ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

***«*САНКТ-ПЕТЕРБУРГСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ПЕТРА ВЕЛИКОГО»**

Высшая школа программной инженерии

Изображение выглядит как текст

Автоматически созданное описание

**КУРСОВАЯ РАБОТА**

**“Разработка многопользовательской системы управления организацией”**

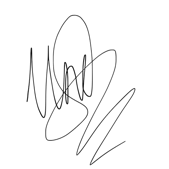
по дисциплине «Базы Данных»

Изображение выглядит как стрела

Автоматически созданное описание

Студент

3530202/90202 Потапова А. М.



Изображение выглядит как стрела

Автоматически созданное описание

Преподаватель Гасанова А. А.

­­

Санкт-Петербург

2021 г.

ЗАДАНИЕ

НА ВЫПОЛНЕНИЕ КУРСОВОГО ПРОЕКТА (КУРСОВОЙ РАБОТЫ)

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| студенту группы | *3530202/90202* | |  | *Потаповой Алине Михайловне* | | | | | | | |
|  | *(номер группы)* | |  | *(фамилия, имя, отчество)* | | | | | | | |
| ***1. Тема проекта (работы)*** | | | *Разработка многопользовательской* | | | | | | | | |
| *системы управления организацией.* | | | | | | | | | | | |
| *Объект автоматизации – ресторан. Задание №9* | | | | | | | | | | | |
| ***2. Срок сдачи студентом законченного проекта (работы)*** | | | | | | | | | | | *25.12.2021* |
| ***3. Исходные данные к проекту (работе)*** | | | | | | |  | | | | |
| *Описание предметной области* | | | | | | | | | | | |
| ***4. Содержание пояснительной записки***(перечень подлежащих разработке вопросов: введение, основная часть (раскрывается структура основной части), заключение, список использованных источников, приложения)*.* | | | | | | | | | | | |
| *Введение. Анализ предметной области. Проектирование схемы данных* | | | | | | | | | | | |
| *Реализация базы данных в среде SQL Server. Разработка представлений и* | | | | | | | | | | | |
| *Хранимых процедур. Разработка клиентского приложения. Тестирование* | | | | | | | | | | | |
| *Заключение. Список используемых источников* | | | | | | | | | | | |
| Примерный объем пояснительной записки | | | | | *15-20* | | | страниц машинописного | | | |
| текста | | | | | | | | | | | |
| 5. Перечень графического материала (с указанием обязательных чертежей и | | | | | | | | | | | |
| плакатов) | *не предоставляется* | | | | | | | | | | |
| 6. Консультанты | |  | | | | | | | | | |
|  | | | | | | | | | | | |
| 7. Дата получения задания: «11» сентября 2021 г. | | | | | | | | | | | |
|  | | | | | | | | | | | |
| Руководитель | | | |  | | |  | | Гасанова И.А. | | |
|  | | | | *(подпись)* | | |  | | *(инициалы, фамилия)* | | |
| Задание принял к исполнению | | | | Изображение выглядит как текст  Автоматически созданное описание | |  | | | Потапова А.М. | | |
|  | | | | *(подпись)* | |  | | | *(инициалы, фамилия)* | | |
|  | | | | | | | | | | 24.09.2021 | |
|  | | | | | | | | | | *(дата)* | |

**­­Оглавление**

[Анализ предметной области 4](#_Toc90486571)

[Схема и описание базы данных 7](#_Toc90486572)

[Запросы 11](#_Toc90486573)

[Описание программы 21](#_Toc90486574)

[Интерфейсы системы 24](#_Toc90486575)

[Заключение 37](#_Toc90486576)

[Список литературы 38](#_Toc90486577)

[Приложение. Код создания таблиц 39](#_Toc90486578)

# **Анализ предметной области**

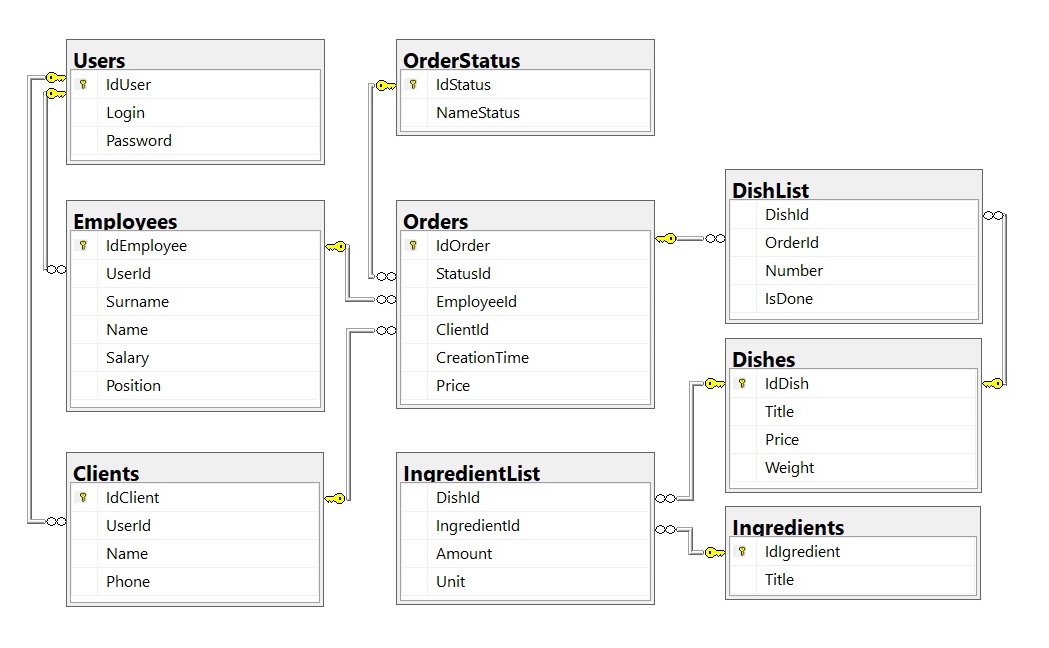
Группы пользователей разрабатываемой информационной системы (ИС)

|  |  |
| --- | --- |
| № пп | Наименование пользователя |
| 1 | Клиент |
| 2 | Работник кухни |
| 2 | Администратор ресторана |

Функции групп пользователей

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № пп | Выполняемая функция | | Входные данные | | Выходные данные | | Функции, которые должны быть реализованы в ИС |
| ***Покупатель*** | | | | | | | |
| 1 | Регистрация | Логин, пароль,  имя, телефон | | Таблица **Users,** **Clients** | | Добавление нового клиента | |
| 2 | Просмотр блюд в меню |  | | Таблица **Dishes** | | Вывод всех блюд в меню | |
| 3 | Просмотр самых популярных блюд в меню |  | | Таблица **Dishes** | | Вывод списка блюд по популярности | |
| 4 | Просмотр информации о блюде | ID блюда | | Таблицы **Dishes,**  **IngredientList** | | Вывод информации о выбранном блюде | |
| 5 | Создание заказа | ID выбранных (-ого) блюд(-а) | | Таблицы **Orders, DishList, Clients** | | Добавление нового заказа | |
| 6 | Просмотр информации о заказе | ID заказа | | Таблицы **Orders, OrderStatus, DishList, Dishes** | | Вывод информации о заказе | |
| 7 | Просмотр истории заказов |  | | Таблицы **Orders, DishList** | | Вывод всех созданных покупателем заказов | |
| 8 | Отмена созданного заказа | ID заказа | | Таблицы **Orders,** **OrderStatus** | | Отмена созданного заказа | |
| 9 | Редактирование персональной информации | Логин, пароль, имя, телефон | | Таблицы **Users, Clients** | | Изменение информации о клиенте | |
| ***Работник кухни*** | | | | | | | |
| 1 | Просмотр всех блюд в меню | |  | | Таблица **Dishes** | | Вывод всех блюд в меню |
| 2 | Просмотр заказов с определенным статусом | | ID заказа, ID статуса заказа | | Таблицы **Order****s,** **OrderStatus** | | Вывод списка заказов с выбранным статусом |
| 3 | Просмотр всех неприготовленных блюд | |  | | Таблицы **Orders**, **DishList, Dishes** | | Вывод списка всех неприготовленных блюд (со значением флага is\_done равным 0) |
| 4 | Просмотр всех приготовленных блюд | |  | | Таблицы **Orders**, **DishList, Dishes** | | Вывод списка всех приготовленных блюд (со значением флага is\_done равным 1) |
| 6 | Просмотр информации о заказе | | ID заказа | | Таблицы **Orders, OrderStatus, DishList** | | Вывод информации о заказе |
| 5 | Просмотр состава (блюд) заказа | | ID заказа | | Таблицы **Orders**, **DishList, Dishes, IngredientList, Ingredients** | | Вывод всех блюд в заказе |
| 7 | Отметка заказа принятым к исполнению | | ID заказа, ID статуса | | Таблицы **Orders, OrderStatus** | | Изменение статуса заказа на «принят» |
| 8 | Отметка блюда приготовленным | | ID блюда | | Таблицы **Dishes** | | Изменение значения флага is\_done в 1 |
| 9 | Отметка заказа приготовленным | | ID заказа, ID статуса | | Таблицы **Orders, OrderStatus** | | Изменение статуса заказа на «приготовлен» |
| 10 | Отметка заказа переданным покупателю | | ID заказа, ID статуса | | Таблицы **Orders, OrderStatus** | | Изменение статуса заказа на «выдан» |
| 11 | Редактирование персональной информации | | Логин, пароль | | Таблицы **Users, Employees** | | Изменение логина и пароля сотрудника |
| ***Администратор ресторана*** | | | | | | | |
| 1 | Просмотр всех блюд в меню | |  | | Таблица **Dishes** | | Вывод всех блюд в меню |
| 2 | Добавление нового блюда в меню | | Название, стоимость, состав (список ингредиентов) блюда | | Таблицы **Dishes, IngredientList, Ingredients** | | Добавление нового блюда в меню |
| 3 | Добавление нового ингредиента | | Название ингредиента | | Таблица **Ingredients** | | Добавление нового ингредиента |
| 4 | Редактирование информации об ингредиенте | | ID ингредиента, название, количество | | Таблица **Ingredients** | | Изменение информации об ингредиенте |
| 5 | Редактирование информации о блюде | | ID блюда, название, цена, состав (список ингредиентов) | | Таблицы **Dishes, IngredientList, Ingredients** | | Изменение информации о блюде |
| 6 | Редактирование информации о сотруднике | | ID сотрудника | | Таблица **Employees** | | Изменение информации о сотруднике |
| 7 | Просмотр списка покупателей | |  | | Таблица **Clients** | | Вывод всех покупателей |
| 8 | Просмотр всех сотрудников | |  | | Таблица **Employees** | | Вывод всех сотрудников |
| 9 | Регистрация нового сотрудника | | Имя, фамилия, логин, зарплата, должность, пароль | | Таблица **Users,** **Employees** | | Добавление нового сотрудника |
| 10 | Назначение сотрудника на исполнение заказа | | ID заказа, ID сотрудника | | Таблицы **Employees**, **Users,** **Orders** | | Добавление связи между заказом и сотрудником |
| 11 | Удаление сотрудника | | ID, имя и фамилия сотрудника | | Таблица **Users,** **Employees** | | Удаление сотрудника |
| 12 | Удаление ингредиента | | ID ингредиента | | Таблица **Ingredients** | | Удаление ингредиента |
| 13 | Удаление блюда из меню | | ID блюда | | Таблица **Dishes** | | Удаление блюда из меню |
| 14 | Редактирование персональной информации | | Логин, пароль | | Таблицы **Users, Employees** | | Изменение логина и пароля сотрудника |

# **Схема и описание базы данных**



Синтаксис описания связей:  
Номер. **Таблица 1** (поле в таблице 1) → **Таблица 2** (поле в таблице 2)

\*описание связи\*

1. **Orders** (StatusId) → **OrderStatus** (IdStatus)

Позволяет узнать статус заказа

2. **Orders** (EmployeeId) → **Employees** (IdEmployee)

Позволяет узнать информацию о сотруднике, который был назначен на исполнение заказа

3. **Orders** (ClientId) → **Clients** (IdClient)

Позволяет узнать информацию о клиенте, который совершил заказ

4. **DishList** (OrderId) → **Orders** (IdOrder)

Позволяет узнать информацию о списке блюд в заказе

5. **DishList** (DishId) → **Dishes** (IdDish)

Позволяет узнать информацию о блюде в заказе

6. **IngredientList** (DishId) → **Dishes** (IdDish)

Позволяет узнать информацию об ингредиентах блюда

7. **IngredientList** (IngredientId) → **Ingredients** (IdIngredient)

Позволяет узнать информацию о конкретном ингредиенте

*Описание таблиц*

**1. Users** – для хранения пользователей



**IdUser** – ID пользователя (PK, авто-инкремент)

**Login** – логин (NOT NULL, UNIQUE)

**Password** – пароль (NOT NULL)

**2.** **Clients** – для хранения клиентов

Изображение выглядит как стол

Автоматически созданное описание

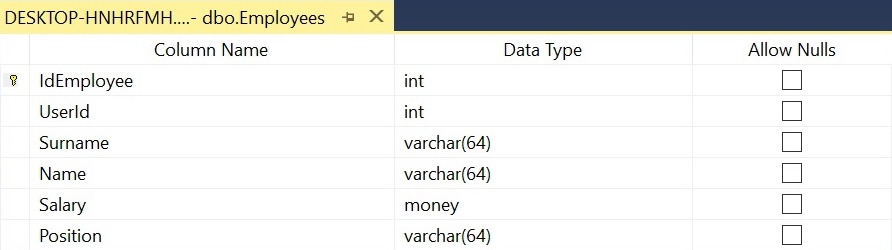
**IdClient** – ID клиента (PK, авто-инкремент)

**UserId** – ID пользователя (FK на Users)

**Name** – имя (NOT NULL)

**Phone** – номер телефона (NOT NULL, UNIQUE)

**3. Employees** – для хранения сотрудников



**IdEmployee** – ID сотрудника (PK, авто-инкремент)

**UserId** – ID пользователя (FK на Users)

**Surname** – фамилия (NOT NULL)

**Name** – имя (NOT NULL)

**Salary** – заработная плата (NOT NULL)

**Position** – должность (NOT NULL)

**4.** **Orders** – для хранения заказов

Изображение выглядит как стол

Автоматически созданное описание

**IdOrder** – ID заказа (PK, авто-инкремент)

**StatusId** – ID статуса (FK на OrderStatus)

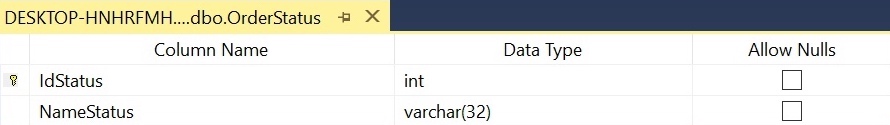
**EmployeeId** – ID сотрудника (FK на Employees)

**ClientId** – ID клиента (FK на Clients)

**CreationTime** – время создания заказа (DEFAULT ‘GETDATE()’)

**Price** – сумма заказа (NOT NULL)

**5.** **OrderStatus** – для хранения статусов заказа



**IdStatus** – ID статуса (PK, авто-инкремент)

**NameStatus** – название статуса (NOT NULL, UNIQUE)

**6.** **IngredientList** – для хранения списка ингредиентов блюда

Изображение выглядит как стол

Автоматически созданное описание

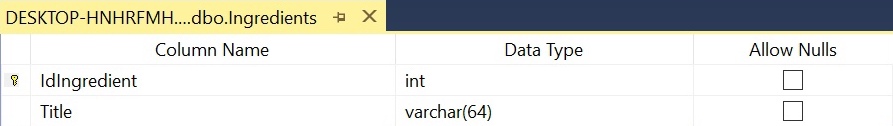
**DishId** – ID блюда (FK на Dishes)

**IngredientId** – ID блюда (FK на Ingredients)

**Amount** – количество ингредиента (NOT NULL)

**Unit** – единица измерения (NOT NULL)

**7.** **Ingredients** – для хранения ингредиентов



**­IdIngredient** – ID вагона (PK, авто-инкремент)

**Title** – название ингредиента (NOT NULL, UNIQUE)

**8.** **Dishes** – для хранения блюд

Изображение выглядит как стол

Автоматически созданное описание

**IdDish** – ID блюда (PK, авто-инкремент)

**Title** – название блюда (NOT NULL, UNIQUE)

**Price** – стоимость блюда (NOT NULL)

**9.** **DishList** – для хранения списка блюд в заказе

­­Изображение выглядит как стол

Автоматически созданное описание



**DishId** – ID блюда (FK на Dishes)

**OrderId** – ID заказа (FK на Orders)

**Number** – количество блюд в заказе (NOT NULL)

**IsDone** – готово ли блюдо (DEFAULT 0)

# **Запросы**

***Простые запросы***

1. Вывод всех блюд в меню

SELECT \* FROM Dishes

1. Вывод всех сотрудников

SELECT \* FROM Employees

1. Вывод всех ингредиентов

SELECT \* FROM Ingredients

1. Изменение значения флага, отвечающего за готовность блюда

DECLARE   
 @tempOrderId int,

@tempDoneDishId int,

@tempIsDone bit

UPDATE DishList SET DishList.IsDone = @tempIsDone  
WHERE DishList.OrderId = @tempOrderId AND DishList.DishId = @tempDoneDishId

1. Изменение информации об ингредиенте

DECLARE   
 @tempIngredientId int,   
 @tempIngredientTitle varchar(64)

UPDATE Ingredients SET Title = @tempIngredientTitle   
WHERE IdIngredient = @tempIngredientId

1. Изменение информации о блюде

DECLARE   
 @tempDishId int,   
 @tempDishTitle varchar(64),   
 @tempDishPrice money,   
 @tempDishWeight int

UPDATE Dishes   
SET Title = @tempDishTitle, Price = @tempDishPrice, Weight = @tempDishWeight   
WHERE IdDish = @tempDishId

1. Изменение информации об ингредиенте блюда

DECLARE

@tempDishId int,

@amount int,

@unit varchar(20)

UPDATE IngredientList

SET Amount = @amount, Unit = @unit

WHERE DishId = @tempDishId

1. Изменение информации о сотруднике

DECLARE

@tempEmoloyeeId int,   
@tempEmoloyeeSurname varchar(64),   
@tempEmoloyeeName varchar(64),   
@tempEmoloyeeSalary money,   
@tempEmoloyeePosition varchar(64)

UPDATE Employees   
SET Surname = @tempEmoloyeeSurname, Name = @tempEmoloyeeName, Salary = @tempEmoloyeeSalary, Position = @tempEmoloyeePosition   
WHERE IdEmployee = @tempEmoloyeeId

1. Удаление сотрудника

DECLARE

@surname varchar(64),

@name varchar(64)  
DELETE FROM Employees   
WHERE Employees.Surname = @surname AND [Employees.Name](https://vk.com/away.php?utf=1&to=http%3A%2F%2FEmployees.Name) = @name

1. Удаление ингредиента

DECLARE

@delIngredientTitle varchar(64)   
DELETE FROM Ingredients   
WHERE Ingredients.Title = @delIngredientTitle

1. Удаление блюда из меню

DECLARE

@delDishTitle varchar(64)   
DELETE FROM Dishes   
WHERE Dishes.Title = @delDishTitle 

***Сложные запросы***

1. Вывод всех блюд в заказе

— Клиент   
SELECT Dishes.Title, DishList.Number, Dishes.Price \* DishList.Number AS 'Total Price' FROM Dishes   
JOIN DishList ON DishList.DishId = Dishes.IdDish   
JOIN Orders ON Orders.IdOrder = DishList.OrderId AND Orders.IdOrder = @order\_id   
JOIN OrderStatus ON Orders.StatusId = OrderStatus.IdStatus

— Сотрудник   
SELECT Dishlist.DishId, Dishes.Title, DishList.Number, DishList.IsDone

FROM Dishes   
JOIN DishList ON DishList.DishId = Dishes.IdDish   
JOIN Orders ON Orders.IdOrder = DishList.OrderId AND Orders.IdOrder = @order\_id    
JOIN OrderStatus ON Orders.StatusId = OrderStatus.IdStatus

1. Вывод информации о блюде

— Клиент   
SELECT Ingredients.Title FROM Ingredients   
JOIN IngredientList ON IngredientList.IngredientId = Ingredients.IdIngredient   
JOIN Dishes ON Dishes.IdDish = IngredientList.DishId AND Dishes.IdDish = @dish\_id

— Сотрудник   
SELECT Ingredients.Title, IngredientList.Amount, IngredientList.Unit FROM Ingredients   
JOIN IngredientList ON IngredientList.IngredientId = Ingredients.IdIngredient   
JOIN Dishes ON Dishes.IdDish = IngredientList.DishId AND Dishes.IdDish = @dish\_id

1. Вывод информации о заказе

— Клиент

SELECT OrderStatus.NameStatus, FORMAT(Orders.CreationTime, 'dd.MM.yyyy HH:mm') AS 'Creation Time', Orders.Price FROM Orders   
JOIN OrderStatus ON Orders.StatusId = OrderStatus.IdStatus AND Orders.IdOrder = @order\_id   
JOIN Employees ON Orders.EmployeeId = Employees.IdEmployee   
JOIN Clients ON Orders.ClientId = Clients.IdClient

— Сотрудник

SELECT OrderStatus.NameStatus, Employees.Surname AS 'Employee', Clients.Name AS 'Client', Clients.Phone, FORMAT(Orders.CreationTime, 'dd.MM.yyyy HH:mm') AS 'Creation Time', Orders.Price FROM Orders   
JOIN OrderStatus ON Orders.StatusId = OrderStatus.IdStatus AND Orders.IdOrder = @order\_id  
JOIN Employees ON Orders.EmployeeId = Employees.IdEmployee   
JOIN Clients ON Orders.ClientId = Clients.IdClient

1. Вывод всех созданных пользователем заказов

SELECT Orders.IdOrder, OrderStatus.NameStatus, FORMAT(Orders.CreationTime, 'dd.MM.yyyy HH:mm') AS 'Creation Time', Orders.Price   
FROM Orders   
JOIN OrderStatus ON OrderStatus.IdStatus = Orders.StatusId   
JOIN Clients ON Clients.IdClient = Orders.ClientId AND Clients.IdClient = @client\_id

1. Вывод списка заказов с определенным статусом

ALTER PROCEDURE OrderListByStatus   
@status varchar(64)   
AS   
BEGIN   
SELECT Orders.IdOrder, FORMAT(Orders.CreationTime, 'dd.MM.yyyy HH:mm')

AS 'Creation Time', Clients.Name AS 'Client', Clients.Phone, Orders.Price

FROM Orders   
JOIN OrderStatus ON Orders.StatusId = OrderStatus.IdStatus AND NameStatus = @status   
JOIN Employees ON Orders.EmployeeId = Employees.IdEmployee   
JOIN Clients ON Orders.ClientId = Clients.IdClient   
ORDER BY Orders.CreationTime DESC   
END; 

1. Вывод всех неприготовленных блюд

SELECT Dishes.Title, DishList.Number FROM Dishes   
JOIN DishList ON DishList.IsDone = 0 AND DishList.DishId = Dishes.IdDish

1. Вывод всех приготовленных блюд

SELECT Dishes.Title, DishList.Number FROM Dishes   
JOIN DishList ON DishList.IsDone = 1 AND DishList.DishId = Dishes.IdDish

1. Добавление связи между сотрудником и заказом

DECLARE   
 @orderId int,   
 @employeeName varchar(64),   
 @employeeSurname varchar(64)    
  
UPDATE Orders SET EmployeeId = (SELECT Employees.IdEmployee

FROM Employees WHERE Name = @employeeName AND Surname = @employeeSurname)   
WHERE IdOrder = @orderId

1. Изменение значения статуса заказа

DECLARE   
 @orderId int,   
 @statusName varchar(64)

UPDATE Orders SET Orders.StatusId = (SELECT IdStatus FROM OrderStatus WHERE NameStatus = @statusName)   
WHERE Orders.IdOrder = @orderId

1. Вывод всех заказов

SELECT Orders.IdOrder, FORMAT(Orders.CreationTime, 'dd.MM.yyyy HH:mm') AS 'Creation Time', Clients.Name AS 'Client', Clients.Phone, Orders.Price, OrderStatus.NameStatus   
FROM Orders   
JOIN OrderStatus ON Orders.StatusId = OrderStatus.IdStatus   
JOIN Employees ON Orders.EmployeeId = Employees.IdEmployee   
JOIN Clients ON Orders.ClientId = Clients.IdClient   
ORDER BY Orders.CreationTime DESC

1. Вывод списка покупателей

CREATE VIEW Clients\_List

AS

(SELECT Clients.Name, Clients.Phone, COUNT(Orders.IdOrder) AS OrderCount FROM Clients

JOIN Orders ON Orders.ClientId = Clients.IdClient

GROUP BY Clients.name, Clients.Phone)

UNION ALL

(SELECT Clients.Name, Clients.Phone, 0 AS OrderCount FROM Clients

WHERE NOT EXISTS (SELECT \* FROM Orders WHERE Orders.ClientId = Clients.IdClient))

***Представления***

1. Вывод всех блюд по убыванию их популярности.

ALTER VIEW TopDishes   
AS   
(SELECT TOP 100 Dishes.IdDish, Dishes.Title, Dishes.Weight, Dishes.Price FROM Dishes JOIN DishList ON DishList.DishId = Dishes.IdDish GROUP BY Dishes.IdDish, Dishes.Title, Dishes.Weight, Dishes.Price HAVING SUM(DishList.Number) > 0

ORDER BY SUM(DishList.Number) DESC)   
UNION ALL   
(SELECT TOP 100 Dishes.IdDish, Dishes.Title, Dishes.Weight, Dishes.Price

FROM Dishes WHERE NOT EXISTS (SELECT \* FROM DishList WHERE DishList.DishId = Dishes.IdDish) ORDER BY Dishes.Title DESC)

1. Вывод всех клиентов по убыванию количества заказов.

ALTER VIEW TopClients   
AS   
(SELECT TOP 100 Clients.Name, Clients.Phone, COUNT(Orders.IdOrder) AS OrderCount FROM Clients JOIN Orders ON Orders.ClientId = Clients.IdClient

GROUP BY Clients.name,   
ORDER BY OrderCount DESC)   
UNION ALL   
(SELECT TOP 100 Clients.Name, Clients.Phone, 0 AS OrderCount FROM Clients   
WHERE NOT EXISTS (SELECT \* FROM Orders WHERE Orders.ClientId = Clients.IdClient))

***Хранимые процедуры***

1. Регистрация сотрудника

CREATE PROCEDURE RegEmployee   
 @login varchar(64),   
 @password varchar(64),   
 @surname varchar(64),   
 @name varchar(64),   
 @salary money,   
 @position varchar(64)   
AS

BEGIN TRY

BEGIN TRANSACTION

INSERT INTO Users(login, password)

VALUES (@login, @password)

INSERT INTO Employees(UserId, Surname, Name, Salary, Position)s

VALUES (SCOPE\_IDENTITY(), @surname, @name, @salary, @position)

SELECT IdEmployee FROM Employees WHERE IdEmployee = SCOPE\_IDENTITY()  
COMMIT TRANSACTION

END TRY

BEGIN CATCH

ROLLBACK TRANSACTION;

END CATCH

1. Регистрация клиента

CREATE PROCEDURE RegClient   
 @login varchar(64),   
 @password varchar(64),   
 @name varchar(64),   
 @phone varchar(15)   
AS   
BEGIN TRY

BEGIN TRANSACTION  
 INSERT INTO Users(login, password)   
 VALUES (@login, @password)   
 INSERT INTO Clients(UserId, Name, Phone)   
 VALUES (SCOPE\_IDENTITY(), @name, @phone)

SELECT IdClient FROM Clients WHERE IdClient = SCOPE\_IDENTITY()

COMMIT TRANSACTION

END TRY

BEGIN CATCH

ROLLBACK TRANSACTION;

END CATCH

1. Добавление ингредиента

CREATE PROCEDURE AddIngredient   
 @title varchar(64)   
AS   
BEGIN TRY

BEGIN TRANSACTION  
 INSERT INTO Ingredients(Title)   
 VALUES (@title)

SELECT IdIngredient FROM Ingredients WHERE IdIngredient = SCOPE\_IDENTITY()  
COMMIT TRANSACTION

END TRY

BEGIN CATCH

ROLLBACK TRANSACTION;

END CATCH

1. Добавление блюда с ингредиентами

CREATE PROCEDURE AddDish

@title varchar(64),

@price money,

@weight int

AS

BEGIN TRY

BEGIN TRANSACTION

INSERT INTO Dishes(Title, Weight, Price)

VALUES (@title, @weight, @price)

SELECT IdDish FROM Dishes WHERE IdDish = SCOPE\_IDENTITY()

COMMIT TRANSACTION

END TRY

BEGIN CATCH

ROLLBACK TRANSACTION;

END CATCH

CREATE PROCEDURE AddDishIngredient

@dishTitle varchar(64),

@ingredientTitle varchar(64),

@amount int,

@unit varchar(20)

AS

BEGIN

INSERT INTO IngredientList(DishId, IngredientId, Amount, Unit)

VALUES ((SELECT IdDish FROM Dishes WHERE Title = @dishTitle), (SELECT IdIngredient FROM Ingredients WHERE Title = @ingredientTitle) , @amount, @unit)

END;

1. Добавление заказа

CREATE PROCEDURE AddOrder   
@clientId int   
AS   
BEGIN   
 INSERT INTO Orders(ClientId, StatusId, Price)   
 VALUES (@clientId, (SELECT IdStatus FROM OrderStatus

WHERE NameStatus = 'Обрабатывается'),

(SELECT SUM(Dishes.Price \* DishList.Number) FROM Dishes, DishList

WHERE DishList.DishId = Dishes.IdDish AND DishList.OrderId IS NULL))   
 UPDATE DishList SET DishList.OrderId = SCOPE\_IDENTITY()   
 WHERE DishList.OrderId IS NULL   
END;

CREATE PROCEDURE AddDishInDishList   
@dishTitle varchar(64),   
@number int   
AS   
BEGIN   
 INSERT INTO DishList(DishId, Number, IsDone)   
 VALUES ((SELECT IdDish FROM Dishes WHERE Title = @dishTitle), @number, '0')   
END;

1. Вход в систему

CREATE PROCEDURE log\_in

@login varchar(64),

@password varchar(64)

AS   
BEGIN

(SELECT IdClient AS Id, 0 AS Role FROM Clients

JOIN Users ON Clients.UserId = Users.IdUser WHERE Users.Login = @login AND Users.Password = @password)

UNION ALL

(SELECT IdEmployee AS Id, 1 AS Role FROM Employees

JOIN Users ON Employees.UserId = Users.IdUser WHERE Users.Login = @login AND Users.Password = @password AND Employees.Position != 'Администратор')

UNION ALL

(SELECT IdEmployee AS Id, 2 AS Role FROM Employees

JOIN Users ON Employees.UserId = Users.IdUser WHERE Users.Login = @login AND Users.Password = @password AND Employees.Position = 'Администратор')

END;

1. Изменение информации о клиенте

CREATE PROCEDURE changeClientInfo

@clientId int,

@login varchar(64),

@password varchar(64),

@name varchar(64),

@phone varchar(16)

AS

BEGIN TRY

BEGIN TRANSACTION

UPDATE Users SET Login = @login, Password = @password

WHERE IdUser = (SELECT UserId From Clients WHERE IdClient = @clientId)

UPDATE Clients SET Name = @name, Phone = @phone

WHERE IdClient = @clientId

COMMIT TRANSACTION

END TRY

BEGIN CATCH

ROLLBACK TRANSACTION;

END CATCH

1. Изменение информации о сотруднике

CREATE PROCEDURE changeEmployeeUser

@employeeId int,

@login varchar(64),

@password varchar(64)

AS

BEGIN

UPDATE Users SET Login = @login, Password = @password

WHERE IdUser = (SELECT UserId From Employees WHERE IdEmployee = @employeeId)

END;

# **Описание программы**

1. Среда разработки: Visual Studio Code
2. Языки программирования: JavaScript, HTML, CSS
3. Движок: Node.js
4. Фреймворки:

* Tedious
* Express
* Body-parser

1. СУБД: Microsoft SQL Server

Предлагаю вам подробно ознакомиться со всеми применяемыми мною инструментами по ходу проектирования ПО.

|  |  |
| --- | --- |
| Инструмент | Описание |
| Среда разработки | **Visual Studio Code** — редактор исходного кода, разработанный Microsoft для Windows, Linux и macOS. Позиционируется как «лёгкий» редактор кода для кроссплатформенной разработки веб- и облачных приложений. |
| Языки программирования | **JavaScript** — мультипарадигменный язык программирования. Поддерживает объектно-ориентированный, императивный и функциональный стили. Обычно используется как встраиваемый язык для программного доступа к объектам приложений.  **HTML** — стандартизированный язык разметки документов для просмотра веб-страниц в браузере. В моем случае веб-браузер получает HTML документ от сервера по протоколам с локального диска, далее интерпретирует код в интерфейс, который будет отображаться на экране монитора.  **CSS** — формальный язык описания внешнего вида документа, написанного с использованием языка разметки. |
| Движок | **Node.js** — среда выполнения кода на JavaScript, которая построена на основе движка JavaScript Chrome V8, который позволяет транслировать вызовы на языке JavaScript в машинный код. Node.js прежде всего предназначен для создания серверных приложений на языке JavaScript |
| Фреймворки | **Tedious** — это фреймворк Node.js, который обеспечивает реализацию протокола TDS, который используется для взаимодействия с экземплярами Microsoft SQL Server.  **Express** — это минималистичный и гибкий веб-фреймворк для приложений Node.js, предоставляющий обширный набор функций для мобильных и веб-приложений.  **Body-parser** — это пакет, который позволяет разбирать тела запросов, хранящиеся в req.body, выступая в роли ПО промежуточного слоя для серверов, основанных на Express. Он обрабатывает запросы до их попадания в соответствующие обработчики. |
| СУБД | **Microsoft SQL Server** — система управления реляционными базами данных (РСУБД), разработанная корпорацией Microsoft. |

# **Интерфейсы системы**

|  |  |
| --- | --- |
| *Окно выбора входа или регистрации* | *Вход в систему*  **Изображение выглядит как текст  Автоматически созданное описание**  *Регистрация клиента*  **Изображение выглядит как текст  Автоматически созданное описание** |
| *Главная страница администратора* | |
| *Раздел «Заказы»* | *Страница заказов*    *Ввод заказов по статусу*  Изображение выглядит как текст  Автоматически созданное описание  *Детали заказа*  Изображение выглядит как текст  Автоматически созданное описание  *Назначение сотрудника на исполнение заказа*  Изображение выглядит как текст  Автоматически созданное описание  *Сотрудник назначен на исполнения заказа* |
| *Раздел «Список клиентов»* | *Вывод списка клиентов по количеству заказов* |
| *Раздел «Список блюд»* | *Вывод списка блюд в меню*    *Поиск блюда по названию*  Изображение выглядит как текст  Автоматически созданное описание  *Вывод состава блюда*    *Изменение информации о блюде*    *Добавление ингредиента блюда*  **Изображение выглядит как текст  Автоматически созданное описание**  *Добавление блюда*  **Изображение выглядит как текст  Автоматически созданное описание** |
| *Раздел «Список сотрудников»* | *Список сотрудников*    *Редактирование информации о сотруднике* |
| *Раздел «Список ингредиентов»* | *Вывод списка ингредиентов*    *Поиск ингредиента по названию*  **Изображение выглядит как текст  Автоматически созданное описание**  *Редактирование информации об ингредиенте*  **Изображение выглядит как текст  Автоматически созданное описание** |
| *Раздел «Регистрация сотрудника»*  *Изображение выглядит как текст  Автоматически созданное описание* | *Регистрация нового сотрудника* |
| *Раздел «Регистрация сотрудника»* | *Личный кабинет (редактирование логина и пароля)* |
| *Главная страница работника кухни* | |
| *Раздел «Заказы»* | *Вывод заказов, на которые назначен/был назначен сотрудник*  *Изображение выглядит как текст  Автоматически созданное описание*  *Детали заказа*    *Изменение значения статуса заказа* |
| *Раздел «Список блюд»* | *Вывод списка блюд*    *Поиск блюда по названию*    *Вывод состава блюда* |
| *Раздел «Блюда в заказах»* | *Вывод списка блюд в заказах*    *Вывод блюд в заказах с определенным статусом*    *Вывод состава блюда*    *Изменение значения флага, отвечающего за готовность блюда* |
| *Раздел «Личный кабинет»* | *Личный кабинет (редактирование логина и пароля)* |
| *Главная страница работника кухни* | |
| *Раздел «Меню»* | *Вывод списка всех блюд в меню*    *Вывод списка всех блюд в меню по популярности*    *Вывод информации о выбранном блюде* |
| *Раздел «Создание заказа»* | *Создание заказа*  *Изображение выглядит как текст  Автоматически созданное описание*  *Отмена создания заказа*  *Изображение выглядит как текст  Автоматически созданное описание*  *Добавление блюда в заказ*  *Изображение выглядит как текст  Автоматически созданное описание*  *Завершение создания заказа* |
| *Раздел «Мои заказы»* | *Вывод списка активных и завершенных заказов*  *Изображение выглядит как текст  Автоматически созданное описание*  *Вывод деталей заказа (отмена заказа)*  *Изображение выглядит как текст  Автоматически созданное описание* |
| *Раздел «Личный кабинет»* | *Личный кабинет (редактирование персональных данных)*  *Изображение выглядит как текст  Автоматически созданное описание* |

# **Заключение**

В результате выполнения курсовой работы мною была реализована многопользовательская система для работы ресторана. По ходу проектирования приложения мною были получены:

* новые теоретические знания,
* практический опыт,
* опыт работы с MSSQL и Node.js

Также хочется упомянуть о сложностях, с которыми я столкнулась по ходу проектирования. Основная трудность состояла в том, что я ранее не имела дело с Node.js, поэтому в начале разработки системы пришлось потратить немало времени на изучение функционала фреймворков этой платформы. Но мне удалось преодолеть все сложности и создать систему с удобным интерфейсом, которая отвечает всем моим требованиям.

# **Список литературы**

1. Документация MS SQL [Электронный ресурс], URL: <https://docs.microsoft.com/ru-ru/sql/?view=sql-server-ver15>

2. Документация языка программирования HTML [Электронный ресурс], URL:<https://devdocs.io/html/>

3. Подключение к SQL с помощью Node.js [Электронный ресурс], URL:

[https://docs.microsoft.com/ru-ru/sql/connect/node-js/step-3-proof-of-concept-connecting-to-sql-using-node-js?view=sql-server-ver15](https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&ved=2ahUKEwiglvTXxNz0AhWXCRAIHQT6BEgQFnoECAkQAQ&url=https%3A%2F%2Fdocs.microsoft.com%2Fru-ru%2Fsql%2Fconnect%2Fnode-js%2Fstep-3-proof-of-concept-connecting-to-sql-using-node-js&usg=AOvVaw37yQF0YNKz4pr_iJGh3csM)

4. Документация Node.js [Электронный ресурс], URL: <https://nodejs.org/en/>

5. Дейт, К. Дж. Введение в системы баз данных, 8-е издание.: Пер. с англ. — М.: Издательский дом "Вильяме", 2005.

# **Приложение. Код создания таблиц**

1. **Users** – для хранения пользователей

CREATE TABLE Users   
 (IdUser int IDENTITY(1,1) PRIMARY KEY,   
 Login varchar(64) NOT NULL,   
 Password varchar(64) NOT NULL,   
 CONSTRAINT AK\_Login UNIQUE(Login))

GO

2. **Clients** – для хранения клиентов

CREATE TABLE Clients   
 (IdClient int IDENTITY(1,1) PRIMARY KEY,   
 UserId int NOT NULL FOREIGN KEY REFERENCES Users

ON DELETE CASCADE ON UPDATE CASCADE,   
 Name varchar(64) NOT NULL,   
 Phone varchar(16) NOT NULL,   
 CONSTRAINT AK\_Phone UNIQUE(Phone))   
GO

3. **Employees** – для хранения сотрудников

CREATE TABLE Employees   
 (IdEmployee int IDENTITY(1,1) PRIMARY KEY,   
 UserId int NOT NULL FOREIGN KEY REFERENCES Users

ON DELETE CASCADE ON UPDATE CASCADE,   
 Surname varchar(64) NOT NULL,   
 Name varchar(64) NOT NULL,   
 Salary money NOT NULL,   
 Position varchar(64) NOT NULL)   
GO

4. **Dishes** – для хранения блюд

CREATE TABLE Dishes   
 (IdDish int IDENTITY(1,1) PRIMARY KEY,   
 Title varchar(64) NOT NULL,   
 CONSTRAINT AK\_DishTitle UNIQUE(Title),   
 Price money NOT NULL,   
 Weight int NOT NULL)   
GO

5. **OrderStatus** – для хранения статусов заказа

CREATE TABLE OrderStatus   
 (IdStatus int IDENTITY(1,1) PRIMARY KEY,   
 NameStatus varchar(32) NOT NULL,   
 CONSTRAINT AK\_NameStatus UNIQUE(NameStatus))   
GO

6. **Ingredients** – для хранения ингредиентов

CREATE TABLE Ingredients   
 (IdIgredient int IDENTITY(1,1) PRIMARY KEY,   
 Title varchar(64) NOT NULL,   
 CONSTRAINT AK\_Title UNIQUE(Title))   
GO   
  
7. **Orders** – для хранения заказов

CREATE TABLE Orders   
 (IdOrder int IDENTITY(1,1) PRIMARY KEY,   
 StatusId int NOT NULL FOREIGN KEY REFERENCES OrderStatus

ON DELETE CASCADE ON UPDATE CASCADE,   
 EmployeeId int FOREIGN KEY REFERENCES Employees

ON DELETE SET NULL ON UPDATE CASCADE,   
 ClientId int NOT NULL FOREIGN KEY REFERENCES Clients

ON DELETE NO ACTION ON UPDATE NO ACTION,   
 CreationTime datetime DEFAULT GETDATE(),   
 Price money)   
GO

8. **DishList** – для хранения списка блюд в заказе

CREATE TABLE DishList   
 (DishId int FOREIGN KEY REFERENCES Dishes

ON DELETE SET NULL ON UPDATE CASCADE,   
 OrderId int NOT NULL FOREIGN KEY REFERENCES Orders

ON DELETE CASCADE ON UPDATE CASCADE,   
 Number int NOT NULL,   
 IsDone bit DEFAULT 0)   
GO

9. **IngredientList** – для хранения списка ингредиентов блюда

CREATE TABLE IngredientList   
 (DishId int NOT NULL FOREIGN KEY REFERENCES Dishes

ON DELETE CASCADE ON UPDATE CASCADE,   
 IngredientId int FOREIGN KEY REFERENCES Ingredients

ON DELETE SET NULL ON UPDATE CASCADE,   
 Amount int NOT NULL,   
 Unit varchar(20) NOT NULL)   
GO