**Содержание**

ВВЕДЕНИЕ4

1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ И ПОСТАНОВКА ЗАДАЧИ5

2 ПРОЕКТИРОВАНИЕ10

2.1 Определение основных сущностей предметной области10

2.2 Построение схемы реляционной базы данных11

2.3 Описание физической модели данных14

2.4 Эскиз интерфейса19

3 ОПИСАНИЕ ПРОГРАММНОЙ РЕАЛИЗАЦИИ25

3.1 Описание структуры базы данных25

3.2 Описание задач автоматизации и интерфейса пользователя36

ВЫВОДЫ45

ПЕРЕЧЕНЬ ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ46

**ВВЕДЕНИЕ**

База данных–это совокупность связанных данных, отражающая состояние объектов и взаимосвязь между собой. Создание базы данных, её поддержка и обеспечение доступа пользователей к ней осуществляется с помощью специального программного инструмента – система управления базами данных (СУБД). СУБД –это комплекс языковых и программных средств, необходимых для создания, обработки и коллективного использования баз данных.

Основные функции СУБД:

1. Непосредственное управление данными во внешней памяти;
2. Управление буферами оперативной памяти;
3. Управление транзакциями;
4. Журнализация;
5. Поддержка языков БД.
6. **АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ**

На сегодняшний день в разных сферах деятельности люди всё больше нуждаются в средствах, позволяющих быстро и безошибочно перерабатывать большое количество информации. Применение таких средств позволяет существенно снизить затраты и повысить эффективность работы.

Создание базы данных пиццерии - это очень удобное решение для оптимизации работы и управления заведением.

В базе данных необходимо обеспечить хранение и обработку информации о следующих объектах: работники, продукты, меню, заказы.

Ведение учета продуктов состоит в следующем: предоставление перечня продуктов, их расчет расхода в течении дня, недели, месяца; формирование остатков продуктов на указанную дату в хранилище.

Приложение должно облегчить процесс формирования меню и его корректировку, в случае необходимости.

Разрабатываемой системе необходимо иметь ряд второстепенных функций для нормального функционирования заведения, а именно: простое добавление информации о работниках, возможность контролировать наличие блюд, расчет прибыли, рейтинга и статистики за определенный период.

В базе данных будет хранится информация о следующих объектах:

* Сотрудники:

1. ФИО: текст, 50 символов.
2. Дата рождения: дата в числовом формате, не может быть больше чем минус 18 лет от текущей даты.
3. Номер телефона: текст с ограничением в 10 символов, уникальный для каждого работника.
4. Адрес: текст, 150 символов.
5. Должность: выбор одной из перечня доступных: менеджер, официант, повар, охранник. Сотрудник может занимать в данный момент только одну должность.
6. Оклад: вещественное число, положительное, четыре цифры до запятой и две после.
7. Дни работы: может принимать одно из значений «Четные» или «Нечетные» дни.
8. Пароль: уникальный десятизначный пароль для входа в систему.

* Блюда:

1. Название: текст, 30 символов.
2. Порция: выбор между «Стандарт», «Маленькая», «Большая».
3. Цена: вещественное число, положительное, три цифры до запятой и две после.
4. Состав: перечень продуктов, текст, 200 символов.
5. Вес блюда в граммах: целое число, четыре цифры.
6. Калорийность: целое число, четыре цифры.
7. Категория: принимает одно из допустимых значений «Первое», «Второе», «Десерт», «Пицца».

* Напитки:

1. Название: текст, 30 символов.
2. Цена: вещественное число, положительное, три цифры до запятой и две после.
3. Объем: целое число положительное, три цифры.
4. Состав: перечень продуктов, текст, 200 символов.
5. Категория: принимает одно из допустимых значений «Холодные», «Горячие», «Алкогольные».

* Продукты в хранилище:

1. Наименование: текст, 30 символов.
2. Количество: целое число, положительное, три цифры.
3. Норма запаса: целое число, уникальное, три цифры.
4. Единицы измерения: принимает одно из допустимых значений «Штуки», «Килограммы», «Литры».

* Заказы

1. Номер заказа: целое число, уникальное, три цифры.
2. Блюда и напитка: перечень блюд и напитков, цены за один, их количество.
3. Имя заказчика: текст, 30 символов.
4. Номер столика: номер столика в зале.
5. Адрес: текст 100, в случае, когда заказ сделан по телефону, и нужна доставка еды курьером.
6. Сотрудник: имя обслужившего.
7. Дата и время принятия: тип даты и времени, не может быть меньше чем текущая дата и время, не может быть пустым.
8. Дата и время подачи: тип даты и времени, не может быть меньше чем текущая дата и время, и дата принятия, не может быть пустым.
9. Статус: выбор между «принят», «готовится», «Доставляется», «Выдан».
10. Номер карты постоянного клиента: целое 7 символов.

* Постоянные клиенты

1. Номер карты: целое число, уникальное, 7 цифр.
2. ФИО: текст, 30 символов.
3. Адрес: текст, 150 символов.
4. Скидка в процентах: целое число, 2 символа.

К системе будет доступ у трех должностных лиц: администратора, официанта, работника кухни.

Доступные действия для администратора:

* + - Просмотр статистики продаж блюд и напитков.
    - Просмотр рейтинга блюд.
    - Добавление информации о работнике.
    - Увольнение работника и удаление о нем информации.
    - Корректирование информации о напитках и блюдах, их удаление и добавление новых.

Доступные действия для администратора официанта:

* + - Формирование нового заказа.
    - Изменение информации определенного заказа.
    - Запрос на печать чека.
    - Добавление карты постоянного клиента.

Доступные действия для администратора официанта работника кухни:

* + - Просмотр сведений о продуктах и их количестве.
    - Изменения списка имеющихся блюд на текущую дату
    - Составление списка необходимых закупок продуктов.

В программе должны бать реализованы следующие функции:

1. Формирование меню.
2. Сортировка и фильтрация пунктов меню.
3. Создания заказа.
4. Формирование чека.
5. Учет продуктов.
6. Пополнение склада.
7. Формирование рейтинга блюд.
8. Статистические вычисления об количестве проданных блюд всего и определенных, за определенные периоды, и вычисление полученной прибыли.
9. Возможность добавления новых блюд.
10. Добавление новых напитков.
11. Возможность полного удаления блюд и напитков, без возможности восстановления.
12. Контроль за правильностью вводимых данных.
13. Добавление и удаление информации о сотрудниках.
14. Вход с разными уровнями доступа.

В базе данных необходимо контролировать следующую информацию:

* Невозможно поставить время доставки раньше времени принятия заказа.
* Нельзя взять на работу сотрудника возраст которого меньше восемнадцати лет.
* Доступные действия для сотрудников в системе должны строго соответствовать их должности.
* Невозможно создать два идентичных пункта меню.
* Невозможно продать блюдо, которого нет в наличии.
* Должны производиться проверка наличия продуктов в достаточном количестве.

В программе должна отправляться на печать следующая информация: чек, статистика за определенный период, список меню.

Строки в таблицах будут приблизительно в таком количестве: сотрудники - 30, блюда ­– 50, столы – 20, график работы работниками будет задаваться администраторами, при зачислении на работу, позже может редактироваться.

Отредактировать уже созданный заказ можно, но с определенными ограничениями: нельзя отменять заказанные блюда, если заказ уже в статусе «готовиться», можно только добавить блюда, также для заказов, которые находятся в статусе «Доставляется» нельзя изменять информацию. Изменить информацию заказа может любой официант.

Заключительным шагом является установление соответствия между сущностями, характеристиками предметной области, отношениями и атрибутами в нотации выбранной СУБД.

**2 ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ**

**2.1 Построение ER-диаграммы.**

На основе выполненного анализа предметной области была построена ER-диаграмма (рисунок 2.1).

Столик

Статус

Карта

Код

Время выд.

Заказчик

Адрес

Сотрудник

Время прин.

Код

ФИО

Дата рожд.

Номер тел.

Адрес

Чек

Работники

Товары

Состав

Заказ

Должность

Оклад

Дни работы

Пароль

Код

Калорийн.

Название

Порция.

Номер тел.

Вес/объем

Категория

Наличие

Код

Блюдо

Продукт

Количество

Код

Блюдо

Колич.

Категор.

Клиент

Код карты

ФИО

Адрес

Скидка

Рисунок 2.1- ER-диаграмма

**2.2 Построение схемы реляционной модели базы данных.**

На основании проведенного анализа предметной области и построенной ER- диаграммы создана реляционная модель базы данных (рисунок 2.2), которая отвечает всем требованиям нормализации отношений.

Id

Ингредиенты

Продукт

Напиток

Id

Категория

Блюдо

Категория

Id

Id

Работники

Id

Должность

Id

Чек

Заказ

Id

Заказчик

Id

Стол

Id

Id

Id

Id

Id

Id

Id

Рисунок 2.2 –Реляционная модель БД

Таблица2.1- Отношение КАТЕГОРИИ НАПИТКОВ

|  |  |
| --- | --- |
| Код категории | Категория напитка |

Таблица2.2.- Отношение КАТЕГОРИИ БЛЮД

|  |  |
| --- | --- |
| Код категории | Категория блюда |

Таблица2.3- Отношение ЧЕК

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Код чека | Имя закащика | Номер стола | Код сотрудника | Время принятия | Время выдачи | Адрес | Статус | Код карточки на скидку |

Таблица2.4- Отношение КАРТА НА СКИДКУ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Код карты | Фамилия | Имя | Адрес | Процент скидки |

Таблица2.5- Отношение БЛЮДА

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Код напитка | Название | Стоимость | Объем | Код категории | Можно ли купить |

Таблица2.6- Отношение НАПИТКИ

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Код блюда | Название | Порция | Стоимость | Масса | Калории | Код категории | Можно ли купить |

Таблица2.7- Отношение СОТРУДНИКИ

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Код сотрудника | Фамилия | Имя | Отчество | Дата рождения | Номер | Адрес | Код должности | Оклад | Дни работы | Пароль |

Таблица2.8- Отношение СОСТАВ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Код записи | Код блюда | Код напитка | Код продукта | Количество |

Таблица2.9- Отношение ДОЛЖНОСТИ

|  |  |
| --- | --- |
| Код должности | Название |

Таблица2.10- Отношение ЗАКАЗ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Код записи | Код блюда | Код напитка | Код чека | Количество |

Таблица2.11- Отношение ПРОДУКТЫ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Код продукта | Название | Количество | Единицы | Мин. количество |

Таблица2.12- Отношение СТОЛЫ

|  |  |
| --- | --- |
| Код стола | Название |

2**.3 Описание физической модели базы данных.**

В качестве СУБД была выбрана MySQL и визуальное приложение для работы с ней Workbench. Первое, что стоит отметить – Workbench дает возможность визуально проектировать базу данных, т.е. составлять схему БД. Визуальное представление БД всегда дает куда большую информацию, чем сухой список таблиц. В таком варианте сразу видно, каким образом связаны между собой таблицы, можно группировать таблицы по каким либо параметрами отражать это на схеме. Во-вторых, программа имеет встроенный редактор SQL-кода, с помощью которого можно быстро внести любые правки в SQL-запросы. При этом возможно строить запросы любой сложности, получать различные выборки из таблиц, связывать их, создавать новые таблицы и редактировать существующие, работать с ключами, полями, записями. В качестве языка был выбран C#.

Выделяются такие достоинства данного языка:

1.С# является полностью объектно-ориентированным языком, где даже типы, встроенные в язык, представлены классами.

2. Реализация, сочетающая построение надежного и эффективного кода, является немаловажным фактором, способствующему успеху C#.

3. Мощная библиотека каркаса поддерживает удобство построения различных типов приложения на C#, позволяя легко строить Web-службы, другие виды компонентов, достаточно просто сохранять и получать информацию из базы данных и других хранилищ данных.

С учетом особенностей выбранной СУБД спроектирована физическая модель для создания ранее реляционной модели (таблица 2.1). Физические модели БД:

Таблица 2.13-Физическая модель таблицы «Сотрудники (Employeers)»

|  |  |  |  |
| --- | --- | --- | --- |
| Имя столбца | Тип | Ограничения | Имя |
| Id | Int | AI, NN, PK | Код |
| Lname | Varchar(20) | NN | Фамилия |
| Fname | Varchar(20) | NN | Имя |
| Sname | Varchar(20) | NN | Отчество |
| Born | Data | NN | Дата рождения |
| Number | Varchar(15) | NN, UN | Номер телефона |
| Adress | Varchar(150) | NN | Адрес |
| Money | Int | NN,US | Зарплата |
| Job | Int | FK,NN | Код должности |
| Days | Enum(‘Четные’, ‘Нечетные’) | NN | Дни работы |
| Password | Varchar(10) | NN,UN | Пароль |

Таблица 2.14-Физическая модель таблицы «Должности (Jobs)

|  |  |  |  |
| --- | --- | --- | --- |
| Имя столбца | Тип | Ограничения | Имя |
| Id | int | PK, NN, AI | Код |
| Job | Varchar(20) | NN,UN | Должность |

Таблица 2.15-Физическая модель таблицы «Меню напитков (Drink)»

|  |  |  |  |
| --- | --- | --- | --- |
| Имя столбца | Тип | Ограничения | Имя |
| Id | Int | PK,NN,AI | Код |
| Name | Varchar(20) | NN | Название |
| Cost | Int | US,NN | Стоимость |
| Litrs | Int | US,NN | Объем |
| Category | Int | FK,NN | Код категории |
| CanShop | Int | NN | Наличие |

Таблица 2.16-Физическая модель таблицы «Меню блюд (Eat)

|  |  |  |  |
| --- | --- | --- | --- |
| Имя столбца | Тип | Ограничения | Имя |
| Id | Int | PK,NN,AI | Код |
| Name | Varchar(20) | NN | Название |
| Portion | Enum(«Маленькая», «Стандарт», «Большая) | -//- | Размер |
| Cost | Int | US,NN | Стоимость |
| Mass | Int | US,NN | Масса |
| Kaloryes | Int | US,NN | Калории |
| Category | Int | FK,NN | Код категории |
| CanShop | Int | NN | Наличие |

Таблица 2.17-Физическая модель таблицы «Категории напитков (KategoryDrink)»

|  |  |  |  |
| --- | --- | --- | --- |
| Имя столбца | Тип | Ограничения | Имя |
| Id | int | PK,AI,NN | Код |
| Category | Varchar(20) | NN | Категория напитков |

Таблица 2.18-Физическая модель таблицы «Категории блюд (KategoryEat)»

|  |  |  |  |
| --- | --- | --- | --- |
| Имя столбца | Тип | Ограничения | Имя |
| Id | int | PK,AI,NN | Код |
| Category | Varchar(20) | NN | Категория блюд |

Таблица 2.19-Физическая модель таблицы «Данные для чека (Checks)»

|  |  |  |  |
| --- | --- | --- | --- |
| Имя столбца | Тип | Ограничения | Имя |
| Id | int | PK,AI,NN | Код |
| Customer | Varchar(30) | NN | Заказчик |
| Table | int | NN,FK | Код стола |
| Продолжение таблицы 2.19 | | | |
| Имя столбца | Тип | Ограничения | Имя |
| Employeer | Int | FK, NN | Код сотрудника |
| DateOrder | DateTime | NN | Дата и время принятия |
| DateGive | DateTime | NN | Дата и время доставки |
| Adress | Varchar (100) | -//- | Адрес доставки  (Если нужно) |
| Status | Enum () | NN | Статус заказа |
| Customer | int | FK | Карта клиента |

Таблица 2.20-Физическая модель таблицы «Заказ (Orderss)»

|  |  |  |  |
| --- | --- | --- | --- |
| Имя столбца | Тип | Ограничения | Имя |
| Id | int | PK,AI,NN | Код |
| Dish | int | FK | Код блюда |
| Brew | Int | FK | Код напитка |
| Count | int | NN,US, default(1) | Количество |
| Check | int | NN,FK | Код чека |

Таблица 2.21-Физическая модель таблицы «Продукты (Products)»

|  |  |  |  |
| --- | --- | --- | --- |
| Имя столбца | Тип | Ограничения | Имя |
| Id | int | PK,AI,NN | Код |
| Name | Varchar(20) | NN | Название |
| Count | Int | US,NN | Количество единиц |
| Ones | Enum(«л»,«кг»,«шт») | NN | Единицы измерения |

Таблица 2.22-Физическая модель таблицы «Состав (Ingredients)»

|  |  |  |  |
| --- | --- | --- | --- |
| Имя столбца | Тип | Ограничения | Имя |
| Id | int | PK,AI,NN | Код |
| Dish | int | FK | Код блюда |
| Brew | Int | FK | Код напитка |
| Count | int | NN,US, default(1) | Количество |
| Product | int | NN,FK | Код продукта |

Таблица 2.23-Физическая модель таблицы «Tables)»

|  |  |  |  |
| --- | --- | --- | --- |
| Имя столбца | Тип | Ограничения | Имя |
| Id | int | PK,AI,NN | Код |
| Name | Varchar(20) | NN | Название |
| CanTake | Bool | NN | Доступен ли |

Таблица 2.24-Физическая модель таблицы «Клиенты (Customers)»

|  |  |  |  |
| --- | --- | --- | --- |
| Имя столбца | Тип | Ограничения | Имя |
| Id | int | PK,AI,NN | Код |
| Lname | Varchar(20) | NN | Фамилия |
| Fname | Varchar(20) | NN | Имя |
| Adress | Varchar(150) | NN | Адрес |
| Procent | int | NN, US | Процент скидки |

Сокращения:

NN ­­­­-- Not null;

US -- Unsigned;

UQ -- Unique;

AI -- Auto increment;

PK -- Primary key;

FK -- Foreign key.

**2.4 Эскиз интерфейса**

Эскиз интерфейса - это визуальный набросок будущего приложения. Он дает возможность сгруппировать объекты и правильно распределить функции. От удобства интерфейса зависит то, насколько клиент «привяжется» к системе, насколько ему удобно будет работать. Интерфейс приложения для работы с БД будет разработан в виде форм с кнопками, полями и переключателями. Далее на рисунках представлены эскизы окон, с которыми будет работать пользователь во время эксплуатации приложения.

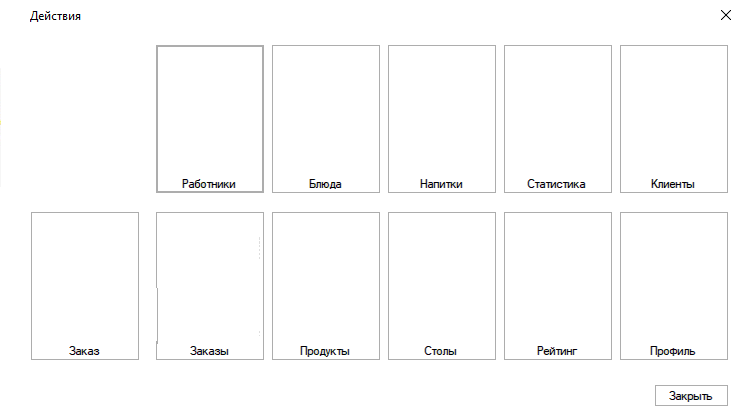


Рисунок 2.3 – Эскиз интерфейса главного меню

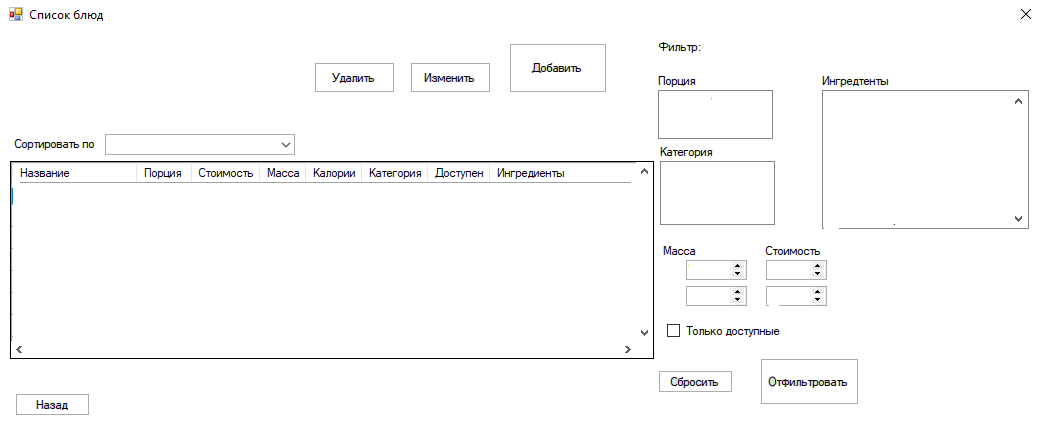


Рисунок 2.4 – Эскиз интерфейса списка блюд

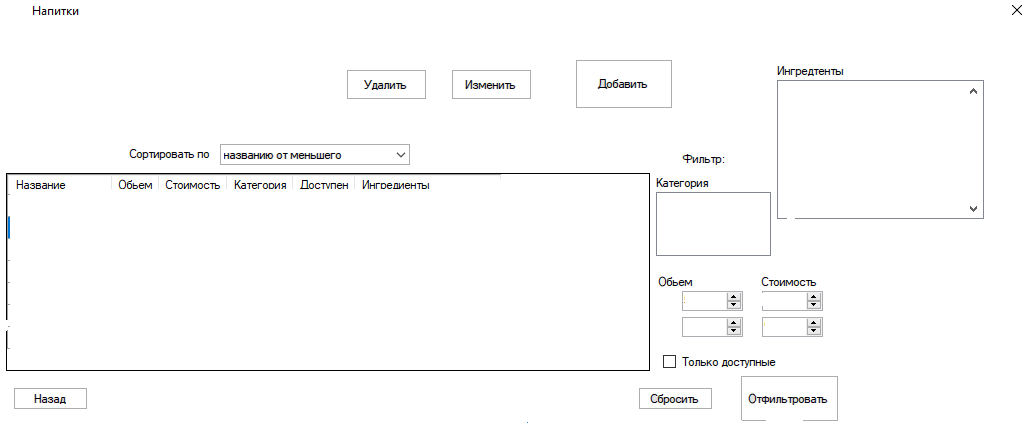


Рисунок 2.5 – Эскиз интерфейса списка напитков

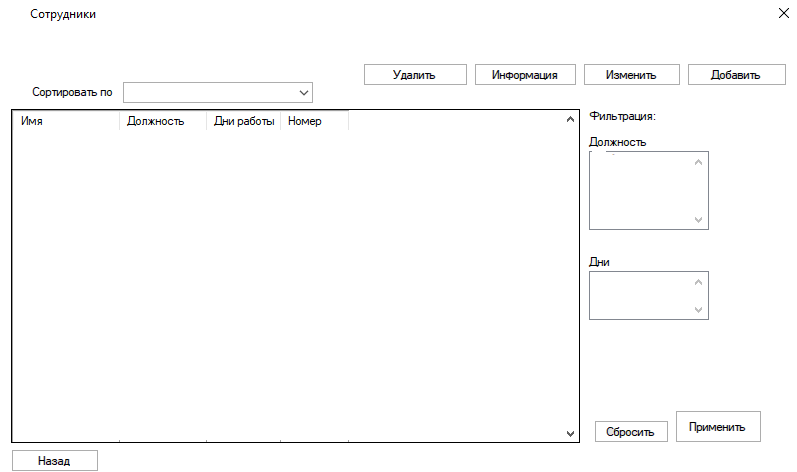


Рисунок 2.6 – Эскиз интерфейса списка сотрудников

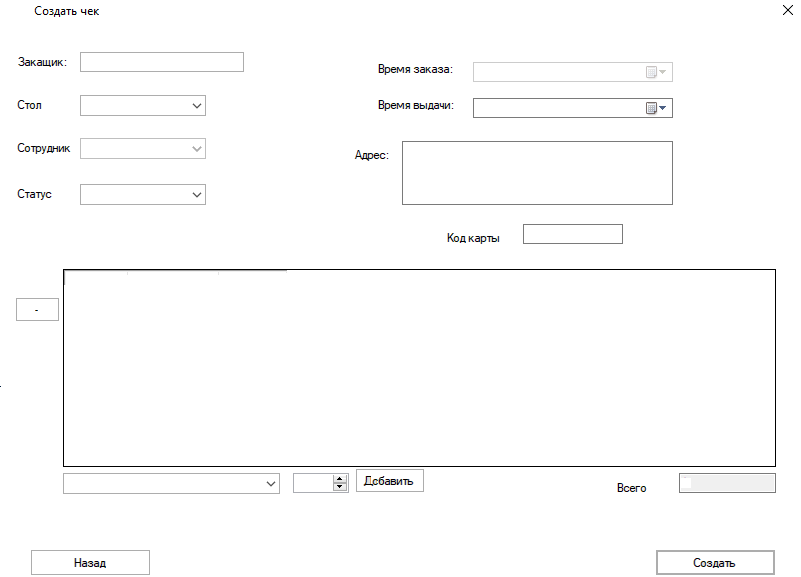


Рисунок 2.7 – Эскиз интерфейса создания нового заказа

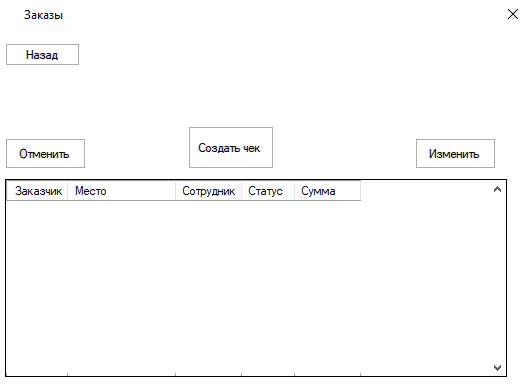


Рисунок 2.8 – Эскиз интерфейса списка заказов

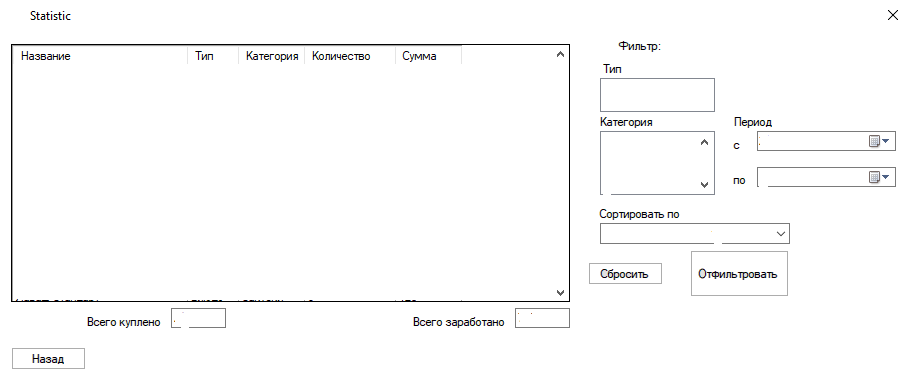


Рисунок 2.9 – Эскиз интерфейса для просмотра статистики

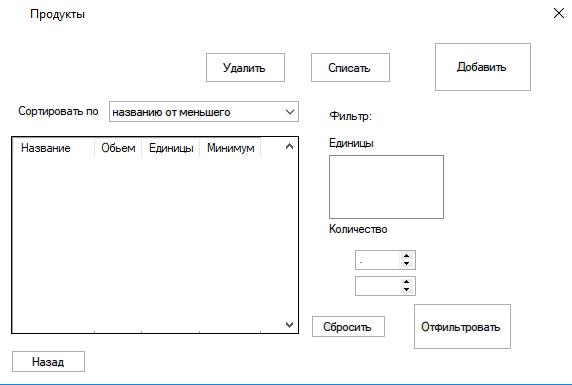


Рисунок 2.10 – Эскиз интерфейса списка продуктов

**3 ОПИСАНИЕ ПРОГРАММНОЙ РЕАЛИЗАЦИИ**

**3.1 Описание структуры базы данных**

В соответствии с физической моделью БД, была создана структура таблиц СУБД MySQL. Далее представлен текст из консоли MySQL-сервера, показывающий, в каком виде создавались и хранятся таблицы данной БД:

CREATE DATABASE IF NOT EXISTS `pizzeria` /\*!40100 DEFAULT CHARACTER SET utf8 \*/;

USE `pizzeria`;

-- MySQL dump 10.13 Distrib 8.0.13, for Win64 (x86\_64)

--

-- Host: 127.0.0.1 Database: pizzeria

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

-- Server version 8.0.13

/\*!40101 SET @OLD\_CHARACTER\_SET\_CLIENT=@@CHARACTER\_SET\_CLIENT \*/;

/\*!40101 SET @OLD\_CHARACTER\_SET\_RESULTS=@@CHARACTER\_SET\_RESULTS \*/;

/\*!40101 SET @OLD\_COLLATION\_CONNECTION=@@COLLATION\_CONNECTION \*/;

SET NAMES utf8 ;

/\*!40103 SET @OLD\_TIME\_ZONE=@@TIME\_ZONE \*/;

/\*!40103 SET TIME\_ZONE='+00:00' \*/;

/\*!40014 SET @OLD\_UNIQUE\_CHECKS=@@UNIQUE\_CHECKS, UNIQUE\_CHECKS=0 \*/;

/\*!40014 SET @OLD\_FOREIGN\_KEY\_CHECKS=@@FOREIGN\_KEY\_CHECKS, FOREIGN\_KEY\_CHECKS=0 \*/;

/\*!40101 SET @OLD\_SQL\_MODE=@@SQL\_MODE, SQL\_MODE='NO\_AUTO\_VALUE\_ON\_ZERO' \*/;

/\*!40111 SET @OLD\_SQL\_NOTES=@@SQL\_NOTES, SQL\_NOTES=0 \*/;

--

-- Table structure for table `categorydrink`

--

DROP TABLE IF EXISTS `categorydrink`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

SET character\_set\_client = utf8mb4 ;

CREATE TABLE `categorydrink` (

`Id` int(11) NOT NULL AUTO\_INCREMENT,

`Names` varchar(20) NOT NULL,

`Enable` int(11) NOT NULL DEFAULT '1',

PRIMARY KEY (`Id`)

) ENGINE=InnoDB AUTO\_INCREMENT=17 DEFAULT CHARSET=utf8;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Table structure for table `categoryeat`

--

DROP TABLE IF EXISTS `categoryeat`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

SET character\_set\_client = utf8mb4 ;

CREATE TABLE `categoryeat` (

`Id` int(11) NOT NULL AUTO\_INCREMENT,

`Names` varchar(20) NOT NULL,

`Enable` int(11) NOT NULL DEFAULT '1',

PRIMARY KEY (`Id`)

) ENGINE=InnoDB AUTO\_INCREMENT=7 DEFAULT CHARSET=utf8;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Table structure for table `checks`

--

DROP TABLE IF EXISTS `checks`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

SET character\_set\_client = utf8mb4 ;

CREATE TABLE `checks` (

`Id` int(11) NOT NULL AUTO\_INCREMENT,

`Customer` varchar(45) CHARACTER SET utf8 COLLATE utf8\_general\_mysql500\_ci NOT NULL,

`Table` int(11) DEFAULT NULL,

`Employeer` int(11) NOT NULL,

`DateOrder` datetime NOT NULL,

`DateGive` datetime NOT NULL,

`Adres` varchar(150) CHARACTER SET utf8 COLLATE utf8\_general\_mysql500\_ci DEFAULT NULL,

`Status` enum('Принят','Готовится','Доставляется','Выдан','Оплачен','Отменен') CHARACTER SET utf8 COLLATE utf8\_general\_mysql500\_ci NOT NULL,

`CardKey` int(11) DEFAULT NULL,

`Enable` int(11) NOT NULL DEFAULT '1',

PRIMARY KEY (`Id`),

KEY `Table\_idx` (`Table`),

KEY `Employeer\_idx` (`Employeer`),

KEY `CardKey\_idx` (`CardKey`),

CONSTRAINT `CardKey` FOREIGN KEY (`CardKey`) REFERENCES `customers` (`id`),

CONSTRAINT `Employeer` FOREIGN KEY (`Employeer`) REFERENCES `employeers` (`id`),

CONSTRAINT `Table` FOREIGN KEY (`Table`) REFERENCES `tables` (`id`)

) ENGINE=InnoDB AUTO\_INCREMENT=58 DEFAULT CHARSET=utf8 COLLATE=utf8\_general\_mysql500\_ci;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Table structure for table `customers`

--

DROP TABLE IF EXISTS `customers`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

SET character\_set\_client = utf8mb4 ;

CREATE TABLE `customers` (

`Id` int(11) NOT NULL AUTO\_INCREMENT,

`Lname` varchar(20) NOT NULL,

`Fname` varchar(20) NOT NULL,

`Adres` varchar(150) NOT NULL,

`Procent` int(11) NOT NULL,

`Enable` int(11) NOT NULL DEFAULT '1',

PRIMARY KEY (`Id`)

) ENGINE=InnoDB AUTO\_INCREMENT=9 DEFAULT CHARSET=utf8;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Table structure for table `drink`

--

DROP TABLE IF EXISTS `drink`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

SET character\_set\_client = utf8mb4 ;

CREATE TABLE `drink` (

`Id` int(11) NOT NULL AUTO\_INCREMENT,

`Names` varchar(30) NOT NULL,

`Volume` int(10) unsigned NOT NULL,

`Cost` int(10) unsigned NOT NULL,

`Category` int(11) NOT NULL,

`CanShop` int(11) NOT NULL,

`Enable` int(11) NOT NULL DEFAULT '1',

PRIMARY KEY (`Id`),

KEY `Category\_idx` (`Category`),

CONSTRAINT `CategoryDrink` FOREIGN KEY (`Category`) REFERENCES `categorydrink` (`id`)

) ENGINE=InnoDB AUTO\_INCREMENT=16 DEFAULT CHARSET=utf8;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Table structure for table `eat`

--

DROP TABLE IF EXISTS `eat`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

SET character\_set\_client = utf8mb4 ;

CREATE TABLE `eat` (

`Id` int(11) NOT NULL AUTO\_INCREMENT,

`Names` varchar(30) NOT NULL,

`Portion` enum('Маленькая','Стандарт','Большая') NOT NULL,

`Cost` int(10) unsigned NOT NULL,

`Mass` int(10) unsigned NOT NULL,

`Kaloryes` int(10) unsigned NOT NULL,

`Category` int(11) NOT NULL,

`CanShop` int(11) NOT NULL,

`Enable` int(11) NOT NULL DEFAULT '1',

PRIMARY KEY (`Id`),

KEY `Category\_idx` (`Category`),

CONSTRAINT `CategoryEat` FOREIGN KEY (`Category`) REFERENCES `categoryeat` (`id`)

) ENGINE=InnoDB AUTO\_INCREMENT=17 DEFAULT CHARSET=utf8;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Table structure for table `employeers`

--

DROP TABLE IF EXISTS `employeers`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

SET character\_set\_client = utf8mb4 ;

CREATE TABLE `employeers` (

`Id` int(11) NOT NULL AUTO\_INCREMENT,

`Lname` varchar(20) NOT NULL,

`Fname` varchar(20) NOT NULL,

`Sname` varchar(20) NOT NULL,

`Born` date NOT NULL,

`Number` varchar(15) NOT NULL,

`Adres` varchar(150) NOT NULL,

`Job` int(11) NOT NULL,

`Days` enum('Четные','Нечетные') NOT NULL,

`Password` varchar(10) DEFAULT NULL,

`Enable` int(11) NOT NULL DEFAULT '1',

PRIMARY KEY (`Id`),

UNIQUE KEY `Password\_UNIQUE` (`Password`),

KEY `Job\_idx` (`Job`),

CONSTRAINT `Job` FOREIGN KEY (`Job`) REFERENCES `job` (`id`)

) ENGINE=InnoDB AUTO\_INCREMENT=28 DEFAULT CHARSET=utf8;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Table structure for table `ingredients`

--

DROP TABLE IF EXISTS `ingredients`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

SET character\_set\_client = utf8mb4 ;

CREATE TABLE `ingredients` (

`Id` int(11) NOT NULL AUTO\_INCREMENT,

`Dish` int(11) DEFAULT NULL,

`Brew` int(11) DEFAULT NULL,

`Product` int(11) NOT NULL,

`Count` int(10) unsigned NOT NULL,

PRIMARY KEY (`Id`),

KEY `Brew\_idx` (`Brew`),

KEY `Dish\_idx` (`Dish`),

KEY `Product\_idx` (`Product`),

CONSTRAINT `Brews` FOREIGN KEY (`Brew`) REFERENCES `drink` (`id`),

CONSTRAINT `Dish` FOREIGN KEY (`Dish`) REFERENCES `eat` (`id`),

CONSTRAINT `Product` FOREIGN KEY (`Product`) REFERENCES `products` (`id`)

) ENGINE=InnoDB AUTO\_INCREMENT=77 DEFAULT CHARSET=utf8;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Table structure for table `job`

--

DROP TABLE IF EXISTS `job`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

SET character\_set\_client = utf8mb4 ;

CREATE TABLE `job` (

`Id` int(11) NOT NULL AUTO\_INCREMENT,

`Names` varchar(30) NOT NULL,

`Enable` int(11) NOT NULL DEFAULT '1',

PRIMARY KEY (`Id`)

) ENGINE=InnoDB AUTO\_INCREMENT=18 DEFAULT CHARSET=utf8;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Table structure for table `orders`

--

DROP TABLE IF EXISTS `orders`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

SET character\_set\_client = utf8mb4 ;

CREATE TABLE `orders` (

`Id` int(11) NOT NULL AUTO\_INCREMENT,

`Dish` int(11) DEFAULT NULL,

`Brew` int(11) DEFAULT NULL,

`Count` int(10) unsigned NOT NULL,

`Check` int(11) NOT NULL,

PRIMARY KEY (`Id`),

KEY `DishOrder\_idx` (`Dish`),

KEY `DrinkOrder\_idx` (`Brew`),

KEY `Check\_idx` (`Check`),

CONSTRAINT `Check` FOREIGN KEY (`Check`) REFERENCES `checks` (`id`),

CONSTRAINT `DishOrder` FOREIGN KEY (`Dish`) REFERENCES `eat` (`id`),

CONSTRAINT `DrinkOrder` FOREIGN KEY (`Brew`) REFERENCES `drink` (`id`)

) ENGINE=InnoDB AUTO\_INCREMENT=67 DEFAULT CHARSET=utf8;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Table structure for table `products`

--

DROP TABLE IF EXISTS `products`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

SET character\_set\_client = utf8mb4 ;

CREATE TABLE `products` (

`Id` int(11) NOT NULL AUTO\_INCREMENT,

`Names` varchar(45) NOT NULL,

`Count` int(10) unsigned NOT NULL,

`Ones` enum('шт','кг','л') NOT NULL,

`MinCount` int(10) unsigned NOT NULL,

`Enable` int(11) NOT NULL DEFAULT '1',

PRIMARY KEY (`Id`),

UNIQUE KEY `Names\_UNIQUE` (`Names`)

) ENGINE=InnoDB AUTO\_INCREMENT=27 DEFAULT CHARSET=utf8;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Table structure for table `tables`

--

DROP TABLE IF EXISTS `tables`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

SET character\_set\_client = utf8mb4 ;

CREATE TABLE `tables` (

`Id` int(11) NOT NULL AUTO\_INCREMENT,

`Names` varchar(45) NOT NULL,

`CanTake` int(11) NOT NULL DEFAULT '1',

`Enable` int(11) NOT NULL DEFAULT '1',

PRIMARY KEY (`Id`)

) ENGINE=InnoDB AUTO\_INCREMENT=15 DEFAULT CHARSET=utf8;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Dumping events for database 'pizzeria'

--

--

-- Dumping routines for database 'pizzeria'

--

/\*!40103 SET TIME\_ZONE=@OLD\_TIME\_ZONE \*/;

/\*!40101 SET SQL\_MODE=@OLD\_SQL\_MODE \*/;

/\*!40014 SET FOREIGN\_KEY\_CHECKS=@OLD\_FOREIGN\_KEY\_CHECKS \*/;

/\*!40014 SET UNIQUE\_CHECKS=@OLD\_UNIQUE\_CHECKS \*/;

/\*!40101 SET CHARACTER\_SET\_CLIENT=@OLD\_CHARACTER\_SET\_CLIENT \*/;

/\*!40101 SET CHARACTER\_SET\_RESULTS=@OLD\_CHARACTER\_SET\_RESULTS \*/;

/\*!40101 SET COLLATION\_CONNECTION=@OLD\_COLLATION\_CONNECTION \*/;

/\*!40111 SET SQL\_NOTES=@OLD\_SQL\_NOTES \*/;

-- Dump completed on 2019-03-10 17:12:15

**3.1 Описание задач автоматизации интерфейса**

Чтобы приступить к работе с приложением, для начала нужно запустить файл CourseWork. При открытии данного файла, появится окно авторизации.

Окно авторизации имеет следующий вид:

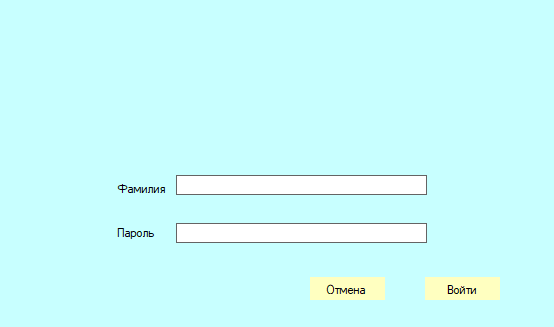


Рисунок 3.1 – Окно «Авторизация»

Данное окно содержит: поле для ввода пароля и логина, кнопки выхода и входа в систему.

После заполнения пользователем полей, ему необходимо нажать на кнопку «Войти» и система проверит правильность вводимых данных и, если все верно, перенаправит его на окно «Главное меню», в случае неправильности вводимых данных система известит пользователя об этом и потребует повторного заполнения полей.

Окно «Главное меню» загружается после авторизации и, в зависимости от должности пользователя может иметь разный функционал.

Главное меню со всеми возможными функциями имеет следующий вид: 

Рисунок 3.2- Окно «Главное меню»

Данное окно предназначено для организации простого и удобного доступа ко всем необходимым для работы действиям. Каждая кнопка отвечает за переход к конкретному окну для работы, кнопка «Закрыть» завершает работу с системой.

После нажатия на кнопку «Сотрудники» пользователь увидит следующую форму:

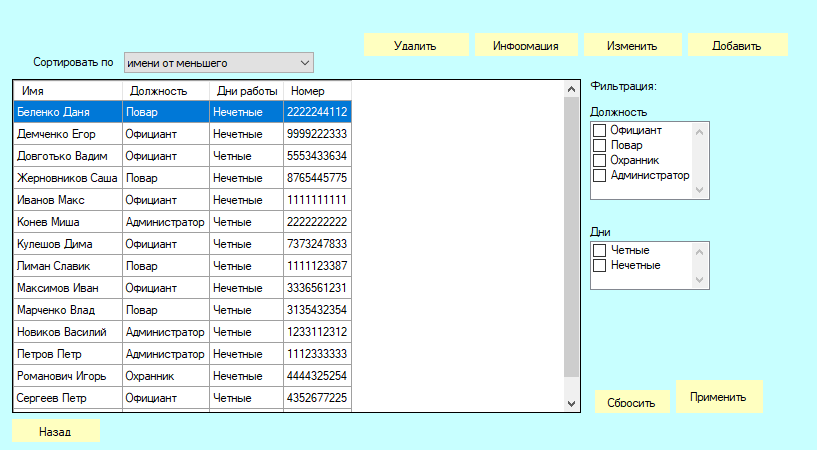


Рисунок 3.3 – Окно «Сотрудники»

Данное окно имеет следующий функционал:

1. Просмотр всех сотрудников заведения с их должностями и днями работы
2. Сортировка по имени, должности и дням работы.
3. Фильтрация по должности и дням работы.
4. После нажатия на кнопку «Информация», пользователь увидит детальные сведенья о выбранном сотруднике.
5. Кнопка «Удалить» отвечает за удаление всей информации о уволенном сотруднике.
6. Кнопки «Добавить» и «Изменить» перенаправляют администратора на новое окно в котором он может добавить нового сотрудника или изменить информацию о уже имеющемся

Окно для добавления пользователя имеет следующий вид:

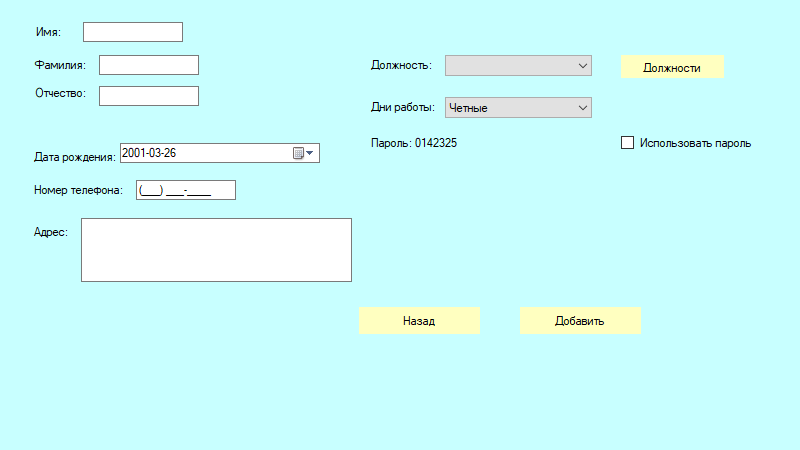


Рисунок 3.4 – Окно «Добавить пользователя»

В данном окне пользователю нужно заполнить все поля и нажать кнопку «Добавить» для сохранения информации о сотруднике, после чего, система вернет его на окно «Сотрудники» в котором будет уже обновленная информация.

После нажатия на кнопку «Блюда» в главном меню сотрудник увидит следующее окно:



Рисунок 3.5 – Окно «Блюда»

В данном окне отображается все блюда которые готовятся в пиццерии, а также предоставляется возможность фильтрации блюд по типу, весу, цене, порции и инградиентам входящих в их состав. Поле сортировки поможет выбрать критерий по которому следует отсортировать блюда.

Окно также иммет следующие основные кнопки:

1. Кнопка удаления блюда – после нажатия на нее система запросит подтверждение удаления выбраного блюда и, после согласия пользователя, удалит всю информацию о нем.
2. Кнопки «Добавить» и «Изменить» необходимы для добавления нового блюда и изменения информпции об уже имеющемся.

Окно добавления после нажатия на кнопку «Добавить» имеет следующий вид:

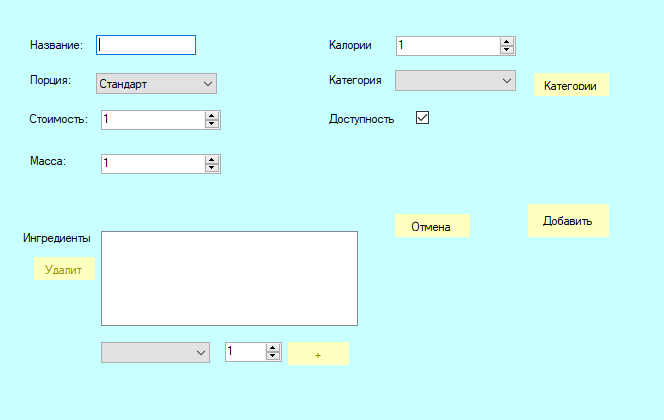


Рисунок 3.6 – Окно «Добавить болюдо»

В случае нажатия «Изменить» форма будет данными выделеного блюда:

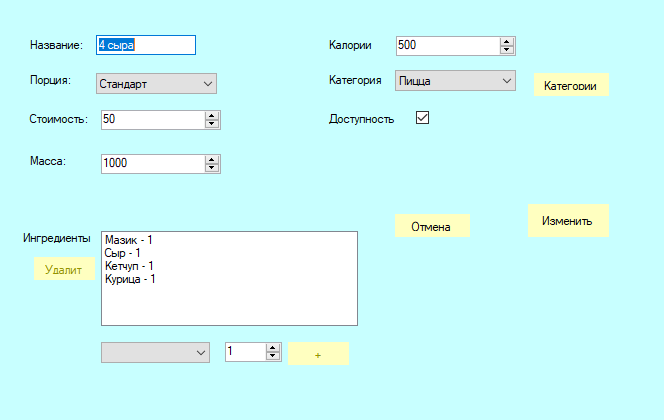


Рисунок 3.7 – Окно «Изменить блюдо»

При работе с данными окнами сотруднику нужно только заполнить поля и нажать кнопку «Добавить» или «Измненить» и система занесет новые данные в базу данны и обновить данные в окне.

Окно «Напитки» имеет функционал аналогичный форме с блюдами, с учетом некоторой корректировки под специфику напитков, оно имеет следующий вид:

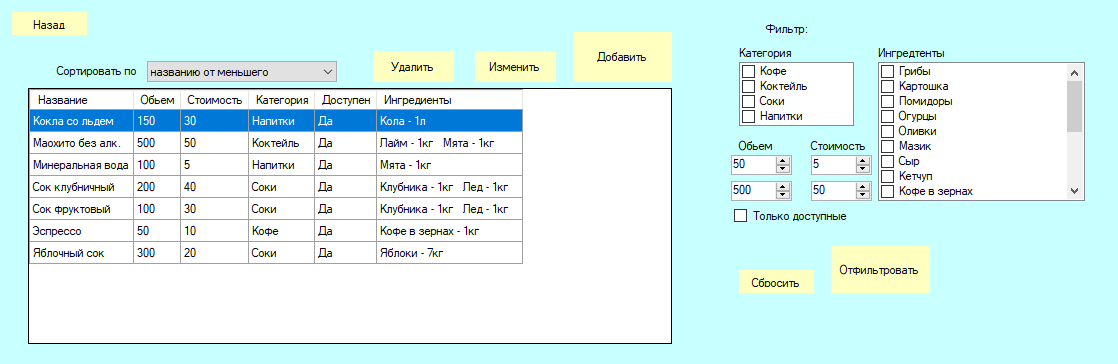


Рисунок 3.8 –Окно «Напитки»

Система также позволяет сотруднику увидеть статитсику, финансовые отчеты и рейтинг продаваемых товаров и блюд.

Окно статистики содержит следуюющую информацию:

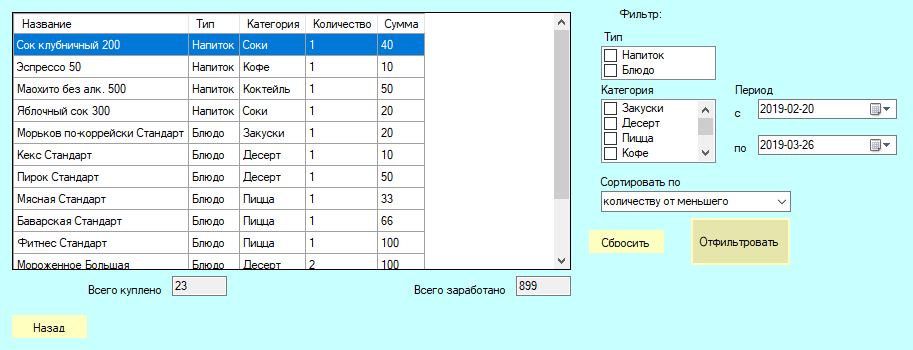


Рисунок 3.9 – Окно «Статистика»

На данной странице пользователем доступна информация о всех проданных блюдах и полученной прибыли за конкретыне периоды, также можно отфильтровать данные для удобства.

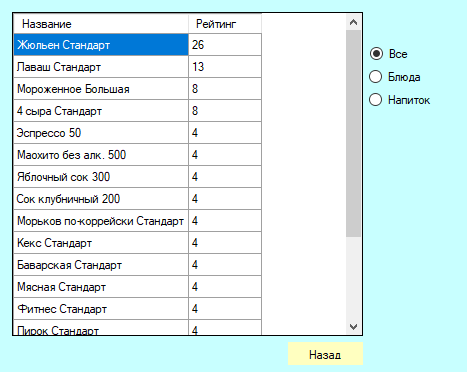


Рисунок 3.10 – Окно «Рейтинг»

Окно рейтинга показывает рейтинг блюд и напитков, который высчитывается по формуле: количствое продаж данного блюда разделить на количнство продаж всех блюд и умножить на 100. Данная информация полезна для оценки рентабельности и популярности продаваемых блюд.

Для оформления заказа официанту необходимо нажать на кнопку «Оформить заказ», после этого он будет перенапрвлен на специальное окно:

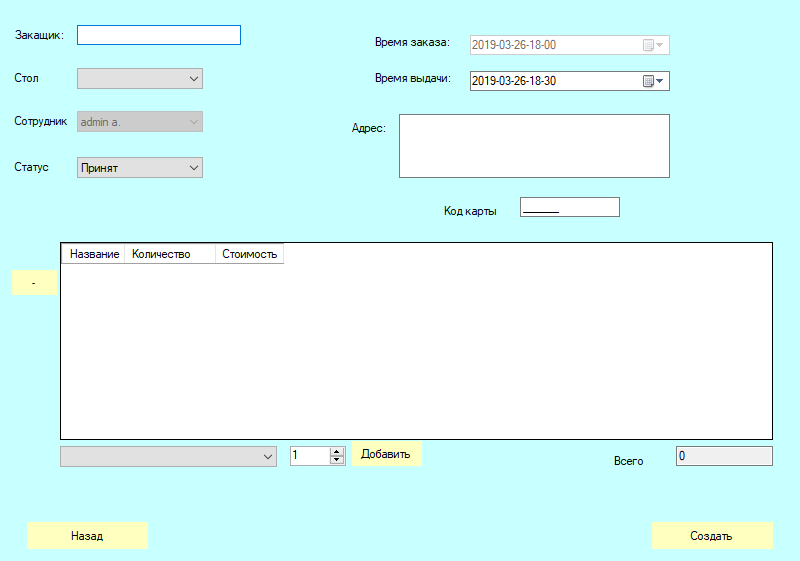


Рисунок 3.11 – окно «Принять заказ»

Вданном окне официант заполняет все необходимые поля и выбирает товары, которые заказал клиент, после чего отправляет данные на оформление и создается новый заказ.

Управлять заказми можно через окно со списком заказов, которое имеет следующий вид:

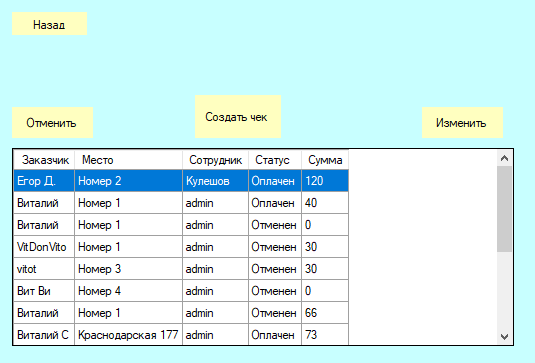


Рисунок 3.12 – Окно «Заказы»

Данное окно имеет следующий функционал:

1. Отмена выбраного заказа.
2. Создание чека для печати определенного заказа.
3. Изменения заказа, если позволяет его статус.

Также система предоставляет данные по складу с продуктами:

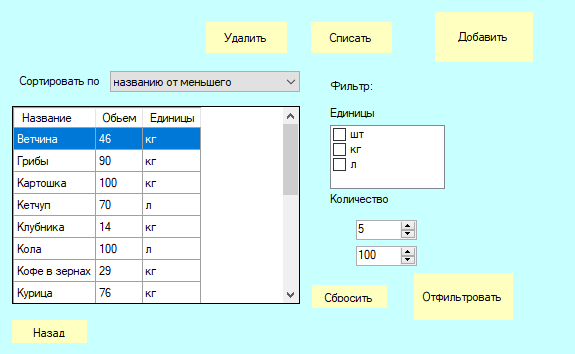


Рисунок 3.13 – Окно «Продукты»

В данном окне сотрудник может посмотреть список дотупных товаров на складе и их количество и закупить их при необходимости.

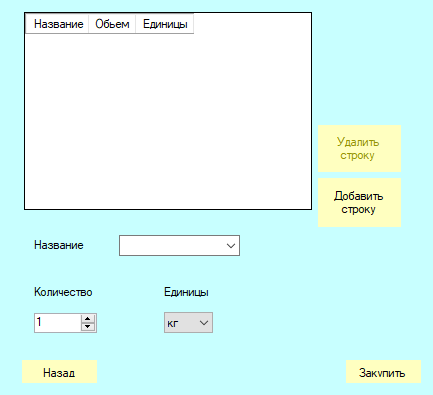


Рисунок 3.14 – Окно «Закупить продукты»

В этом окне сотрудник создает список покупок выбирая необходимые товары и их количество, после чего нажав на кнопку «Закупить», произойдет закупка и обновится список продуктов на складе.

**ВЫВОДЫ**

Результатом выполнения курсовой работы стало разработанная база данных и программа-приложение, которое может использоваться в пиццерии.

В данной курсовой работе были успешно реализованы все поставленные задачи, были подробно рассмотрены: задача курсового проекта, а также порядок выполнения задания.

Программа-приложение имеет интуитивно понятный интерфейс, что способствует быстрому внедрению программы. Программа имеет большое количество различных программных возможностей и ориентирована на пользователя, не имеющего большого опыта работы с программными продуктами такого типа. Это качество значительно расширяет круг возможных пользователей программы и увеличивает ее коммерческую привлекательность.

**ПЕРЕЧЕНЬ ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ**

1. Державний стандарт України. ДСТУ 3008-95 Документація. Звіти в сфері науки і техніки. Структура і правила оформлення.

2. Теория и практика построения баз данных. 8-еизд. / Д. Кренке. –СПб .: Питер, 2003.–800с.

3. Герберт Шилдт – С# 4.0. Полное руководство – Вильямс, 2011– 1056с.

4. Глушаков С.В., Ломотько Д.В. Базы данных. (Учебный курс).– Харьков, 2002.–504с.

5. Мартин Грабер .SQL.– К .: Ид-во «ЛОРИ», 2003.–644с.

6. Гарсиа-Молина, Гектор, Ульман, Джеффри, Уидом, Дженнифер, Системы баз данных. Полный курс .: Пер.сангл.– М .: Издательство дом «Вильямс», 2003.–1088с.

7. Аткинсон, Леон .MySQL. Библиотека профессионала .: Пер.сангл. –М .: Издательскийдом «Вильяме», 2002.–624с.