class Functions

{

public static bool VerifyPassword(string password)

{

if (password.Contains(" ")) return false;

if (password.Length == 0) return false;

return true;

}

public static int[] StringToIntArray(string str)

{

int[] arr=new int[new Regex("-").Matches(str).Count+1];

string[] strArray = str.Split('-');

for(int i =0; i<strArray.Length;i++)

{

arr[i] = Convert.ToInt32(strArray[i]);

}

return arr;

}

public static string IntArrayToString(int[] arr)

{

string str = "";

foreach(int i in arr)

{

str += i + "-";

}

return str.Substring(0, str.Length - 1);

}

public static ArrayList StringToArrayList(string str)

{

ArrayList al = new ArrayList();

string[] strArray = str.Split('-');

for (int i = 0; i < strArray.Length; i++)

{

al.Add(Convert.ToInt32(strArray[i]));

}

return al;

}

public static string ArrayListToString(ArrayList arr)

{

string str = "";

foreach (int i in arr)

{

str += i + "-";

}

return str.Substring(0, str.Length - 1);

}

}

}

/// Код программы Profiler

namespace SuperPizzeria

{

/// <summary>

/// Логика взаимодействия для Profiler.xaml

/// </summary>

public partial class Profiler : Window

{

ApplicationContext db;

Order selectedOrder;

int UserId;

public Profiler(int UserId)

{

InitializeComponent();

this.UserId = UserId;

db = new ApplicationContext();

UserName.DataContext = db.Users.Find(UserId);

db.Orders.Load();

Orders.ItemsSource = db.Orders.Local.ToBindingList();

}

private void Accept\_Click(object sender, RoutedEventArgs e)

{

selectedOrder.Status = 2;

Accept.IsEnabled = false;

Decline.IsEnabled = false;

db.SaveChanges();

Orders.Items.Refresh();

db.Dispose();

Orders.SelectionChanged -= Orders\_SelectionChanged;

Orders.ItemsSource = null;

db = new ApplicationContext();

db.Orders.Where(i => UserId == i.UserId).Load();

Orders.ItemsSource = db.Orders.Local.ToBindingList();

Orders.SelectionChanged += Orders\_SelectionChanged;

}

private void Decline\_Click(object sender, RoutedEventArgs e)

{

selectedOrder.Status = 2;

selectedOrder.Sale = 0;

Accept.IsEnabled = false;

Decline.IsEnabled = false;

db.SaveChanges();

Orders.Items.Refresh();

db.Dispose();

Orders.SelectionChanged -= Orders\_SelectionChanged;

Orders.ItemsSource = null;

db = new ApplicationContext();

db.Orders.Where(i => UserId == i.UserId).Load();

Orders.ItemsSource = db.Orders.Local.ToBindingList();

Orders.SelectionChanged += Orders\_SelectionChanged;

}

private void Orders\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

selectedOrder = (Order)Orders.SelectedItem;

if (selectedOrder.Status == 1)

{

Accept.IsEnabled = true;

Decline.IsEnabled = true;

}

}

}

}

/// Код программы Registration

namespace SuperPizzeria

{

/// <summary>

/// Логика взаимодействия для Registration.xaml

/// </summary>

public partial class Registration : Page

{

Window wind;

public Registration(Window win)

{

InitializeComponent();

if (((AuthReg)win).IsEmployeeMode) Name.Content = "Регистрация сотрудника";

else Name.Content = "Регистрация";

wind = win;

}

private void Button\_Click(object sender, RoutedEventArgs e)

{

((AuthReg)wind).AuthRegFrame.Content = new Authorization(wind);

}

private void Button\_Click\_1(object sender, RoutedEventArgs e)

{

if (!Functions.VerifyPassword(PasswordField.Text))

{

MessageBox.Show("Введён некорретный пароль", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

return;

}

using (ApplicationContext db = new ApplicationContext())

{

var users = db.Users.Where(i => i.Login == LoginField.Text);

if (users.Count() > 0)

{

MessageBox.Show("Пользователи с таким логином уже существуют", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

return;

}

int statusOfUser = !((AuthReg)wind).IsEmployeeMode ? 1 : 2;

User user = new User { Login = LoginField.Text, Password = PasswordField.Text, Status = statusOfUser };

db.Users.Add(user);

db.SaveChanges();

MessageBox.Show($"Вы успешно зарегистрировались по ником {LoginField.Text}", "Поздравляю", MessageBoxButton.OK, MessageBoxImage.Information);

}

}

}

}

Приложение Б

(справочное)

Формы входных и выходных документов

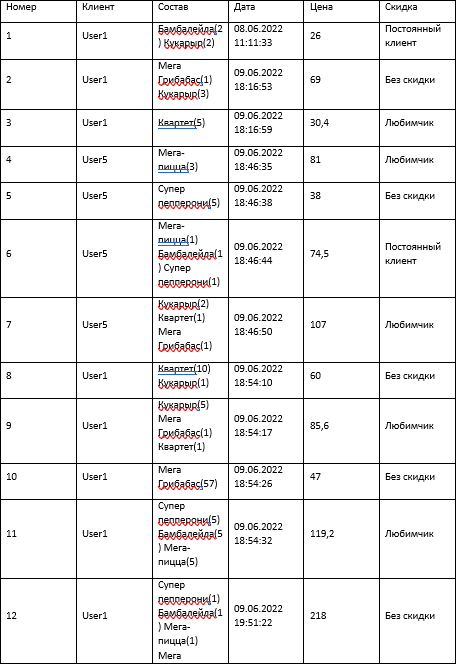


Рисунок Б.1

Приложение В

(справочное)

Результаты работы программы

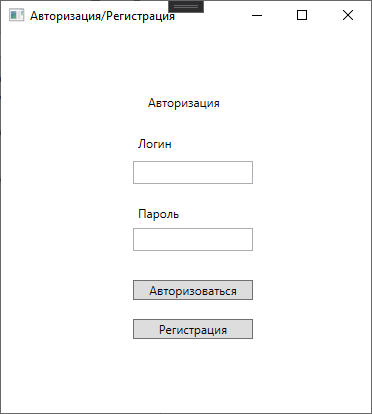


Рисунок В.1

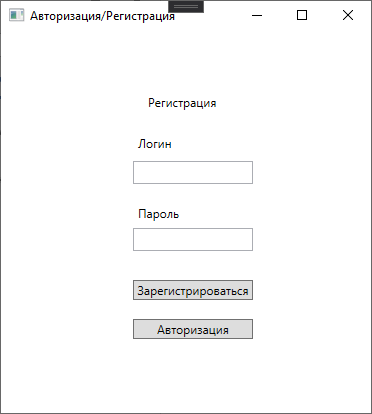


Рисунок В.2

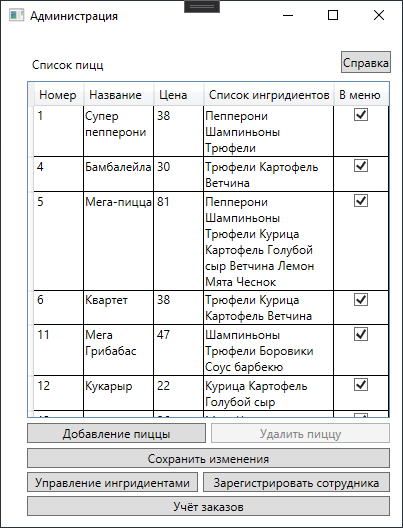


Рисунок В.3

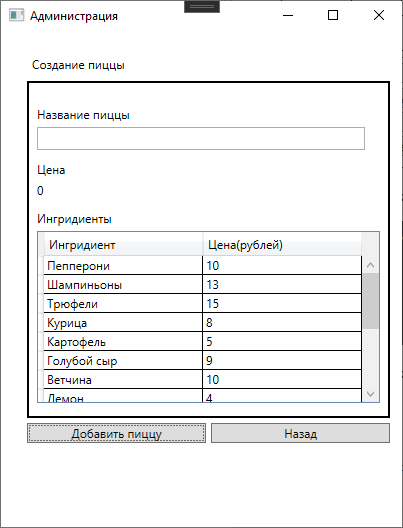


Рисунок В.4

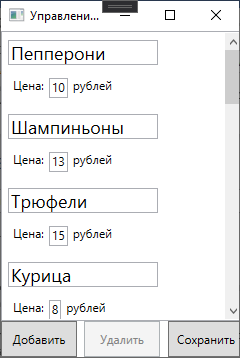


Рисунок В.5

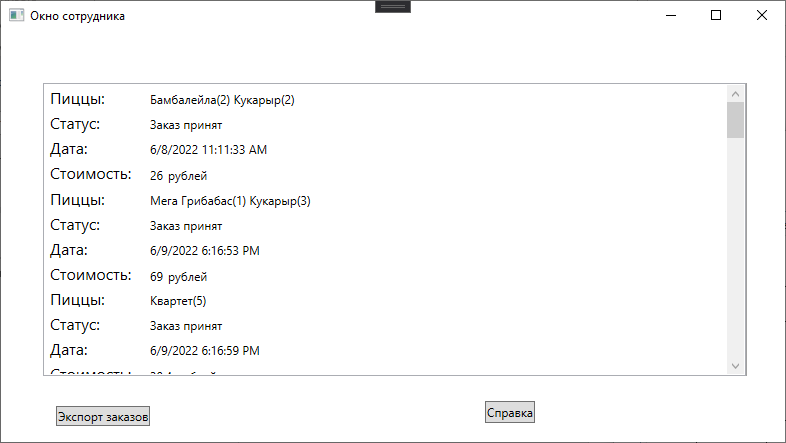


Рисунок В.6

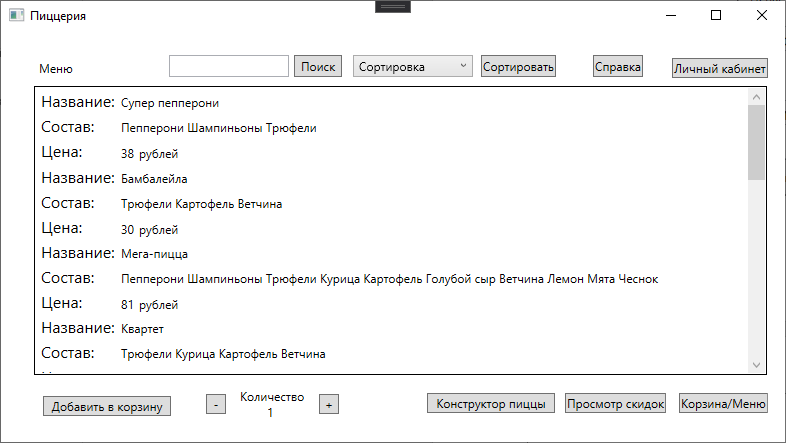


Рисунок В.7

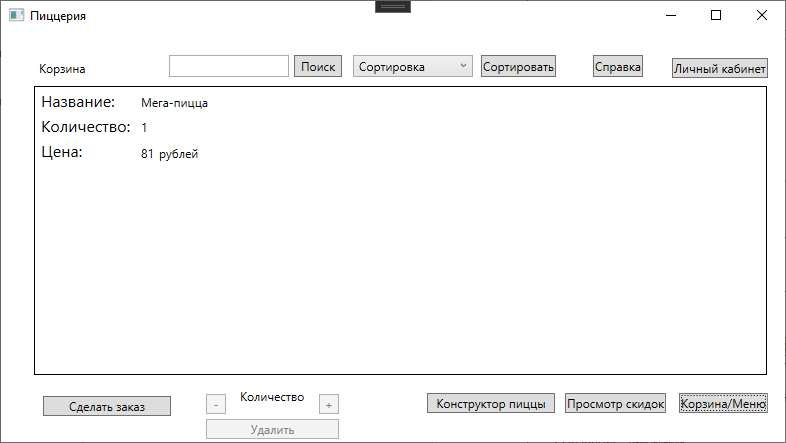


Рисунок В.8

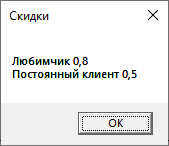


Рисунок В.9

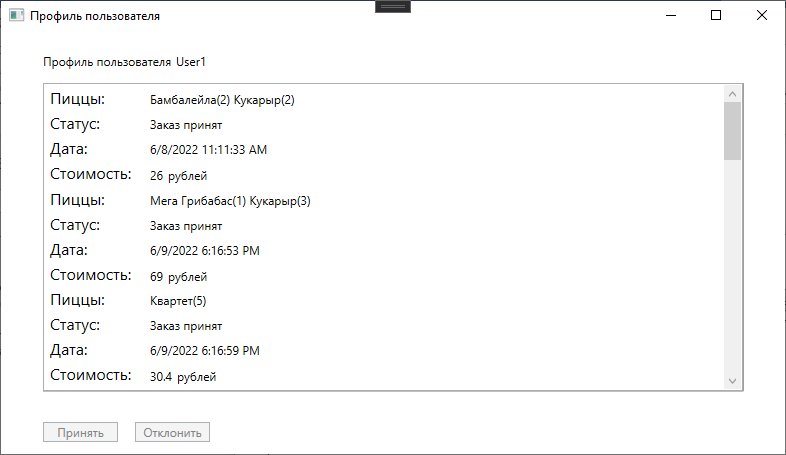


Рисунок В.10

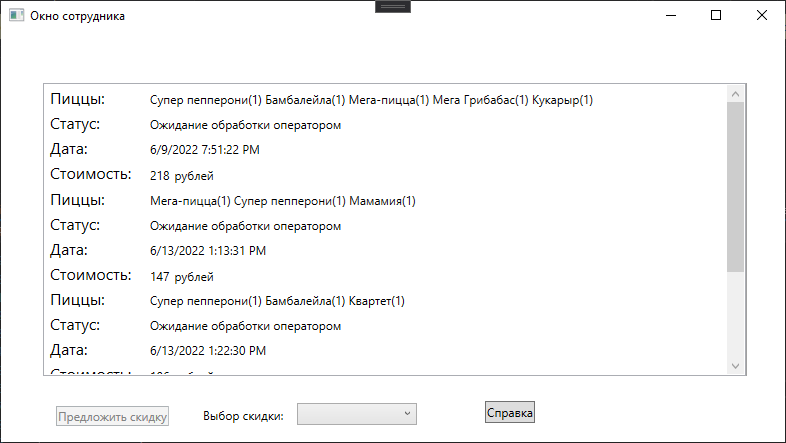


Рисунок В.11

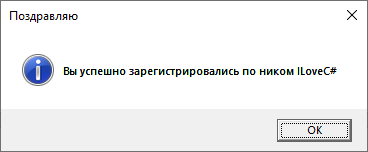


Рисунок В.12

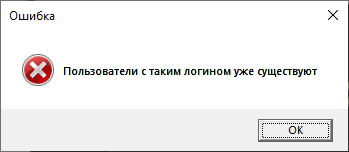


Рисунок В.13

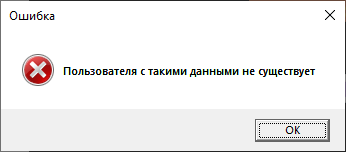


Рисунок В.14

**Удостоверяющий лист**

электронного документа – курсовой проект

Тема КП «Программное средство для учета заказов в пиццерии»

Обозначение \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ КП Т.x\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Разработчик x Руководитель x

(Ф.И.О.) (Ф.И.О.)

Председатель цикловой комиссии x

(Ф.И.О.)

Подписи лиц, ответственных за разработку электронного документа

|  |  |  |
| --- | --- | --- |
| Состав ЭД | Разработчик | Руководитель |
| Пояснительная записка (на  бумажном носителе формата  А4), файл ПЗ.docx, ПЗ.pdf |  |  |
| ГЧ №1, файл  Диаграмма ДВИ.drawio |  |  |
| ГЧ №2, файл  Диаграмма компонентов.drawio |  |  |
| ГЧ №3, файл  Диаграмма ERD.drawio |  |  |
| ГЧ №4, файл  Диграмма последовательности.drawio |  |  |
| ГЧ №5, файл  Диаграмма классов.drawio |  |  |
| ГЧ №6, файл  Диаграмма деятельности.drawio |  |  |
| Программная документация:  файлы CartUnit.cs, Functions.cs, Ingridient.cs, Order.cs, Pizza.cs, Sale.cs,  User.cs, Order.cs, OrderContext.cs, CarBrand.cs, CarBrandContext.cs,  ApplicationContext.cs, Administration.cs, Authorization.cs, AuthReg.cs, Ingridients.cs,  MainWindow.cs, OrdersAccounting.xaml, Profiler.xaml, Registration.cs. |  |  |
| Установочный пакет SuperPizzeria.exe |  |  |
| Тип носителя: USB-флеш-накопитель |  |  |

**Этикетка**

для курсовых проектов

**Курсовой проект**

Тема «Программное средство для учета заказов в пиццерии»

КП Т.x

Разработан x

Утвержден x

Разработчик: x

Руководитель: x

Технические средства x.

Программные средства: MS Visual Studio 2019 Community, MS SQL Server 2019,

C#, MS Windows 11, Draw.io, .NET Framework 6.0

**Состав документа:**

Пояснительная записка – ПЗ.docx, ПЗ.pdf

Программные документы – CartUnit.cs, Functions.cs, Ingridient.cs, Order.cs, Pizza.cs, Sale.cs,

User.cs, Order.cs, OrderContext.cs, CarBrand.cs, CarBrandContext.cs,

ApplicationContext.cs, Administration.cs, Authorization.cs, AuthReg.cs, Ingridients.cs,

MainWindow.cs, OrdersAccounting.xaml, Profiler.xaml, Registration.cs.

Графическая часть – файлы Диаграмма ERD.drawio, Диаграмма ДВИ.drawio,

Диаграмма деятельности.drawio, Диаграмма классов.drawio,

Диаграмма компонентов.drawio, Диаграмма последовательности.drawio.

Сведения о защите информации: отсутствуют