Частное учреждение образования

«Колледж бизнеса и права»

РАЗРАБОТКА ПРИЛОЖЕНИЯ ДЛЯ АВТОМАТИЗАЦИИ РАБОЧЕГО МЕСТА СОТРУДНИКА ОНЛАЙН-Ресторана БЫСТРОГО пиТАНия

Пояснительная записка

к курсовому проекту по предмету

«Конструирование программ и языки программирования»

КП Т.197020.401

Руководитель проекта (А.В.Кривошеина)

Обучающийся (Е.В.Тихомиров)

2024

СОДЕРЖАНИЕ

[Введение 4](#_Toc167632486)

Изм.

Лист

№ докум.

Подпись

Датаа

Лист

3

КП Т.197020.401

Разраб.

Тихомиров Е.В.

Провер.

Кривошеина А.В.

Реценз.

Н. Контр.

Утверд.

Лит.

Листов

КБП

у

[1 Описание задачи 6](#_Toc167632487)

[1.1 Анализ предметной области 6](#_Toc167632488)

[1.2 Постановка задачи 7](#_Toc167632489)

[2 Проектирование системы 8](#_Toc167632490)

[2.1 Требования к приложению 8](#_Toc167632491)

[2.2 Проектирование модели 9](#_Toc167632492)

[2.3 Концептуальный прототип 12](#_Toc167632493)

[3 Описание реализации программного средства 25](#_Toc167632494)

[3.1 Инструменты разработки и применяемые технологии 25](#_Toc167632495)

[3.2 Порядок авторизации пользователей 26](#_Toc167632496)

[3.3 Организация данных 27](#_Toc167632497)

[3.4 Функции: логическая и физическая организация 36](#_Toc167632498)

[3.5 Входные и выходные данные 40](#_Toc167632499)

[3.6 Функциональное тестирование 40](#_Toc167632500)

[3.7 Описание справочной системы 50](#_Toc167632501)

[4 Применение 51](#_Toc167632502)

[4.1 Назначение программного средства 51](#_Toc167632503)

[4.2 Условия применения 51](#_Toc167632504)

[Заключение 52](#_Toc167632505)

[Список использованных источников 53](#_Toc167632506)

[Приложение А Текст программных модулей 54](#_Toc167632507)

[Приложение Б Формы входных и выходных документов 57](#_Toc167632508)

[Приложение В Результаты работы программы 59](#_Toc167632509)

*Разработка автоматизированного рабочего места агента*

*по недвижимости*

# В ведение

В наше время доставка блюд из ресторанов быстрого питания становится все более востребованной услугой. Однако, для эффективной организации доставки, необходимо автоматизировать множество рутинных задач, таких как учет заказов, отслеживание клиентов. Применение специализированного программного обеспечения позволяет оптимизировать деятельность системы доставки заказов, повысить качество обслуживания клиентов и эффективность работы сотрудников.

Целью курсового проектирования является создание приложения для автоматизации учета заказов онлайн-ресторана. Основные задачи приложения включают в себя:

ведение базы данных заказов;

отслеживание статуса заказов и их состав;

формирование учета склада;

анализ статистики деятельности онлайн-ресторана.

Актуальность разработки данного приложения обусловлена необходимостью повышения эффективности работы работников онлайн-ресторана за счет автоматизации ключевых бизнес-процессов. Использование специализированного программного обеспечения позволит сотрудникам оптимизировать свою работу, снизить вероятность ошибок и улучшить качество обслуживания клиентов.

В ходе курсового проектирования будут рассмотрены вопросы проектирования, разработки и внедрения приложения для автоматизации рабочего места сотрудника онлайн-ресторана. Пояснительная записка включает в себя описание задачи, проектирование системы, реализацию программного средства и область его применения.

В приложении А представлен текст программных модулей приложения.

В приложении Б представлен результаты работы программы.

# Описание задачи

## Анализ предметной области

Предметной областью решаемой задачи является онлайн-ресторан и его деятельность. Объектом решаемой задачи является автоматизация процесса ведения учета деятельности сотрудников бюро ремонтов электронной техники и их взаимодействия с клиентами.

Автоматизированное рабочее место (АРМ) сотрудника онлайн-ресторана - это программное обеспечение, разработанное для автоматизации и упрощения работы специалистов. Оно обычно включает в себя набор инструментов и функций, которые помогают сотрудникам эффективно управлять своими задачами и обязанностями, связанными с приемом, обработкой, готовкой и доставкой заказов.

Сотрудник онлайн-ресторана - это специалист, задача которого принять заказ, приготовить его, и доставить его клиенту. Суть работы сотрудника заключается в умении максимально быстро и качественно выполнить приготовление заказа, а также выстроить эффективное взаимодействие с клиентами. Каждый сотрудник обладает своим списком заказов, а также клиентами, с которыми он работает, в связи с чем будет целесообразно организовать отображение у каждого сотрудника только своих данных, которые он добавил.

Компьютерная обработка информации и процессов не только обеспечивает эффективное выполнение задач сотрудников онлайн-ресторана, но и позволяет улучшить качество обслуживания клиентов, оптимизировать процессы и принимать обоснованные управленческие решения, ниже перечислены ее преимущества в данной сфере:

Учет и хранение информации о заказах. АРМ позволяет вести базу данных заказов, фиксировать поступающие заявки, отслеживать статус приготовления и историю взаимодействия с каждым клиентом.

Управление процессом ремонта. АРМ автоматизирует такие задачи, как распределение заявок между сотрудниками, учет ингредиентов на складе.

Анализ эффективности работы онлайн-ресторана. АРМ предоставляет руководству всестороннюю аналитическую информацию - статистику по выполненным заказам, количество заказов и т.д. Это позволяет оптимизировать бизнес-процессы и принимать обоснованные управленческие решения.

Повышение качества обслуживания клиентов. АРМ обеспечивает прозрачность процесса ремонта для клиентов, позволяет своевременно информировать их о статусе заявки и сроках исполнения. Это способствует росту лояльности клиентов и улучшению репутации бюро ремонтов.

Организация взаимодействия сотрудников. АРМ поддерживает обмен информацией между специалистами, упрощает координацию их действий и контроль со стороны руководства.

Таким образом, разработка АРМ для онлайн-ресторана позволит повысить эффективность, прозрачность и управляемость всех бизнес-процессов, обеспечить высокое качество обслуживания клиентов.

## Постановка задачи

Функции, которые должны быть автоматизированы для онлайн-ресторана:

* Программное средство должно позволять пользователю вводить данные о заказе, такие как ФИО, адрес, список блюд, повара и курьера.
* Создание отдельных учетных записей для польователей.
* Отслеживание процесса приготовления и доставки заказа.
* Просмотр продуктивности сотрудников.
* Просмотр истории выполненных заказов.
* Удаление пользователей из базы данных.

Интерфейс приложения должен быть максимально простым и интуитивно понятным для сотрудников онлайн-ресторана, обеспечивая комфортное взаимодействие.

В отличие от существующих CRM-систем для сферы недвижимости, данное программное средство будет специально разработано для нужд онлайн-ресторана, включая автоматизацию составления заказов и анализ статистики.Проектирование системы

# Требования к приложению

Средства защиты будущего приложения:

* для обеспечения безопасности приложения и защиты от несанкционированного доступа необходимо использовать механизмы авторизации. Это позволит убедиться в подлинности пользователей и управлять доступом к данным и функциям приложения;
* для хранения данных о клиентах и пользователях можно использовать защищенную базу данных, которая содержит ограничения для доступа и хранимой информации.

Общие требования:

* контент приложения должен содержать в обязательном порядке список заказов и работников;
* интерфейс программного средства должен быть в темных оттенках, а цветовая палитра должна соответствовать требованиям заказчика;
* интерфейс программного средства должен обеспечивать необходимую функциональность для ведения учёта заказов;
* интерфейс должен быть информативным и отображать данные в удобном и понятном виде. Это позволит быстро и эффективно искать, а также обрабатывать данные;
* интерфейс должен быть простым и интуитивно понятным для пользователя. Это позволит быстро и эффективно заниматься рутинной работой – интерфейс должен быть доступен для пользователей с различным уровнем компьютерной грамотности;
* расположение элементов управления должно быть логичным и удобным для пользователя;
* размер элементов управления должен быть достаточным для удобного взаимодействия с пользователем;
* форма и стиль элементов управления должны соответствовать требованиям дизайна и общей концепции приложения;
* шрифт должен быть читабельным и соответствовать дизайну приложения.

Требования к аппаратным и операционным ресурсам:

* операционная система: Windows 7, 8, 8.1, 10, 11;
* объем памяти устройства: не менее 2 Гб;
* доступ к интернету и его скорость: для работы приложения требуется доступ в интернет;
* дополнительные устройства (мышь, клавиатура, принтер и т.д.).

## Проектирование модели

Исходя из исследования предметной области можно выделить следующие сущности проекта.

Каждый администратор характеризуется следующими данными:

* логин;
* пароль;

Каждый повар, с которым работал администратор, характеризуется следующими данными:

* идентификационный номер;
* логин;
* пароль;
* количество выполненных заказов;

Каждый курьер, с которым работал администратор, характеризуется следующими данными:

* идентификационный номер;
* логин;
* пароль;
* количество доставленных заказов;

Каждый заказ характеризуется следующими данными:

* номер заказа;
* кодировка заказа;
* имя клиента;
* адрес клиента;
* номер повара, ответственного за заказ;
* номер курьера, ответственного за заказ;

Основные задачи:

* Добавление новых сотрудников и информации о них в базу;
* Добавление новых заказов и информации о них в базу;
* Упрощение процесса передачи заказов сотрудникам;

Таким образом, сотрудник онлайн-ресторана сможет эффективно управлять информацией о заказах и сотрудниках, что позволит автоматизировать и оптимизировать его рабочий процесс.

Диаграмма «Сущность-связь» представлена на рисунке 2.2.1.

Изображение выглядит как снимок экрана, круг, черно-белый, текст

Автоматически созданное описание

Рисунок 2.2.1 – Диаграмма «Сущность-связь» в нотации Чена

Диаграмма вариантов использования (ДВИ) – это инструмент, используемый для моделирования функциональных требований системы. Они описывают взаимодействие внешних пользователей или других систем (актеров) с системой для достижения конкретных целей (вариантов использования). Данная диаграмма для разрабатываемого средства представлена на первом листе графической части.

Диаграмма классов – это диаграмма, которая используется для визуализации структуры системы путем отображения ее классов, атрибутов, методов и отношений между классами. Диаграмма классов для разрабатываемого средства представлена на втором листе графической части.

Диаграмма деятельности – это один из типов диаграмм UML, используемый для моделирования динамических аспектов системы. Она описывает последовательность действий или потока управления в системе и используется для визуализации бизнес-процессов, логики работы алгоритмов, а также рабочих процессов в программных системах. Диаграмма деятельности для функции составления заказа представлена на третьем листе графической части.

Диаграмма последовательности – это еще один из типов диаграмм UML, который используется для представления взаимодействий между объектами в системе в хронологическом порядке, показывая, как объекты обмениваются сообщениями для выполнения функциональности. Диаграмма последовательности для функции составления заказа представлена на четвертом листе графической части.

## Концептуальный прототип

Содержимое окна авторизации:

* текстовое поле «Username:» будет позволять вводить логин учетной записи;
* текстовое поле «Password» будет позволять ввести пароль от аккаунта;
* три переключателя, «Администратор», «Повар» и «Курьер» будут осуществлять регистрацию под определенную роль.
* кнопка «Login», оформляет вход в систему под данными, указаными в поля «Username» и «Password»

На рисунке 2.3.1 представлен прототип окна авторизации.

Изображение выглядит как текст, снимок экрана, дизайн

Автоматически созданное описание

Рисунок 2.3.1 – Макет окна авторизации

Содержимое окна администратора:

* Левый список вкладок(окон) для пользования приложением;
* таблицы будут использоваться для отображения данных на форме.

На рисунках 2.3.2-2.3.8 представлены макеты главного окна управления таблицами с различными данными (сотрудники, заказы, выполненные заказы, склад).

Изображение выглядит как снимок экрана, фиолетовый, текст, Фиолетовый

Автоматически созданное описание

Рисунок 2.3.2 – Макет окна администратора

Содержимое окна для добавления работника:

* три кнопки добавления с выбором администратора, повара или курьера;
* текстовое поле «Логин» будет позволять ввести логин сотрудника;
* текстовое поле «Пароль» будет позволять ввести пароль учетной записи сотрудника;
* текстовое поле «Ваш пароль» используется для ввода пароля текущего администратора для подтверждения создания учетной записи сотрудника;

Изображение выглядит как текст, снимок экрана, фиолетовый, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок 2.3.3 – Макет окна создания учетных записей для сотрудников

Содержимое окна для удаления работника:

* три кнопки удаления с выбором администратора, повара или курьера;
* текстовое поле «Логин» будет позволять ввести логин сотрудника;
* текстовое поле «Пароль» будет позволять ввести пароль учетной записи сотрудника;
* текстовое поле «Ваш пароль» используется для ввода пароля текущего администратора для подтверждения удаления учетной записи сотрудника;

Изображение выглядит как текст, снимок экрана, фиолетовый, Фиолетовый

Автоматически созданное описание

Рисунок 2.3.4 – Макет окна удаления учетной записи сотрудника

Содержимое окна для просмотра работников:

* три кнопки с выбором администратора, повара или курьера;

Изображение выглядит как снимок экрана, текст, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок 2.3.5 – Макет окна таблицы с клиентами

Содержимое окна для добавления заказа:

* десять кнопок для добавления блюд;
* текстовое поле «Имя» будет позволять ввести имя клиента;
* текстовое поле «Адрес» будет позволять ввести адрес клиента;
* текстовое поле «ID Повара» используется для назначения повара на данный заказ;
* текстовое поле «ID Курьера» используется для назначения курьера на данный заказ;
* кнопка «Добавить заказ» отправляет заказ на кухню;
* кнопка «Обновить» обновляет таблицу заказов;

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок 2.3.6 – Макет окна таблицы c заказами

Изображение выглядит как текст, снимок экрана, Мультимедийное программное обеспечение, фиолетовый

Автоматически созданное описание

Рисунок 2.3.7 – Макет окна таблицы с ингредиентами

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок 2.3.8 – Макет окна таблицы с выполненными заказами

Изображение выглядит как текст, снимок экрана, линия, дизайн

Автоматически созданное описание

Рисунок 2.3.8 – Макет окна таблицы с заказами

# Описание реализации программного средства

## Инструменты разработки и применяемые технологии

Исходя из анализа предметной области и рассмотренных функциональных возможностей приложения автор курсового проекта принял решение о выборе следующих инструментов разработки.

В качестве среды разработки использовать интегрированную среду разработки (IDE) Microsoft Visual Studio. Это позволит эффективно создавать, отлаживать и собирать приложение, а также использовать множество инструментов и функций, доступных в Visual Studio. [7]

Эта среда разработки предоставляет возможность разрабатывать средства на таких языках как C++, C#, F#, JavaScript и Python.

Программа использует язык программирования C# с использованием WPF для создания пользовательского интерфейса. Также она использует СУБД MSSQL Server для хранения и управления данными. [1], [8], [6]

Одним из ключевых элементов разработки на C# является использование библиотеки .NET Framework, которая предоставляет широкий спектр функциональности для разработки приложений под Windows. Кроме того, для работы с данными в приложении вы можете использовать ADO.NET, что позволит вам удобно взаимодействовать с базой данных MSSQL Server.

Примеры приложений, которые можно разработать с использованием данных технологий, включают учетные системы, системы управления заказами, приложения для управления складом и другие десктопные приложения, требующие работы с данными.

В проектируемом программном обеспечении данные технологии могут использоваться следующим образом:

* использование WPF для создания интуитивно понятного и простого в использовании пользовательского интерфейса;
* использование MSSQL Server для хранения и управления данными, такими как информация о пользователях, заказах, транзакциях и т.д.;
* использование ADO.NET для удобной работы с данными и обеспечения совместимости с базой данных; [3]

WPF (Windows Presentation Forms) обеспечивает создание интуитивно понятных и простых пользовательских интерфейсов для приложений Windows. Этот инструмент позволяет разработчикам быстро и легко создавать приложения, используя элементы управления, такие как кнопки, текстовые поля и таблицы, что делает его идеальным для бизнес-приложений.

MSSQL Server – мощная система управления базами данных от Microsoft, используется для хранения и управления данными, такими как информация о пользователях, заказах и транзакциях. Она обеспечивает высокую производительность, безопасность и надежность данных, а также предлагает инструменты для администрирования и резервного копирования.

**ADO.NET** (ActiveX Data Object для .NET) — технология, предоставляющая доступ к данным и управление ими, хранящимися в базе данных или других источниках (Microsoft SQL Server, Microsoft Access, Microsoft Excel, Microsoft Outlook, Microsoft Exchange, Oracle, OLE DB, ODBC, XML, текстовые файлы), основанных на платформе .NET Framework и входящая в состав .NET Framework 2.0, представляет собой набор библиотек.

## Порядок авторизации пользователей

В разрабатываемом программном средстве предусмотрено разделение ролей на администратора, повара и курьера. Повар – это сотрудник, который принимает заказ; после приготовления, заказ передается курьеру. Курьер ­– это сотрудник, который принимает готовый заказ с кухни для доставки клиенту. Администратор – это сотрудник сервиса, который занимается наблюдением за активностью сотрудников.

Авторизованному повару доступен тот же функционал по приготовлению заказа, что и курьеру, с той лишь разницей что он помечает заказы доставленными, после чего они не будут отображаться в таблице текущих.

Администратору будет доступен функционал добавления новых сотрудников, удаление текущих сотрудников, просмотр информации о них, добавление заказов, просмотр текущих заказов, просмотр склада, просмотр выполненных заказов.

Вход в систему осуществляется путем ввода логина, пароля и выбора окна, в случае если введенный телефон существует и пароль совпадает, пользователь получает доступ к функционалу приложения. Возможность самостоятельной регистрации отсутствует. Все новые пользователи добавляются через аккаунт администратора.

## 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 вторичные ключи. Достоинством реляционной модели является простота и удобство физической реализации.

Реляционная модель базы данных подразумевает нормализацию всех таблиц данных. Нормализация – это формальный метод анализа отношений на основе их первичного ключа и функциональных зависимостей, существующих между их атрибутами. Структура базы данных разрабатываемого программного средства включает пять таблиц.

Таблица «orders» хранит информацию о заказах. Структура приведена в в таблице 3.1.

Таблица 3.1 – Структура таблицы «Orders»

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Тип данных | Размер, байт | Описание |
| OrderID | INT | 8 | Идентификационный номер |
| OrderCode | NVARCHAR(20) | 20 | Код заказа |
| OrderName | NVARCHAR(20) | 20 | Имя клиента |
| Address | NVARCHAR(50) | 50 | Адрес клиента |
| CookID | INT | 8 | Идентификационный номер повара |
| CourierID | INT | 8 | Идентификационный номер курьера |
| IsCooked | INT | 8 | Отображение приготовленности заказа |
| IsDelivered | INT | 8 | Отображение состояния доставки заказа |

Таблица «User1» хранит информацию о мастерских . Структура приведена в таблице 3.2.

Таблица 3.2 – Структура таблицы «User1»

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Тип данных | Размер, байт | Описание |
| ID | INTEGER | 8 | Идентификационный номер |
| Username | NVARCHAR(20) | 20 | Логин |
| Password | INT | 8 | Пароль |
| Completed | INT | 8 | Количество выполненных заказов |

Таблица «User2» хранит информацию о всех сотрудников, структура приведена в таблице 3.3.

Таблица 3.3 – Структура таблицы «User2»

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Тип данных | Размер, байт | Описание |
| ID | INTEGER | 8 | Идентификационный номер |
| Username | NVARCHAR(20) | 20 | Логин |
| Password | INT | 8 | Пароль |
| Completed | INT | 8 | Количество выполненных заказов |

Таблица «Admin» хранит информацию о клиентах, структура приведена в таблице 3.4.

Таблица 3.4 – Структура таблицы «Admin»

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Тип данных | Размер, байт | Описание |
| Username | NVARCHAR(20) | 8 | Логин |
| Password | TEXT | 32 | Пароль |

Таблица «Products» хранит информацию о продуктах, структура приведена в таблице 3.5.

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Тип данных | Размер, байт | Описание |
| Buns | INT | 8 | Кол-во булочек |
| Tomatoes | INT | 8 | Кол-во помидоров |
| Cheese | INT | 8 | Кол-во сыра |
| Salad | INT | 8 | Кол-во салата |
| Patties | INT | 8 | Кол-во булочек |
| Onions | INT | 8 | Кол-во булочек |
| Fries | INT | 8 | Кол-во булочек |

На рисунке 3.3.1 представлена структура таблиц базы данных.

Изображение выглядит как текст, диаграмма, число, Шрифт

Автоматически созданное описание

Рисунок 3.3.1 – Структура таблиц базы данных

## Функции: логическая и физическая организация

На диаграмме вариантов использования представлен весь функционал пользователя и администратора.

Основной функцией для работы с базой данных является функция добавления данных в БД.

Полный код программы, вместе с функциями представлен в приложении А.

## Входные и выходные данные

Входные и выходные данные должны быть представлены в приложении Б. Но т.к. не было реализованно отчет о успешном заказе, такие данные отсутсвуют.

## Функциональное тестирование

Функциональное тестирование является важной частью процесса обеспечения качества программного обеспечения. Оно направлено на проверку функциональности системы с целью убедиться, что она соответствует требованиям и ожиданиям пользователей. Прежде чем перейти к разработке тест-кейсов, необходимо провести анализ требований, понять функциональность приложения и определить основные сценарии использования. Это позволит сосредоточиться на наиболее важных и критических аспектах приложения. Подходящие тест-кейсы должны быть разработаны для каждого сценария использования, чтобы проверить, что функции приложения работают правильно и соответствуют ожиданиям пользователей. На таблицах 3.6 - 3.6.6 представлен набор тест-кейсов для программного средства.

Таблица 3.6 – Тест-кейс для функции авторизации администратора в приложении

|  |  |  |
| --- | --- | --- |
| Действие | Ожидаемый результат | Результат теста |
| Открыть приложение «SynthFoods» | Приложение «SynthFoods» открыто и доступно | Пройден |
| Ввести в поле «Username» и «Password» строки «SanyaPlot», «bekerov19» | Отображение введенного логина и пароля в поле | Пройден. |
| Нажать на кнопку «Повар» и «Войти» | Открыта страница повара | Пройден. |

Таблица 3.7 – Тест-кейс для функции авторизации администратора в приложении

|  |  |  |
| --- | --- | --- |
| Действие | Ожидаемый результат | Результат теста |
| Открыть приложение «SynthFoods» | Приложение «SynthFoods» открыто и доступно | Пройден |
| Ввести в поле «Username» и «Password» строки «DanyaSpeedster», «420Ss8420» | Отображение введенного логина и пароля в поле | Пройден. |
| Нажать на кнопку «Курьер» и «Войти» | Открыта страница курьера | Пройден. |

Таблица 3.8 – Тест-кейс для функции авторизации администратора в приложении

|  |  |  |
| --- | --- | --- |
| Действие | Ожидаемый результат | Результат теста |
| Открыть приложение «SynthFoods» | Приложение «SynthFoods» открыто и доступно | Пройден |
| Ввести в поле «Username» и «Password» строки «hz», «12345678» | Отображение введенного логина и пароля в поле | Пройден. |
| Нажать на кнопку «Администратор» и «Войти» | Открыта страница администратора | Пройден. Результат изображен на картинке Б.2 |

Таблица 3.9 – Тест-кейс для функции оповещения о неправильных данных при авторизации

|  |  |  |
| --- | --- | --- |
| Действие | Ожидаемый результат | Результат теста |
| Открыть приложение «SynthFoods» | Приложение «SynthFoods» открыто и доступно | Пройден |
| Ввести в поле «Username» и «Password» строки «hzzzzzz», «1234» | Отображение введенного логина и пароля в поле | Пройден. |
| Нажать на кнопку «Администратор» и «Войти» | Оповещение о неверном вводе. | Пройден. |

Таблица 3.10 – Тест-кейс для функции добавления пользователя в базу данных

|  |  |  |
| --- | --- | --- |
| Действие | Ожидаемый результат | Результат теста |
| Войти в форму администратора | Форма администратора открыта | Пройден |
| Нажать на кнопку «Добавить пользователя» | Форма добавления пользователя | Пройден. |
| Ввести в текстовые поля логин, пароль и пароль администратора в соответствующие поля | Данные введены | Пройден. |
| Нажать кнопку «Курьер» и «Добавить» | Пользователь типа «Курьер» добавлен | Пройден. |

Таблица 3.11 – Тест-кейс для функции удаления пользователя из базы данных

|  |  |  |
| --- | --- | --- |
| Действие | Ожидаемый результат | Результат теста |
| Войти в форму администратора | Форма администратора открыта | Пройден |
| Нажать на кнопку «Удалить пользователя» | Форма удаления пользователя | Пройден. |
| Ввести в текстовые поля логин, пароль и пароль администратора в соответствующие поля | Данные введены | Пройден. |
| Нажать кнопку «Курьер» и «Удалить» | Пользователь типа «Курьер» удален | Пройден. |

Таблица 3.11 – Тест-кейс для функции удаления пользователя из базы данных

|  |  |  |
| --- | --- | --- |
| Действие | Ожидаемый результат | Результат теста |
| Войти в форму администратора | Форма администратора открыта | Пройден |
| Нажать на кнопку «Добавить заказ» | Форма добавления заказа | Пройден. |
| Ввести в текстовые поля имя, адрес, ID повара и ID курьера соответствующие поля | Данные введены | Пройден. |
| Нажать кнопку «Добавить» | Заказ добавлен | Пройден. |

## Описание справочной системы

Для разработки справочной системы ничего не использовалось.

## Назначение программного средства

Название разработанного программного средства – «SynthFoods».

Назначение программного средства – автоматизация рабочего места сотрудника бюро ремонтов электронной техники.

Реализованные функции:

* ввод данных о администраторах, курьерах и поварах;
* редактирование данных о сотрудниках и заказах;
* просмотр текущих и выполненных заказов;

Ограничения на область применения программного средства включают различные аспекты, которые необходимо учитывать при его использовании. Эти ограничения могут быть связаны с техническими, функциональными и организационными аспектами. Технические аспекты включают в себя совместимость с оборудованием, операционная система, производительность, интернет-соединение (требуется для соединения с базой данных). К функциональным ограничениям относится объем хранимых данных. К организационным ограничениям можно отнести: обучение пользователей работе с АРМ, поддержка и обслуживание программного средства, стоимость.

## Условия применения

Описание программного обеспечения, которое необходимо для функционирования программы:

* ПС разработано для работы на операционной системе Windows;
* для разработки ПС могут использоваться различные библиотеки и фреймворки на языке программирования C#;
* для разработки ПС может использоваться интегрированная среда разработки (IDE) на языке программирования C#, в данном случае, Visual Studio;
* для работы ПС могут потребоваться дополнительные приложения, такие как браузер для доступа к интернету, текстовый редактор для редактирования конфигурационных файлов Microsoft Word;
* для хранения клиентов и другой информации, связанной с ремонтной деятельностью, может потребоваться база данных. Для этого может использоваться различное программное обеспечение, такое как MSSQL Server.

## Заключение

В ходе разработки программного средства была поставлена задача разработать автоматизированное рабочее место сотрудника онлайн-ресторана быстрого питания.

Для выполнения данной задачи было разработано программное средство, которое позволяет пользователю вводить данные о работниках (логин и пароль), а затем добавляет данные в базу данных.

Для создания программного средства был использован язык программирования С#. Все поставленные задачи были выполнены успешно, и программное средство было реализовано в полном объеме.

Разработанные функции включают в себя функции ввода данных, функции просмотра и удаления.

Преимуществами разработанного программного средства являются его удобный графический интерфейс, возможность быстрого и удобного поиска информации.Одним из недостатков может быть сложность использования для пользователей, которые используют приложение на нескольких устройствах.

Для дальнейшего усовершенствования программного средства можно рассмотреть возможность добавления новых функций, таких как расчеты, кроссплатформенность, более удобный интерфейс, а также оптимизация приложения для быстрой и более надежной работы. Для достижения этих целей можно использовать новые средства, технологии и способы разработки программного обеспечения.

# Список использованных источников

1. Общие сведения о C# [Электронный ресурс]. – Режим доступа : https://learn.microsoft.com/ru-ru/dotnet/csharp/. - Дата доступа : 26.05.2024.
2. Общие сведения о Dr.Explain [Электронный ресурс]. – Режим доступа : https://www.drexplain.ru/. - Дата доступа : 10.06.2024.
3. ADO.NET [Электронный ресурс]. – Режим доступа : https://learn.microsoft.com/ru-ru/dotnet/framework/data/adonet/. – Дата доступа 28.05.2024.
4. Общие сведения о LINQ [Электронный ресурс]. – Режим доступа : https://learn.microsoft.com/ru-ru/dotnet/csharp/linq/. – Дата доступа 28.05.2024.
5. Общие сведения о ReportViewer [Электронный ресурс]. – Режим доступа : https://learn.microsoft.com/ru-ru/sql/reporting-services/application-integration/using-the-winforms-reportviewer-control?view=sql-server-ver16 - Дата доступа : 26.05.2024.
6. Общие сведения о SQL Server Management Studio [Электронный ресурс]. – Режим доступа : https://learn.microsoft.com/ru-ru/sql/ssms/. – Дата доступа 28.05.2024.
7. Общие сведения о Visual Studio [Электронный ресурс]. – Режим доступа : https://visualstudio.microsoft.com/ru/. – Дата доступа 10.06.2024.
8. Руководство по классическим приложениям [Электронный ресурс]. – Режим доступа : https://learn.microsoft.com/ru-ru/dotnet/desktop/wpf/overview/?view=netdesktop-8.0 – Дата доступа 28.05.2024.

# Приложение А

**(обязательное)  
Текст программных модулей**

Код окна логина.

namespace WpfApp6

{

public partial class MainWindow : Window

{

private DatabaseHelper db;

public MainWindow()

{

InitializeComponent();

db = new DatabaseHelper();

BtnLogin.Visibility = Visibility.Hidden;

string connectionString = "Server=ABUBACHIRTRAHOV\\BDISUBIDE; Initial Catalog=synthfoods; Integrated Security = True";

}

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

{

if (AdmnRB.IsChecked == true)

{

if (db.CheckUserAdmin(txtUsername.Text, txtPassword.Text))

{

string adminInfoPW = txtPassword.Text;

string adminInfoUN = txtUsername.Text;

MessageBox.Show("Login successful!");

Window1 am = new Window1(adminInfoUN, adminInfoPW, db, this);

am.WindowStartupLocation = WindowStartupLocation.CenterScreen;

am.Show();

this.Hide();

}

else { MessageBox.Show("Login data not found."); }

}

else if (User1RB.IsChecked == true)

{

if (db.CheckUser1(txtUsername.Text, txtPassword.Text))

{

OrderViewer u1 = new OrderViewer(this, txtUsername.Text);

u1.WindowStartupLocation = WindowStartupLocation.CenterScreen;

u1.Show();

this.Hide();

}

else { MessageBox.Show("Login data not found."); }

}

else

{

if (db.CheckUser2(txtUsername.Text, txtPassword.Text))

{

MenuWindow mw = new MenuWindow(this, txtUsername.Text);

mw.WindowStartupLocation = WindowStartupLocation.CenterScreen;

mw.Show();

this.Hide();

}

else { MessageBox.Show("Login data not found."); }

}

}

private void txtUsername\_Cl(object sender, MouseEventArgs e)

{

txtUsername.Text = "";

}

public void ClearForAdminExit()

{

txtUsername.Text = "";

txtPassword.Text = "";

AdmnRB.IsChecked = false;

}

private void txtPassword\_Cl(object sender, MouseEventArgs e)

{

txtPassword.Text = "";

}

private void RBCheck(object sender, RoutedEventArgs e)

{

if(AdmnRB.IsChecked == true || User1RB.IsChecked == true|| User2RB.IsChecked == true)

{

BtnLogin.Visibility = Visibility.Visible;

}

}

}

}  
Код окна администратора:

namespace WpfApp6

{

/// <summary>

/// Логика взаимодействия для Window1.xaml

/// </summary>

public partial class Window1 : Window

{

public static bool masteradmin = false;

private string masterdaminpw = "m4s73r4dm1nm010d3c";

private string connString;

private string[] dishName = new string[] { "Cheeseburger", "Diablo", "Double Cheeseburger", "Cheeseburger Tower", "Ultra Burger", "Coolah", "Fries", "Sundae Icecream", "Cappucino", "Caesar Salad"};

private int[] quantityList = new int[] { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };

private string orderString = "";

public Window1()

{

InitializeComponent();

string connectionString = "Server=ABUBACHIRTRAHOV\\BDISUBIDE; Initial Catalog=synthfoods; Integrated Security = True";

}

private MainWindow loginwindow;

private MaterialAmount ma = new MaterialAmount();

public Window1(string un, string pw, DatabaseHelper dbhelp, MainWindow e)

{

InitializeComponent();

string connectionString = "Server=ABUBACHIRTRAHOV\\BDISUBIDE; Initial Catalog=synthfoods; Integrated Security = True";

connString = connectionString;

db = dbhelp;

adminPW = pw;

loginwindow = e;

}

private DatabaseHelper db;

private string adminPW;

private void ShowAddUserMenu(object sender, RoutedEventArgs e)

{

addUserGroup.Visibility = Visibility.Visible;

addUser\_pw.Visibility = Visibility.Visible;

addUser\_un.Visibility = Visibility.Visible;

adminButton.Visibility = Visibility.Visible;

user1Button.Visibility = Visibility.Visible;

user2Button.Visibility = Visibility.Visible;

verifyField.Visibility = Visibility.Visible;

addUser.Visibility = Visibility.Visible;

AdminDeletingRB.Visibility = Visibility.Hidden;

CookDeletingRB.Visibility = Visibility.Hidden;

CourierDeletingRB.Visibility = Visibility.Hidden;

UsernameOfDeletingUser.Visibility = Visibility.Hidden;

PasswordOfDeletingUser.Visibility = Visibility.Hidden;

PasswordVerifyForDelete.Visibility = Visibility.Hidden;

UserDeletion.Visibility = Visibility.Hidden;

delpwlbl.Visibility = Visibility.Hidden;

delunlbl.Visibility = Visibility.Hidden;

delvpwlbl.Visibility = Visibility.Hidden;

dataGridRegistredUsers.Visibility = Visibility.Hidden;

ShowAdminsInDataGrid.Visibility = Visibility.Hidden;

ShowCooksInDataGrid.Visibility = Visibility.Hidden;

ShowCouriersInDataGrid.Visibility = Visibility.Hidden;

OrderGrid.Visibility = Visibility.Hidden;

NameBox.Visibility = Visibility.Hidden;

AddressBox.Visibility = Visibility.Hidden;

OrderConfirm.Visibility = Visibility.Hidden;

UpdateTable.Visibility = Visibility.Hidden;

OrderDataGrid.Visibility = Visibility.Hidden;

CourierIDTB.Visibility = Visibility.Hidden;

CookIDTB.Visibility = Visibility.Hidden;

nl.Visibility = Visibility.Hidden;

al.Visibility = Visibility.Hidden;

icl.Visibility = Visibility.Hidden;

icoul.Visibility = Visibility.Hidden;

StorageDataGrid.Visibility = Visibility.Hidden;

UpdateStorageGrid.Visibility = Visibility.Hidden;

MasterAdminPassword.Visibility = Visibility.Hidden;

ConfirmMasterAdmin.Visibility = Visibility.Hidden;

CompletedOrders.Visibility = Visibility.Hidden;

ShowHistoryOfOrders.Visibility = Visibility.Hidden;

}

private void AddUserButton(object sender, RoutedEventArgs e)

{

if(verifyField.Text != adminPW)

{

MessageBox.Show("Cannot verify admin.");

return;

}

if (adminButton.IsChecked == true) db.AddAdmin(addUser\_un.Text, addUser\_pw.Text);

else if (user1Button.IsChecked == true) db.AddUser1(addUser\_un.Text, addUser\_pw.Text);

else if (user2Button.IsChecked == true) db.AddUser2(addUser\_un.Text, addUser\_pw.Text);

else

{

MessageBox.Show("Please, choose a role.");

}

}

private void ShowAdmins(object sender, RoutedEventArgs e)

{

List<AdminClass> admins = new List<AdminClass>();

using (var connection = new SqlConnection(connString))

{

connection.Open();

var command = new SqlCommand("SELECT Username, Password FROM Admin", connection);

var reader = command.ExecuteReader();

while (reader.Read())

{

AdminClass admin = new AdminClass

{

Username = reader.GetString(0),

Password = reader.GetString(1),

};

admins.Add(admin);

}

}

dataGridRegistredUsers.ItemsSource = admins;

}

private void ShowCooks(object sender, RoutedEventArgs e)

{

List<CookClass> admins = new List<CookClass>();

using (var connection = new SqlConnection(connString))

{

connection.Open();

var command = new SqlCommand("SELECT Username, Password, Completed FROM User1", connection);

var reader = command.ExecuteReader();

while (reader.Read())

{

CookClass admin = new CookClass

{

Username = reader.GetString(0),

Password = reader.GetString(1),

Completed = reader.GetInt32(2),

};

admins.Add(admin);

}

}

dataGridRegistredUsers.ItemsSource = admins;

}

private void ShowCouriers(object sender, RoutedEventArgs e)

{

List<CourierClass> admins = new List<CourierClass>();

using (var connection = new SqlConnection(connString))

{

connection.Open();

var command = new SqlCommand("SELECT Usernamecour, Password, Completed FROM User2", connection);

var reader = command.ExecuteReader();

while (reader.Read())

{

CourierClass admin = new CourierClass

{

Username = reader.GetString(0),

Password = reader.GetString(1),

Completed = reader.GetInt32(2),

};

admins.Add(admin);

}

}

dataGridRegistredUsers.ItemsSource = admins;

}

private void ShowOverallUsersMenu(object sender, RoutedEventArgs e)

{

addUserGroup.Visibility = Visibility.Hidden;

addUser\_pw.Visibility = Visibility.Hidden;

addUser\_un.Visibility = Visibility.Hidden;

adminButton.Visibility = Visibility.Hidden;

user1Button.Visibility = Visibility.Hidden;

user2Button.Visibility = Visibility.Hidden;

verifyField.Visibility = Visibility.Hidden;

addUser.Visibility = Visibility.Hidden;

AdminDeletingRB.Visibility = Visibility.Hidden;

CookDeletingRB.Visibility = Visibility.Hidden;

CourierDeletingRB.Visibility = Visibility.Hidden;

UsernameOfDeletingUser.Visibility = Visibility.Hidden;

PasswordOfDeletingUser.Visibility = Visibility.Hidden;

PasswordVerifyForDelete.Visibility = Visibility.Hidden;

UserDeletion.Visibility = Visibility.Hidden;

delpwlbl.Visibility = Visibility.Hidden;

delunlbl.Visibility = Visibility.Hidden;

delvpwlbl.Visibility = Visibility.Hidden;

ShowAdminsInDataGrid.Visibility = Visibility.Visible;

ShowCooksInDataGrid.Visibility = Visibility.Visible;

dataGridRegistredUsers.Visibility = Visibility.Visible;

ShowCouriersInDataGrid.Visibility = Visibility.Visible;

MasterAdminPassword.Visibility = Visibility.Hidden;

ConfirmMasterAdmin.Visibility = Visibility.Hidden;

OrderGrid.Visibility = Visibility.Hidden;

NameBox.Visibility = Visibility.Hidden;

AddressBox.Visibility = Visibility.Hidden;

OrderConfirm.Visibility = Visibility.Hidden;

UpdateTable.Visibility = Visibility.Hidden;

OrderDataGrid.Visibility = Visibility.Hidden;

CourierIDTB.Visibility = Visibility.Hidden;

CookIDTB.Visibility = Visibility.Hidden;

nl.Visibility = Visibility.Hidden;

al.Visibility = Visibility.Hidden;

icl.Visibility = Visibility.Hidden;

icoul.Visibility = Visibility.Hidden;

StorageDataGrid.Visibility = Visibility.Hidden;

UpdateStorageGrid.Visibility = Visibility.Hidden;

DostavkaButton.Visibility = Visibility.Hidden;

MasterAdminPassword.Visibility = Visibility.Hidden;

ConfirmMasterAdmin.Visibility = Visibility.Hidden;

CompletedOrders.Visibility = Visibility.Hidden;

ShowHistoryOfOrders.Visibility = Visibility.Hidden;

}

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

{

string username = UsernameOfDeletingUser.Text;

string password = PasswordOfDeletingUser.Text;

if (masteradmin == true)

{

if (AdminDeletingRB.IsChecked == true) db.DeleteAdmin(username, password);

else if (CookDeletingRB.IsChecked == true) db.DeleteUser1(username, password);

else if (CourierDeletingRB.IsChecked == true) db.DeleteUser2(username, password);

else MessageBox.Show("Выберите тип пользователя.");

}

else MessageBox.Show("Необходимы права мастер-админа.");

}

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

{

addUserGroup.Visibility = Visibility.Hidden;

addUser\_pw.Visibility = Visibility.Hidden;

addUser\_un.Visibility = Visibility.Hidden;

adminButton.Visibility = Visibility.Hidden;

user1Button.Visibility = Visibility.Hidden;

user2Button.Visibility = Visibility.Hidden;

verifyField.Visibility = Visibility.Hidden;

addUser.Visibility = Visibility.Hidden;

AdminDeletingRB.Visibility = Visibility.Visible;

CookDeletingRB.Visibility = Visibility.Visible;

CourierDeletingRB.Visibility = Visibility.Visible;

UsernameOfDeletingUser.Visibility = Visibility.Visible;

PasswordOfDeletingUser.Visibility = Visibility.Visible;

PasswordVerifyForDelete.Visibility = Visibility.Visible;

UserDeletion.Visibility = Visibility.Visible;

delpwlbl.Visibility = Visibility.Visible;

delunlbl.Visibility = Visibility.Visible;

delvpwlbl.Visibility = Visibility.Visible;

ShowAdminsInDataGrid.Visibility = Visibility.Hidden;

ShowCooksInDataGrid.Visibility = Visibility.Hidden;

dataGridRegistredUsers.Visibility = Visibility.Hidden;

ShowCouriersInDataGrid.Visibility = Visibility.Hidden;

MasterAdminPassword.Visibility = Visibility.Hidden;

ConfirmMasterAdmin.Visibility = Visibility.Hidden;

OrderGrid.Visibility = Visibility.Hidden;

NameBox.Visibility = Visibility.Hidden;

AddressBox.Visibility = Visibility.Hidden;

OrderConfirm.Visibility = Visibility.Hidden;

UpdateTable.Visibility = Visibility.Hidden;

OrderDataGrid.Visibility = Visibility.Hidden;

CourierIDTB.Visibility = Visibility.Hidden;

CookIDTB.Visibility = Visibility.Hidden;

nl.Visibility = Visibility.Hidden;

al.Visibility = Visibility.Hidden;

icl.Visibility = Visibility.Hidden;

icoul.Visibility = Visibility.Hidden;

StorageDataGrid.Visibility = Visibility.Hidden;

UpdateStorageGrid.Visibility = Visibility.Hidden;

DostavkaButton.Visibility = Visibility.Hidden;

MasterAdminPassword.Visibility = Visibility.Hidden;

ConfirmMasterAdmin.Visibility = Visibility.Hidden;

CompletedOrders.Visibility = Visibility.Hidden;

ShowHistoryOfOrders.Visibility = Visibility.Hidden;

}

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

{

if (MasterAdminPassword.Text == masterdaminpw)

{

masteradmin = true;

MessageBox.Show("Мастер-админ подтвержден. Все функции разблокированы.");

MasterAdminActive.Content = "Мастер-админ активен";

}

}

private void MasterAdminMenu(object sender, RoutedEventArgs e)

{

addUserGroup.Visibility = Visibility.Hidden;

addUser\_pw.Visibility = Visibility.Hidden;

addUser\_un.Visibility = Visibility.Hidden;

adminButton.Visibility = Visibility.Hidden;

user1Button.Visibility = Visibility.Hidden;

user2Button.Visibility = Visibility.Hidden;

verifyField.Visibility = Visibility.Hidden;

addUser.Visibility = Visibility.Hidden;

AdminDeletingRB.Visibility = Visibility.Hidden;

CookDeletingRB.Visibility = Visibility.Hidden;

CourierDeletingRB.Visibility = Visibility.Hidden;

UsernameOfDeletingUser.Visibility = Visibility.Hidden;

PasswordOfDeletingUser.Visibility = Visibility.Hidden;

PasswordVerifyForDelete.Visibility = Visibility.Hidden;

UserDeletion.Visibility = Visibility.Hidden;

delpwlbl.Visibility = Visibility.Hidden;

delunlbl.Visibility = Visibility.Hidden;

delvpwlbl.Visibility = Visibility.Hidden;

ShowAdminsInDataGrid.Visibility = Visibility.Hidden;

ShowCooksInDataGrid.Visibility = Visibility.Hidden;

dataGridRegistredUsers.Visibility = Visibility.Hidden;

ShowCouriersInDataGrid.Visibility = Visibility.Hidden;

MasterAdminPassword.Visibility = Visibility.Visible;

ConfirmMasterAdmin.Visibility = Visibility.Visible;

StorageDataGrid.Visibility = Visibility.Hidden;

UpdateStorageGrid.Visibility = Visibility.Hidden;

DostavkaButton.Visibility = Visibility.Hidden;

OrderGrid.Visibility = Visibility.Hidden;

NameBox.Visibility = Visibility.Hidden;

AddressBox.Visibility = Visibility.Hidden;

OrderConfirm.Visibility = Visibility.Hidden;

UpdateTable.Visibility = Visibility.Hidden;

OrderDataGrid.Visibility = Visibility.Hidden;

CourierIDTB.Visibility = Visibility.Hidden;

CookIDTB.Visibility = Visibility.Hidden;

nl.Visibility = Visibility.Hidden;

al.Visibility = Visibility.Hidden;

icl.Visibility = Visibility.Hidden;

icoul.Visibility = Visibility.Hidden;

CompletedOrders.Visibility = Visibility.Hidden;

ShowHistoryOfOrders.Visibility = Visibility.Hidden;

}

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

{

quantityList[0]++;

ma.NeededBuns += 2;

ma.NeededTomatoes += 1;

ma.NeededCheese += 1;

ma.NeededSalad += 1;

ma.NeededPatties += 1;

}

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

{

quantityList[1]++;

ma.NeededBuns += 2;

ma.NeededTomatoes += 1;

ma.NeededCheese += 1;

ma.NeededSalad += 1;

ma.NeededPatties += 1;

ma.NeededOnions += 2;

}

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

{

quantityList[2]++;

ma.NeededBuns += 2;

ma.NeededTomatoes += 2;

ma.NeededCheese += 2;

ma.NeededSalad += 1;

ma.NeededPatties += 2;

}

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

{

quantityList[3]++;

ma.NeededBuns += 2;

ma.NeededTomatoes += 2;

ma.NeededCheese += 2;

ma.NeededSalad += 2;

ma.NeededPatties += 3;

}

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

{

quantityList[4]++;

ma.NeededBuns += 3;

ma.NeededTomatoes += 3;

ma.NeededCheese += 4;

ma.NeededSalad += 2;

ma.NeededPatties += 3;

ma.NeededFries += 1;

}

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

{

quantityList[5]++;

}

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

{

quantityList[6]++;

ma.NeededFries += 3;

}

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

{

quantityList[7]++;

}

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

{

quantityList[8]++;

}

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

{

quantityList[9]++;

ma.NeededSalad += 4;

ma.NeededOnions += 1;

ma.NeededTomatoes += 3;

}

private void ShowOrderMenu(object sender, RoutedEventArgs e)

{

addUserGroup.Visibility = Visibility.Hidden;

addUser\_pw.Visibility = Visibility.Hidden;

addUser\_un.Visibility = Visibility.Hidden;

adminButton.Visibility = Visibility.Hidden;

user1Button.Visibility = Visibility.Hidden;

user2Button.Visibility = Visibility.Hidden;

verifyField.Visibility = Visibility.Hidden;

addUser.Visibility = Visibility.Hidden;

AdminDeletingRB.Visibility = Visibility.Hidden;

CookDeletingRB.Visibility = Visibility.Hidden;

CourierDeletingRB.Visibility = Visibility.Hidden;

UsernameOfDeletingUser.Visibility = Visibility.Hidden;

PasswordOfDeletingUser.Visibility = Visibility.Hidden;

PasswordVerifyForDelete.Visibility = Visibility.Hidden;

UserDeletion.Visibility = Visibility.Hidden;

delpwlbl.Visibility = Visibility.Hidden;

delunlbl.Visibility = Visibility.Hidden;

delvpwlbl.Visibility = Visibility.Hidden;

ShowAdminsInDataGrid.Visibility = Visibility.Hidden;

ShowCooksInDataGrid.Visibility = Visibility.Hidden;

dataGridRegistredUsers.Visibility = Visibility.Hidden;

ShowCouriersInDataGrid.Visibility = Visibility.Hidden;

MasterAdminPassword.Visibility = Visibility.Hidden;

ConfirmMasterAdmin.Visibility = Visibility.Hidden;

OrderGrid.Visibility = Visibility.Visible;

NameBox.Visibility = Visibility.Visible;

AddressBox.Visibility = Visibility.Visible;

OrderConfirm.Visibility = Visibility.Visible;

UpdateTable.Visibility = Visibility.Visible;

OrderDataGrid.Visibility = Visibility.Visible;

CourierIDTB.Visibility = Visibility.Visible;

CookIDTB.Visibility = Visibility.Visible;

nl.Visibility = Visibility.Visible;

al.Visibility = Visibility.Visible;

icl.Visibility = Visibility.Visible;

icoul.Visibility = Visibility.Visible;

CompletedOrders.Visibility = Visibility.Hidden;

ShowHistoryOfOrders.Visibility = Visibility.Hidden;

StorageDataGrid.Visibility = Visibility.Hidden;

UpdateStorageGrid.Visibility = Visibility.Hidden;

DostavkaButton.Visibility = Visibility.Hidden;

MasterAdminPassword.Visibility = Visibility.Hidden;

ConfirmMasterAdmin.Visibility = Visibility.Hidden;

}

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

{

List<StorageClass> st = new List<StorageClass>();

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

{

orderString += (i.ToString() + quantityList[i].ToString());

orderString += "|";

}

StorageClass storage = new StorageClass();

using (var connection = new SqlConnection(connString))

{

connection.Open();

var command = new SqlCommand("SELECT Buns, Tomatoes,Cheese, Salad, Patties, Onions, Fries FROM Storage", connection);

var reader = command.ExecuteReader();

while (reader.Read())

{

storage = new StorageClass

{

Buns = reader.GetInt32(0),

Tomatoes = reader.GetInt32(1),

Cheese = reader.GetInt32(2),

Salad = reader.GetInt32(3),

Patties = reader.GetInt32(4),

Onions = reader.GetInt32(5),

Fries = reader.GetInt32(6),

};

}

reader.Close();

storage -= ma;

if(storage.ContainsNegativeValues())

{

MessageBox.Show("Недостаточно материалов на складе.");

storage += ma;

}

using (command = new SqlCommand("UPDATE Storage SET Buns = @Buns, Tomatoes = @Tomatoes, Cheese = @Cheese, Salad = @Salad, Patties = @Patties, Onions = @Onions, Fries = @Fries", connection))

{

command.Parameters.AddWithValue("@Buns", storage.Buns);

command.Parameters.AddWithValue("@Tomatoes", storage.Tomatoes);

command.Parameters.AddWithValue("@Cheese", storage.Cheese);

command.Parameters.AddWithValue("@Salad", storage.Salad);

command.Parameters.AddWithValue("@Patties", storage.Patties);

command.Parameters.AddWithValue("@Onions", storage.Onions);

command.Parameters.AddWithValue("@Fries", storage.Fries);

command.ExecuteNonQuery();

}

st.Add(storage);

}

OrderDataGrid.ItemsSource = st;

db.AddOrder(NameBox.Text, AddressBox.Text, orderString,CookIDTB.Text, CourierIDTB.Text);

}

public void UpdateTableClick(object sender, RoutedEventArgs e)

{

List<OrderClass> orders = new List<OrderClass>();

using (var connection = new SqlConnection(connString))

{

connection.Open();

string command = @"

SELECT Orders.OrderID,

Orders.OrderCode,

Orders.OrderName,

Orders.Address,

Orders.IsCooked,

Orders.IsDelivered,

User1.username,

User2.Usernamecour

FROM Orders

INNER JOIN User1 ON Orders.CookID = User1.ID

INNER JOIN User2 ON Orders.CourierID = User2.ID

WHERE Orders.IsDelivered = 0;";

using (SqlConnection conn = new SqlConnection(connString))

{

SqlDataAdapter adapter = new SqlDataAdapter(command, conn);

DataTable dataTable = new DataTable();

adapter.Fill(dataTable);

OrderDataGrid.ItemsSource = dataTable.DefaultView;

}

}

}

private void ShowStorage(object sender, RoutedEventArgs e)

{

using (SqlConnection sql\_conn = new SqlConnection("Server=ABUBACHIRTRAHOV\\BDISUBIDE; Initial Catalog=synthfoods; Integrated Security = True"))

{

sql\_conn.Open();

var sql\_cmd = sql\_conn.CreateCommand();

sql\_cmd.CommandText = "SELECT \* FROM Storage";

using (var sqlite\_datareader = sql\_cmd.ExecuteReader())

{

var dataTable = new DataTable();

dataTable.Load(sqlite\_datareader);

StorageDataGrid.ItemsSource = dataTable.DefaultView;

}

sql\_conn.Close();

}

}

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

{

addUserGroup.Visibility = Visibility.Hidden;

addUser\_pw.Visibility = Visibility.Hidden;

addUser\_un.Visibility = Visibility.Hidden;

adminButton.Visibility = Visibility.Hidden;

user1Button.Visibility = Visibility.Hidden;

user2Button.Visibility = Visibility.Hidden;

verifyField.Visibility = Visibility.Hidden;

addUser.Visibility = Visibility.Hidden;

AdminDeletingRB.Visibility = Visibility.Hidden;

CookDeletingRB.Visibility = Visibility.Hidden;

CourierDeletingRB.Visibility = Visibility.Hidden;

UsernameOfDeletingUser.Visibility = Visibility.Hidden;

PasswordOfDeletingUser.Visibility = Visibility.Hidden;

PasswordVerifyForDelete.Visibility = Visibility.Hidden;

UserDeletion.Visibility = Visibility.Hidden;

delpwlbl.Visibility = Visibility.Hidden;

delunlbl.Visibility = Visibility.Hidden;

delvpwlbl.Visibility = Visibility.Hidden;

ShowAdminsInDataGrid.Visibility = Visibility.Hidden;

ShowCooksInDataGrid.Visibility = Visibility.Hidden;

dataGridRegistredUsers.Visibility = Visibility.Hidden;

ShowCouriersInDataGrid.Visibility = Visibility.Hidden;

MasterAdminPassword.Visibility = Visibility.Hidden;

ConfirmMasterAdmin.Visibility = Visibility.Hidden;

OrderGrid.Visibility = Visibility.Hidden;

NameBox.Visibility = Visibility.Hidden;

AddressBox.Visibility = Visibility.Hidden;

OrderConfirm.Visibility = Visibility.Hidden;

UpdateTable.Visibility = Visibility.Hidden;

OrderDataGrid.Visibility = Visibility.Hidden;

CourierIDTB.Visibility = Visibility.Hidden;

CookIDTB.Visibility = Visibility.Hidden;

nl.Visibility = Visibility.Hidden;

al.Visibility = Visibility.Hidden;

icl.Visibility = Visibility.Hidden;

icoul.Visibility = Visibility.Hidden;

StorageDataGrid.Visibility = Visibility.Visible;

UpdateStorageGrid.Visibility = Visibility.Visible;

DostavkaButton.Visibility = Visibility.Visible;

CompletedOrders.Visibility = Visibility.Hidden;

ShowHistoryOfOrders.Visibility = Visibility.Hidden;

MasterAdminPassword.Visibility = Visibility.Hidden;

ConfirmMasterAdmin.Visibility = Visibility.Hidden;

}

private void ClosedWinddow(object sender, EventArgs e)

{

loginwindow.Show();

loginwindow.ClearForAdminExit();

this.Hide();

}

private void DeleteOrderInDataGrid(object sender, RoutedEventArgs e)

{

if (OrderDataGrid.SelectedItem != null)

{

OrderClass selectedRow = (OrderClass)OrderDataGrid.SelectedItem;

int orderId = selectedRow.Id;

DeleteOrder(orderId);

UpdateTableClick(sender, e);

}

else

{

MessageBox.Show("Пожалуйста, выберите строку для удаления.");

}

}

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

{

if (OrderDataGrid.SelectedItem != null)

{

OrderClass selectedOrder = (OrderClass)OrderDataGrid.SelectedItem;

int orderId = selectedOrder.Id;

DeleteOrder(orderId);

}

else

{

MessageBox.Show("Пожалуйста, выберите строку для удаления.");

}

}

private void DeleteOrder(int orderId)

{

int attempts = 5; // Количество повторных попыток

bool success = false;

for (int i = 0; i < attempts; i++)

{

try

{

using (var connection = new SqlConnection(connString))

{

connection.Open();

string query = "DELETE FROM Orders WHERE OrderID = @Id";

using (var command = new SqlCommand(query, connection))

{

command.Parameters.AddWithValue("@Id", orderId);

command.ExecuteNonQuery();

}

}

success = true;

break;

}

catch (SQLiteException ex) when (ex.ResultCode == SQLiteErrorCode.Locked)

{

Thread.Sleep(1000);

}

}

if (success)

{

MessageBox.Show("Строка успешно удалена.");

}

else

{

MessageBox.Show("Не удалось удалить строку из-за блокировки базы данных.");

}

}

private void Button\_Click\_3(object sender, RoutedEventArgs e)

{

List < StorageClass > st = new List<StorageClass>();

StorageClass storage = new StorageClass();

using (var connection = new SqlConnection(connString))

{

connection.Open();

var command = new SqlCommand("SELECT Buns, Tomatoes,Cheese, Salad, Patties, Onions, Fries FROM Storage", connection);

var reader = command.ExecuteReader();

while (reader.Read())

{

storage = new StorageClass

{

Buns = reader.GetInt32(0),

Tomatoes = reader.GetInt32(1),

Cheese = reader.GetInt32(2),

Salad = reader.GetInt32(3),

Patties = reader.GetInt32(4),

Onions = reader.GetInt32(5),

Fries = reader.GetInt32(6),

};

}

reader.Close();

storage += 20;

using (command = new SqlCommand("UPDATE Storage SET Buns = @Buns, Tomatoes = @Tomatoes, Cheese = @Cheese, Salad = @Salad, Patties = @Patties, Onions = @Onions, Fries = @Fries", connection))

{

command.Parameters.AddWithValue("@Buns", storage.Buns);

command.Parameters.AddWithValue("@Tomatoes", storage.Tomatoes);

command.Parameters.AddWithValue("@Cheese", storage.Cheese);

command.Parameters.AddWithValue("@Salad", storage.Salad);

command.Parameters.AddWithValue("@Patties", storage.Patties);

command.Parameters.AddWithValue("@Onions", storage.Onions);

command.Parameters.AddWithValue("@Fries", storage.Fries);

command.ExecuteNonQuery();

}

st.Add(storage);

}

OrderDataGrid.ItemsSource = st;

ShowStorage(sender, e);

}

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

{

List<OrderClass> orders = new List<OrderClass>();

using (var connection = new SqlConnection(connString))

{

connection.Open();

string command = @"

SELECT Orders.OrderID,

Orders.OrderCode,

Orders.OrderName,

Orders.Address,

User1.username,

User2.Usernamecour

FROM Orders

INNER JOIN User1 ON Orders.CookID = User1.ID

INNER JOIN User2 ON Orders.CourierID = User2.ID

WHERE Orders.IsDelivered = 1;";

using (SqlConnection conn = new SqlConnection(connString))

{

SqlDataAdapter adapter = new SqlDataAdapter(command, conn);

DataTable dataTable = new DataTable();

adapter.Fill(dataTable);

CompletedOrders.ItemsSource = dataTable.DefaultView;

}

}

}

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

{

addUserGroup.Visibility = Visibility.Hidden;

addUser\_pw.Visibility = Visibility.Hidden;

addUser\_un.Visibility = Visibility.Hidden;

adminButton.Visibility = Visibility.Hidden;

user1Button.Visibility = Visibility.Hidden;

user2Button.Visibility = Visibility.Hidden;

verifyField.Visibility = Visibility.Hidden;

addUser.Visibility = Visibility.Hidden;

AdminDeletingRB.Visibility = Visibility.Hidden;

CookDeletingRB.Visibility = Visibility.Hidden;

CourierDeletingRB.Visibility = Visibility.Hidden;

UsernameOfDeletingUser.Visibility = Visibility.Hidden;

PasswordOfDeletingUser.Visibility = Visibility.Hidden;

PasswordVerifyForDelete.Visibility = Visibility.Hidden;

UserDeletion.Visibility = Visibility.Hidden;

delpwlbl.Visibility = Visibility.Hidden;

delunlbl.Visibility = Visibility.Hidden;

delvpwlbl.Visibility = Visibility.Hidden;

ShowAdminsInDataGrid.Visibility = Visibility.Hidden;

ShowCooksInDataGrid.Visibility = Visibility.Hidden;

dataGridRegistredUsers.Visibility = Visibility.Hidden;

ShowCouriersInDataGrid.Visibility = Visibility.Hidden;

MasterAdminPassword.Visibility = Visibility.Hidden;

ConfirmMasterAdmin.Visibility = Visibility.Hidden;

OrderGrid.Visibility = Visibility.Hidden;

NameBox.Visibility = Visibility.Hidden;

AddressBox.Visibility = Visibility.Hidden;

OrderConfirm.Visibility = Visibility.Hidden;

UpdateTable.Visibility = Visibility.Hidden;

OrderDataGrid.Visibility = Visibility.Hidden;

CourierIDTB.Visibility = Visibility.Hidden;

CookIDTB.Visibility = Visibility.Hidden;

nl.Visibility = Visibility.Hidden;

al.Visibility = Visibility.Hidden;

icl.Visibility = Visibility.Hidden;

icoul.Visibility = Visibility.Hidden;

StorageDataGrid.Visibility = Visibility.Hidden;

UpdateStorageGrid.Visibility = Visibility.Hidden;

DostavkaButton.Visibility = Visibility.Hidden;

CompletedOrders.Visibility = Visibility.Visible;

ShowHistoryOfOrders.Visibility = Visibility.Visible;

MasterAdminPassword.Visibility = Visibility.Hidden;

ConfirmMasterAdmin.Visibility = Visibility.Hidden;

}

}

}

Код формы повара:

namespace WpfApp6

{

public partial class OrderViewer : Window

{

private string[] dishName = new string[] { "Cheeseburger", "Diablo", "Double Cheeseburger", "Cheeseburger Tower", "Ultra Burger", "Coolah", "Fries", "Sundae Icecream", "Cappucino", "Caesar Salad" };

private List<string> orderCodes;

private int currentIndex;

MainWindow prev;

string connectionString;

string uname;

public OrderViewer()

{

InitializeComponent();

LoadOrderCodes();

DisplayCurrentOrderCode();

}

SqlConnection sql\_conn;

public OrderViewer(MainWindow p, string username)

{

uname = username;

InitializeComponent();

LoadOrderCodes();

DisplayCurrentOrderCode();

prev = p;

connectionString = "Server=ABUBACHIRTRAHOV\\BDISUBIDE; Initial Catalog=synthfoods; Integrated Security = True";

sql\_conn = new SqlConnection(connectionString);

}

private void LoadOrderCodes()

{

orderCodes = new List<string>();

string connectionString = "Server=ABUBACHIRTRAHOV\\BDISUBIDE; Initial Catalog=synthfoods; Integrated Security=True";

string query = @"

use synthfoods

DECLARE @CookID INT;

SET @CookID = (SELECT User1.ID FROM User1 WHERE Username = @Username);

SELECT OrderCode

FROM Orders

WHERE Orders.CookID = @CookID AND Orders.IsCooked = 0;

";

string username = uname;

using (SqlConnection connection = new SqlConnection(connectionString))

{

SqlCommand command = new SqlCommand(query, connection);

command.Parameters.AddWithValue("@Username", username);

connection.Open();

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

orderCodes.Add(reader.GetString(0));

}

reader.Close();

connection.Close();

}

currentIndex = 0;

}

private void DisplayCurrentOrderCode()

{

if (currentIndex < orderCodes.Count)

{

string displayText = DecodeOrder(orderCodes[currentIndex]);

OrderCodeTextBlock.Text = displayText;

}

else

{

NextButton.Visibility = Visibility.Hidden;

OrderDataGrid.Visibility = Visibility.Hidden;

OrderCodeTextBlock.Text = "Заказов пока нет.";

}

}

private string DecodeOrder(string orderCode)

{

string result = "";

string[] orderItems = orderCode.Split('|', System.StringSplitOptions.RemoveEmptyEntries);

foreach (string item in orderItems)

{

if (item.Length < 2) continue;

int dishIndex = int.Parse(item[0].ToString());

int quantity = int.Parse(item[1].ToString());

if (quantity > 0 && dishIndex < dishName.Length)

{

result += $"{dishName[dishIndex]} x{quantity}\n";

}

}

return result;

}

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

{

if (currentIndex < orderCodes.Count)

{

using (var connection = new SqlConnection(connectionString))

{

List<OrderClass> orders = new List<OrderClass>();

connection.Open();

string command = @$"

use synthfoods

DECLARE @CookID INT;

SET @CookID = (SELECT User1.ID FROM User1 WHERE Username = '{uname}');

DECLARE @CompletedOrderID INT

SET @CompletedOrderID = (

SELECT TOP 1 Orders.OrderID

FROM Orders

WHERE IsCooked != 1 AND Orders.CookID = @CookID);

UPDATE Orders

Set IsCooked = 1

WHERE @CompletedOrderID = OrderID AND IsCooked != 1;

UPDATe User1

SET Completed = Completed + 1

WHERE @CookID = User1.ID;

";

using (SqlCommand cmd = new SqlCommand(command, connection))

{

using (SqlConnection conn = new SqlConnection(connectionString))

{

SqlDataAdapter adapter = new SqlDataAdapter(command, conn);

DataTable dataTable = new DataTable();

adapter.Fill(dataTable);

OrderDataGrid.ItemsSource = dataTable.DefaultView;

}

}

}

using (var connection = new SqlConnection(connectionString))

{

List<OrderClass> orders = new List<OrderClass>();

connection.Open();

string command = @$"

use synthfoods

DECLARE @CookID INT;

SET @CookID = (

SELECT User1.ID

FROM User1

WHERE username = '{uname}'

);

DECLARE @CompletedOrderID INT;

SET @CompletedOrderID = (

SELECT TOP 1 Orders.OrderID

FROM Orders

WHERE IsCooked != 1 AND Orders.CookID = @CookID );

SELECT OrderID, OrderCode, OrderName, Address, CookID, IsCooked

FROM Orders

WHERE Orders.CookID = @CookID AND IsCooked = 0;";

using (SqlCommand cmd = new SqlCommand(command, connection))

{

using (SqlConnection conn = new SqlConnection(connectionString))

{

SqlDataAdapter adapter = new SqlDataAdapter(command, conn);

DataTable dataTable = new DataTable();

adapter.Fill(dataTable);

OrderDataGrid.ItemsSource = dataTable.DefaultView;

}

}

}

currentIndex++;

DisplayCurrentOrderCode();

}

else

{

using (var connection = new SqlConnection(connectionString))

{

List<OrderClass> orders = new List<OrderClass>();

connection.Open();

string command = @$"

use synthfoods

DECLARE @CookID INT;

SET @CookID = (

SELECT User1.ID

FROM User1

WHERE username = '{uname}'

);

DECLARE @CompletedOrderID INT;

SET @CompletedOrderID = (

SELECT TOP 1 Orders.OrderID

FROM Orders

WHERE IsCooked != 1 AND Orders.CookID = @CookID );

SELECT OrderID, OrderCode, OrderName, Address, CookID, IsCooked

FROM Orders

WHERE Orders.CookID = @CookID AND IsCooked = 0;";

using (SqlCommand cmd = new SqlCommand(command, connection))

{

using (SqlConnection conn = new SqlConnection(connectionString))

{

SqlDataAdapter adapter = new SqlDataAdapter(command, conn);

DataTable dataTable = new DataTable();

adapter.Fill(dataTable);

OrderDataGrid.ItemsSource = dataTable.DefaultView;

}

}

}

OrderCodeTextBlock.Text = "Все заказы сделаны!";

}

}

private void SHowPrev(object sender, System.EventArgs e)

{

prev.Show();

prev.ClearForAdminExit();

}

private void Loaded(object sender, RoutedEventArgs e)

{

using (var connection = new SqlConnection(connectionString))

{

List<OrderClass> orders = new List<OrderClass>();

connection.Open();

string command = @$"

use synthfoods

DECLARE @CookID INT;

SET @CookID = (SELECT User1.ID FROM User1 WHERE Username = '{uname}');

SELECT OrderID, OrderCode, OrderName, Address, CookID, IsCooked

FROM Orders

WHERE Orders.CookID = @CookID AND Orders.IsCooked = 0";

using (SqlCommand cmd = new SqlCommand(command, connection))

{

using (SqlConnection conn = new SqlConnection(connectionString))

{

SqlDataAdapter adapter = new SqlDataAdapter(command, conn);

DataTable dataTable = new DataTable();

adapter.Fill(dataTable);

OrderDataGrid.ItemsSource = dataTable.DefaultView;

}

}

}

}

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

{

}

}

}

Код формы для курьера:

namespace WpfApp6

{

public partial class MenuWindow : Window

{

private string[] dishName = new string[] { "Cheeseburger", "Diablo", "Double Cheeseburger", "Cheeseburger Tower", "Ultra Burger", "Coolah", "Fries", "Sundae Icecream", "Cappucino", "Caesar Salad" };

private List<string> orderCodes;

private int currentIndex;

MainWindow prev;

string connectionString;

string uname;

public MenuWindow()

{

InitializeComponent();

LoadOrderCodes();

DisplayCurrentOrderCode();

}

SqlConnection sql\_conn;

public MenuWindow(MainWindow p, string username)

{

uname = username;

InitializeComponent();

LoadOrderCodes();

DisplayCurrentOrderCode();

prev = p;

connectionString = "Server=ABUBACHIRTRAHOV\\BDISUBIDE; Initial Catalog=synthfoods; Integrated Security = True";

sql\_conn = new SqlConnection(connectionString);

}

private void LoadOrderCodes()

{

orderCodes = new List<string>();

string connectionString = "Server=ABUBACHIRTRAHOV\\BDISUBIDE; Initial Catalog=synthfoods; Integrated Security=True";

string query = @$"

use synthfoods

DECLARE @CourierID INT;

SET @CourierID = (SELECT User2.ID FROM User2 WHERE Usernamecour = '{uname}');

SELECT OrderCode

FROM Orders

WHERE Orders.CourierID = @CourierID AND Orders.IsDelivered = 0 AND Orders.IsCooked = 1;

";

using (SqlConnection connection = new SqlConnection(connectionString))

{

SqlCommand command = new SqlCommand(query, connection);

connection.Open();

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

orderCodes.Add(reader.GetString(0));

}

reader.Close();

connection.Close();

}

currentIndex = 0;

}

private void DisplayCurrentOrderCode()

{

if (currentIndex < orderCodes.Count)

{

string displayText = DecodeOrder(orderCodes[currentIndex]);

OrderCodeTextBlock.Text = displayText;

}

else

{

OrderCodeTextBlock.Text = "Заказов пока нет.";

}

}

private string DecodeOrder(string orderCode)

{

string result = "";

string[] orderItems = orderCode.Split('|', System.StringSplitOptions.RemoveEmptyEntries);

foreach (string item in orderItems)

{

if (item.Length < 2) continue;

int dishIndex = int.Parse(item[0].ToString());

int quantity = int.Parse(item[1].ToString());

if (quantity > 0 && dishIndex < dishName.Length)

{

result += $"{dishName[dishIndex]} x{quantity}\n";

}

}

return result;

}

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

{

if (currentIndex < orderCodes.Count)

{

using (var connection = new SqlConnection(connectionString))

{

List<OrderClass> orders = new List<OrderClass>();

connection.Open();

string command = @$"

use synthfoods

DECLARE @CourierID INT;

SET @CourierID = (SELECT User2.ID FROM User2 WHERE Usernamecour = '{uname}');

DECLARE @CompletedOrderID INT

SET @CompletedOrderID = (

SELECT TOP 1 Orders.OrderID

FROM Orders

WHERE IsCooked = 1 AND Orders.CourierID = @CourierID AND Orders.IsDelivered = 0);

UPDATE Orders

SET IsDelivered = 1

WHERE @CompletedOrderID = OrderID AND IsDelivered = 0 AND IsCooked = 1;

UPDATe User2

SET Completed = Completed + 1

WHERE @CourierID = User2.ID;

";

using (SqlCommand cmd = new SqlCommand(command, connection))

{

using (SqlConnection conn = new SqlConnection(connectionString))

{

SqlDataAdapter adapter = new SqlDataAdapter(command, conn);

DataTable dataTable = new DataTable();

adapter.Fill(dataTable);

OrderDataGrid.ItemsSource = dataTable.DefaultView;

}

}

}

using (var connection = new SqlConnection(connectionString))

{

List<OrderClass> orders = new List<OrderClass>();

connection.Open();

string command = @$"

use synthfoods

DECLARE @CourierID INT;

SET @CourierID = (

SELECT User2.ID

FROM User2

WHERE usernamecour = '{uname}'

);

DECLARE @CompletedOrderID INT;

SET @CompletedOrderID = (

SELECT TOP 1 Orders.OrderID

FROM Orders

WHERE IsCooked = 1 AND Orders.CourierID = @CourierID AND Orders.IsDelivered = 0);

SELECT OrderID, OrderCode, OrderName, Address, CourierID, IsCooked

FROM Orders

WHERE Orders.CourierID = @CourierID AND IsDelivered = 0;";

using (SqlCommand cmd = new SqlCommand(command, connection))

{

using (SqlConnection conn = new SqlConnection(connectionString))

{

SqlDataAdapter adapter = new SqlDataAdapter(command, conn);

DataTable dataTable = new DataTable();

adapter.Fill(dataTable);

OrderDataGrid.ItemsSource = dataTable.DefaultView;

}

}

}

currentIndex++;

DisplayCurrentOrderCode();

}

else

{

using (var connection = new SqlConnection(connectionString))

{

List<OrderClass> orders = new List<OrderClass>();

connection.Open();

string command = @$"

use synthfoods

DECLARE @CourierID INT;

SET @CourierID = (

SELECT User2.ID

FROM User2

WHERE usernamecour = '{uname}'

);

DECLARE @CompletedOrderID INT;

SET @CompletedOrderID = (

SELECT TOP 1 Orders.OrderID

FROM Orders

WHERE IsCooked = 1 AND Orders.CourierID = @CourierID AND Orders.IsDelivered = 0);

SELECT OrderID, OrderCode, OrderName, Address, CourierID, IsCooked

FROM Orders

WHERE Orders.CookID = @CourierID AND IsDelivered = 0;";

using (SqlCommand cmd = new SqlCommand(command, connection))

{

using (SqlConnection conn = new SqlConnection(connectionString))

{

SqlDataAdapter adapter = new SqlDataAdapter(command, conn);

DataTable dataTable = new DataTable();

adapter.Fill(dataTable);

OrderDataGrid.ItemsSource = dataTable.DefaultView;

}

}

}

OrderCodeTextBlock.Text = "Все заказы сделаны!";

}

}

private void SHowPrev(object sender, System.EventArgs e)

{

prev.Show();

prev.ClearForAdminExit();

}

private void Loaded(object sender, RoutedEventArgs e)

{

using (var connection = new SqlConnection(connectionString))

{

List<OrderClass> orders = new List<OrderClass>();

connection.Open();

string command = @$"

use synthfoods

DECLARE @CourierID INT;

SET @CourierID = (SELECT User2.ID FROM User2 WHERE Usernamecour = '{uname}');

SELECT OrderID, OrderCode, OrderName, Address, CookID, IsDelivered

FROM Orders

WHERE Orders.CourierID = @CourierID AND Orders.IsDelivered = 0";

using (SqlCommand cmd = new SqlCommand(command, connection))

{

using (SqlConnection conn = new SqlConnection(connectionString))

{

SqlDataAdapter adapter = new SqlDataAdapter(command, conn);

DataTable dataTable = new DataTable();

adapter.Fill(dataTable);

OrderDataGrid.ItemsSource = dataTable.DefaultView;

}

}

}

}

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

{

}

}

# Приложение Б

**(справочное)  
Результаты работы программы**

Изображение выглядит как текст, снимок экрана, дизайн

Автоматически созданное описание

Рисунок Б.1 – Окно авторизации с заполненными полями

Изображение выглядит как снимок экрана, фиолетовый, текст, Фиолетовый

Автоматически созданное описание

Рисунок Б.2 – Главное окно

Изображение выглядит как текст, снимок экрана, программное обеспечение, фиолетовый

Автоматически созданное описание

Рисунок Б.3 – Окно заказов

Изображение выглядит как снимок экрана, текст, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок Б.4 – Окно сотрудников

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок Б.5 – Окно готовых заказов

Изображение выглядит как текст, снимок экрана, фиолетовый, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок Б.6 – Окно добавления работников

Изображение выглядит как текст, снимок экрана, фиолетовый, Фиолетовый

Автоматически созданное описание

Рисунок Б.7 – Окно удаления работников

Изображение выглядит как текст, снимок экрана, фиолетовый, Фиолетовый

Автоматически созданное описание

Рисунок Б.8 – Окно с мастерскими

Изображение выглядит как текст, снимок экрана, фиолетовый, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок Б.9 – Окно с содержимым склада

Изображение выглядит как текст, снимок экрана, Шрифт, дизайн

Автоматически созданное описание

Рисунок Б.10 – Окно с заказом для повара и курьера

*КП Т.197020.401 ГЧ*

КБП

У

Т. Контр.

Разраб. Разраб.

Тихомиров Е.В.

Провер. Провер.

Кривошеина А.В.

Реценз.

Утверд.

Н. Контр.

№ докум.

Подпись

Дата

Изм.

Лист

Масса

Лит.

Масштаб

Лист 1

Листов 4

Инв.№подл.

Подп. и дата

Взам.инв.№

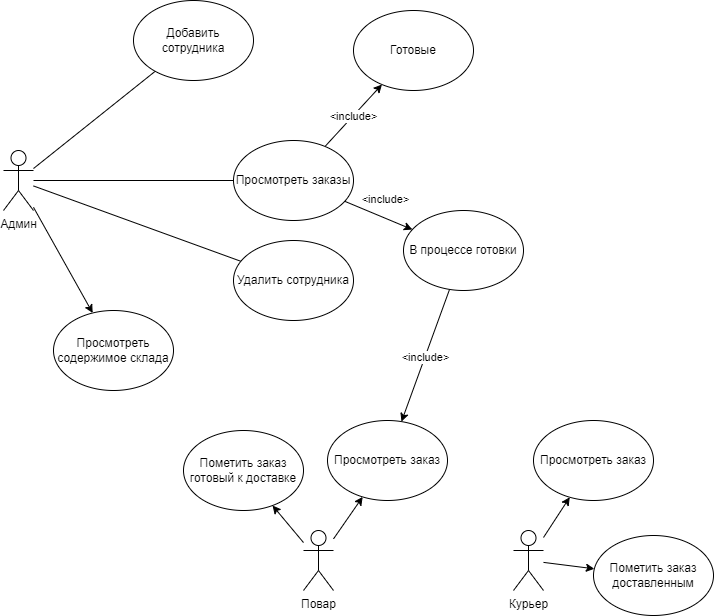
Инв.№дубл.

Подп. и дата

КП Т.197020.401 ГЧ Т.ХХХХХХ.401 ГЧ

*Разработка автоматизированного рабочего места сотрудника онлайн-ресторана*

*Диаграмма вариантов использования*



*Диаграмма вариантов использования*

Изображение выглядит как текст, снимок экрана, Шрифт, диаграмма

Автоматически созданное описание

*Разработка автоматизированного рабочего места сотрудника онлайн-ресторана*

*КП Т.197020.401 ГЧ*

*Диаграмма классов*

КБП

У

Т. Контр.

Разраб. Разраб.

Тихомиров Е.В.

Провер. Провер.

Кривошеина А.В.

Реценз.

Утверд.

Н. Контр.

№ докум.

Подпись

Дата

Изм.

Лист

Масса

Лит.

Масштаб

Лист 2

Листов 4

Инв.№подл.

Подп. и дата

Взам.инв.№

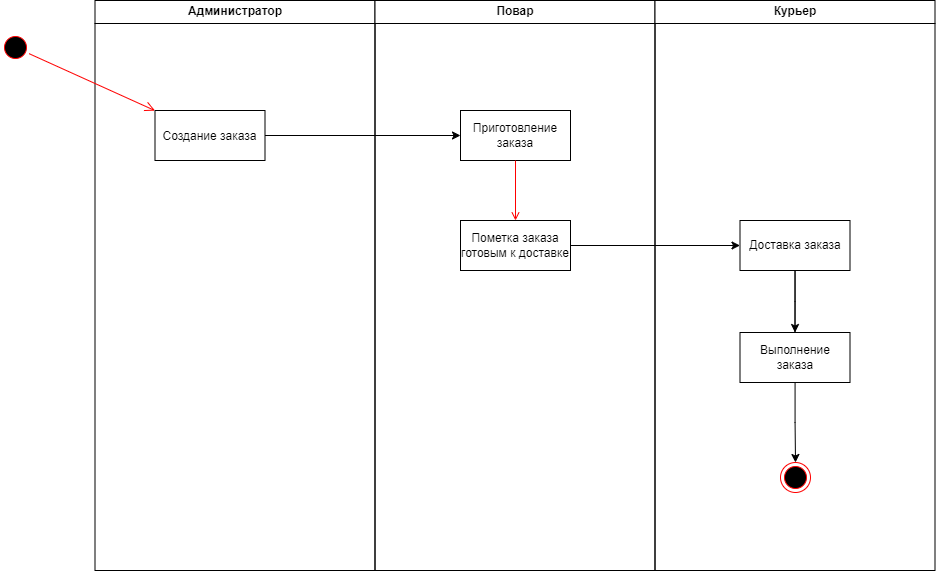
Инв.№дубл.

Подп. и дата

КП Т.197020.401 ГЧ Т.ХХХХХХ.401 ГЧ

*Разработка автоматизированного рабочего места сотрудника*

*онлайн-ресторана*

****

*КП Т.197020.401 ГЧ*

*Диаграмма деятельности (для функции составления заказа)*

КБП

У

Т. Контр.

Разраб. Разраб.

Тихомиров Е.В.

Провер. Провер.

Кривошеина А.В.

Реценз.

Утверд.

Н. Контр.

№ докум.

Подпись

Дата

Изм.

Лист

Масса

Лит.

Масштаб

Лист 3

Листов 4

Инв.№подл.

Подп. и дата

Взам.инв.№

Инв.№дубл.

Подп. и дата

КП Т.197020.401 ГЧ Т.ХХХХХХ.401 ГЧ

Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание

*Разработка автоматизированного рабочего места сотрудника*

*онлайн-ресторана*

*КП Т.197020.401 ГЧ*

*Диаграмма последовательности (для функции составления заказа)*

КБП

У

Т. Контр.

Разраб. Разраб.

Тихомиров Е.В.

Провер. Провер.

Кривошеина А.В.

Реценз.

Утверд.

Н. Контр.

№ докум.

Подпись

Дата

Изм.

Лист

Масса

Лит.

Масштаб

Лист 4

Листов 4

Инв.№подл.

Подп. и дата

Взам.инв.№

Инв.№дубл.

Подп. и дата

КП Т.197020.401 ГЧ Т.ХХХХХХ.401 ГЧ

**Удостоверяющий лист**

электронного документа – курсовой проект

Разработка автоматизированного рабочего места сотрудника бюро ремонта электронной техники

Тема

Обозначение \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_КП Т.197020.401\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кривошеина А.В.

Тихомиров Е.В.

Разработчик \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Ф.И.О.) (Ф.И.О.)

Подписи лиц, ответственных за разработку электронного документа

|  |  |  |
| --- | --- | --- |
| Состав электронного документа | Разработчик | Руководитель |
| Пояснительная записка (на бумажном носителе формата А4), Тихомиров\_Т-197\_ПЗ.docх |  |  |
| ГЧ, ДВИ.docx |  |  |
| ГЧ, Классов.docx |  |  |
| ГЧ, Деятельности.docx |  |  |
| ГЧ, Последовательности.docx |  |  |
| Папка с проектом «SynthFoods» |  |  |
| Установочный пакет программного средства «SynthFoods.exe» |  |  |
| Файлы базы данных «synthfoods.bacpac» |  |  |
| Тип носителя: флеш-накопитель |  |  |

**Этикетка**

для курсовых проектов

**Курсовой проект**

Разработка автоматизированного рабочего места сортудника бюра ремонта электронной техники

Тема «

»

КП Т.197020.401

Разработан

Утвержден 15.02.2024

Руководитель: Кривошеина А.В.

Технические средства: ПК (Windows 7, 8, 8.1, 10, 11), ноутбук (Windows 7, 8, 8.1, 10, 11, macOS)

Программные средства: SqlClient.dll, SqlServer.dll,

**Состав документа:**

Пояснительная записка – Тихомиров\_Т-197\_ПЗ.docх

Графическая часть – ДВИ.docx, Последовательности.docx, Классов.docx, Деятельности.docx

Папка с проектом – synthfoods

Установочный пакет программного средства – synthfoods

Файлы базы данных – synthfoods.db

Сведения о защите информации: администратор: логин hz, пароль 12345678