



Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА К КУРСОВОМУ ПРОЕКТУ

НА ТЕМУ:

*«Создание приложения, содержащего информацию
о сети точек питания»*

Студент ИУ7-65Б
(Группа)

(Подпись, дата) Т.М. Оберган
(И.О.Фамилия)

Руководитель курсового проекта

(Подпись, дата) Б.А. Мухамеджанов
(И.О.Фамилия)

Москва, 2020 г.

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

УТВЕРЖДАЮ
Заведующий кафедрой ИУ7
(Индекс)
И.В. Рудаков
(И.О.Фамилия)
« ____ » _____ 20 ____ г.

З А Д А Н И Е

на выполнение курсового проекта

по дисциплине Базы данных

Студент группы ИУ7-65Б

Оберган Татьяна Максимовна
(Фамилия, имя, отчество)

Тема курсового проекта Создание приложения, содержащего информацию о сети точек питания

Направленность КП (учебный, исследовательский, практический, производственный, др.)
учебный

Источник тематики (кафедра, предприятие, НИР) кафедра

График выполнения проекта: 25% к 4 нед., 50% к 7 нед., 75% к 11 нед., 100% к 14 нед.

Задание Разработать программу, предоставляющую интерфейс для получения информации о точках питания, их меню и составления списка любимых блюд на основе этой информации.

Оформление курсового проекта:

Расчетно-пояснительная записка на 20-30 листах формата А4.

Перечень графического (иллюстративного) материала (чертежи, плакаты, слайды и т.п.)

На защиту проекта должна быть предоставлена презентация, состоящая из 15-20 слайдов.

На слайдах должны быть отражены: постановка задачи, использованные методы и алгоритмы, расчетные соотношения, структура комплекса программ, интерфейс.

Дата выдачи задания «17» марта 2020г.

Руководитель курсового проекта

Б.А. Мухамеджанов
(Подпись, дата) (И.О.Фамилия)

Студент

Т.М. Оберган
(Подпись, дата) (И.О.Фамилия)

Оглавление

Введение	4
1. Аналитическая часть.....	5
1.1 Постановка задачи	5
1.2 Формализация данных	5
1.3 Типы пользователей	5
1.4 Выбор СУБД	6
1.4.1 Основные функции СУБД.....	6
1.4.2 Классификация СУБД по модели данных	7
1.5 Выводы из аналитического раздела	8
2. Конструкторская часть	9
2.1 Проектирование базы данных	9
2.2 Требования к программе.....	11
2.3 Проектирование приложения.....	11
3. Технологическая часть.....	14
3.1 Выбор и обоснование языка программирования и среды разработки.....	14
3.2 Структура и состав классов.....	14
3.3 Сведения о модулях программы.....	19
3.4 Интерфейс программы.....	19
Заключение	23
Список использованной литературы.....	24

Введение

В современном мире существует множество сетей точек питания, которые обслуживают миллионы посетителей. У каждой точки разный ассортимент. Было бы удобно следить за информацией о меню и любимых блюдах в наличии разных точек в одном приложении.

Цель данной работы – реализовать приложение, которое поможет лучше ориентироваться в меню сети столовых и буфетов.

Чтобы достигнуть поставленной цели, требуется решить следующие задачи:

- 1) формализовать задание, определить необходимый функционал;
- 2) провести анализ СУБД;
- 3) описать структуру базы данных, включая объекты, из которых она состоит;
- 4) спроектировать приложение для доступа к БД;
- 5) создать и заполнить БД;
- 6) реализовать интерфейс для доступа к БД;
- 7) разработать программное обеспечение, которое позволит пользователю получать и изменять информацию о существующих точках питания и их меню.

1. Аналитическая часть

В данном разделе будет проанализирована поставленная задача и рассмотрены различные способы ее реализации.

1.1 Постановка задачи

Необходимо разработать программу для отображения информации о сети точек питания. В частности, о местоположении, типе точки, меню. Пользователь должен иметь возможность формировать персональный список блюд для дальнейшего просмотра информации о наличии этого блюда внутри сети точек питания.

1.2 Формализация данных

База данных должна хранить информацию о:

- точке питания и ее меню;
- блюдах;
- пользователях и их избранном списке блюд.

Таблица 1.1 – категории и сведения о данных

Категория	Сведения
Питательная	Название, тип(буфет/столовая/блинная/итд), местоположение
Блюдо	Название, тип, количество калорий, цена
Меню	К какой питательной относится, блюдо, день недели в который подается, состояние блюда (в наличии/готовится/нет в наличии)
Пользователь	Логин, пароль, права доступа
Личный список избранных блюд	Владелец списка, избранные блюда

1.3 Типы пользователей

Из задачи ясно, что для создания личного списка нужна авторизация пользователей. Это делит пользователей на авторизованных и неавторизованных.

Для управления сетью питалень принято решение ввести роль модератора и администратора.

Таблица 1.2 – типы пользователей и их функционал

Тип пользователя	Функционал
Неавторизованный	Регистрация, авторизация
Авторизованный	Просмотр информации о питальнях и их меню, просмотр информации о блюде, где оно есть в наличии, редактирование списка избранных блюд.
Модератор	Просмотр информации о питальнях и их меню, просмотр информации о блюде, где оно есть в наличии, редактирование списка избранных блюд. Изменение меню сети питалень.
Администратор	Просмотр информации о питальнях и их меню, просмотр информации о блюде, где оно есть в наличии, редактирование списка избранных блюд. Изменение меню сети питалень. Изменение прав доступа пользователей.

1.4 Выбор СУБД

Система управления базами данных, сокр. СУБД — совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных[1].

1.4.1 Основные функции СУБД

Основными функциями СУБД являются:

- управление данными во внешней памяти;
- управление данными в оперативной памяти с использованием дискового кэша;
- журнализация изменений, резервное копирование и восстановление базы данных после сбоев;
- поддержка языков БД.

1.4.2 Классификация СУБД по модели данных

Модель данных — это абстрактное, самодостаточное, логическое определение объектов, операторов и прочих элементов, в совокупности составляющих абстрактную машину доступа к данным, с которой взаимодействует пользователь. Эти объекты позволяют моделировать структуру данных, а операторы — поведение данных. [2]

Существует 3 основных типа моделей организации данных:

- иерархическая;
- сетевая;
- реляционная.

В иерархической модели данных используется представление базы данных в виде древовидной структуры, состоящей из объектов различных уровней. Между объектами существуют связи, каждый объект может включать в себя несколько объектов более низкого уровня. Такие объекты находятся в отношении предка к потомку, при этом возможна ситуация, когда объект-предок имеет несколько потомков, тогда как у объекта-потомка обязателен только один предок.

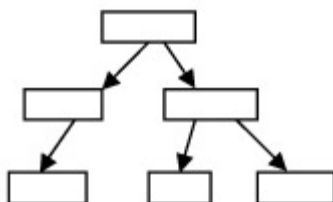


Рис. 1.1 – структура иерархической модели данных

В сетевой модели данных, в отличие от иерархической, у потомка может иметься любое число предков. Сетевая БД состоит из набора экземпляров определенного типа записи и набора экземпляров определенного типа связей между этими записями.

Главным недостатком сетевой модели данных являются жесткость и высокая сложность схемы базы данных, построенной на основе этой модели. Так как логика процедуры выбора данных зависит от физической организации этих данных, то эта модель не является полностью независимой от приложения.

Иначе говоря, если будет необходимо изменить структуру данных, то нужно будет изменять и приложение.

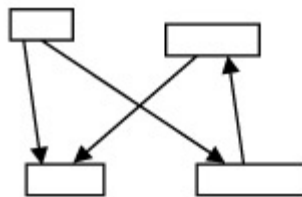


Рис. 1.2 – структура сетевой модели данных

Реляционная модель данных является совокупностью данных и состоит из набора двумерных таблиц. При табличной организации отсутствует иерархия элементов. Таблицы состоят из строк – записей и столбцов – полей. На пересечении строк и столбцов находятся конкретные значения. Для каждого поля определяется множество его значений. За счет возможности просмотра строк и столбцов в любом порядке достигается гибкость выбора подмножества элементов.

Реляционная модель является удобной и наиболее широко используемой формой представления данных.

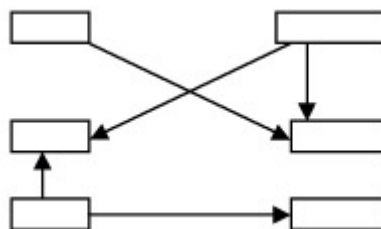


Рис. 1.3 – структура реляционной модели данных

Наиболее популярными реляционными СУБД являются Oracle, Microsoft SQL Server и PostgreSQL.

1.5 Выводы из аналитического раздела

В данном разделе была проанализирована поставленная задача и рассмотрены способы ее реализации. Также были рассмотрены разные типы СУБД. В качестве используемой в данной работе была выбрана реляционная СУБД Microsoft SQL Server.

2. Конструкторская часть

В данном разделе будет спроектированы база данных и приложение.

2.1 Проектирование базы данных

База данных должна хранить рассмотренные в таблице 1.1 данные. В соответствии с этой таблицей можно выделить следующие таблицы:

- таблица питания Eatery;
- таблица блюд Meal;
- таблица меню Menu;
- таблица пользователей User;
- таблица избранных блюд ChosenMeals.

Таблица **Eatery** должна хранить информацию о питальнях:

- id – уникальный идентификатор питальни, PK, uniqueidentifier;
- name – имя питальни, nchar(30);
- type – тип питальни (буфет / столовая / блинная), nchar(20);
- location – адрес или название здания, местоположение питальни; nchar(20);
- description – дополнительное описание питальни, nchar(100);

Таблица **Meal** должна хранить информацию о блюдах:

- id - уникальный идентификатор блюда, PK, uniqueidentifier;
- name – название блюда, nchar(20);
- type – тип блюда, nchar(20);
- kkal – количество калорий в блюде, может быть NULL, если неизвестно, int;
- cost – цена блюда, int.

Таблица **Menu** должна хранить информацию о меню:

- eateryID – идентификатор питальни, FK, uniqueidentifier;
- mealID - идентификатор блюда, FK, uniqueidentifier;
- day – номер дня с начала цикла в который блюдо должно быть в наличии, может быть NULL, если меню питальни ациклично, int;

- amount – количество порций в наличии, может быть NULL, если питальня не предоставляет информации о количестве оставшихся порций, int;
- state – состояние блюда в этой питальне: есть в наличии, готовится, нет в наличии, может быть NULL, если питальня не предоставляет информацию о блюдах в наличии, nchar(20).

Таблица **User** должна хранить информацию о пользователях:

- id - уникальный идентификатор пользователя, PK, uniqueidentifier;
- login – логин пользователя, используется для авторизации, должен быть уникальным, nchar(20);
- password – пароль пользователя, используется для авторизации, должен храниться в хешированном состоянии, nchar(64);
- permission – права доступа пользователя, может быть NULL, если права специальным образом не заданы, int.

Таблица **ChoosenMeals** должна хранить информацию о избранных пользователем блюдах:

- userID - идентификатор пользователя, FK, uniqueidentifier;
- mealID - идентификатор блюда, FK, uniqueidentifier.

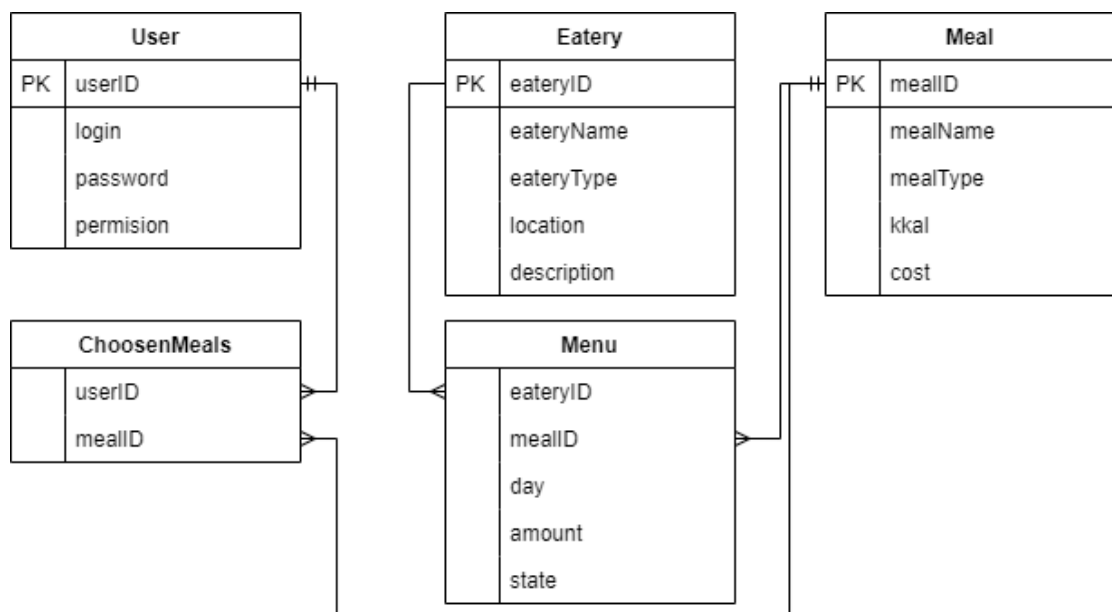


Рис. 2.1 – ER - диаграмма

2.2 Требования к программе

Для успешной реализации поставленной задачи должна предоставлять следующие возможности:

- авторизация;
- регистрация;
- вывод списка всех питалень сети;
- вывод списка питалень сети с фильтрацией по расположению или типу;
- вывод меню конкретной питальни;
- добавление блюда в избранное;
- удаление блюда из избранного;
- просмотр избранных блюд;
- вывод всех питалень, где избранное блюдо есть в наличии.

Возможности модератора:

- просмотр и изменение меню сети питалень:
 - смена статуса блюда;
 - смена количества блюда;
 - добавление нового блюда;
 - изменение имени или типа блюда.

Возможности администратора:

- изменение прав доступа пользователей.

2.3 Проектирование приложения

Программу можно разделить на 2 основные части:

- front-end – приложение с которым взаимодействует пользователь, отображение данных;
- back-end – взаимодействие с базой данных: получение, удаление и изменение данных.

Чтобы следовать принципу разделения интерфейса, каждую из частей стоит выделить в отдельный проект. Также стоит отдельно добавить проект для

данных т.к. они разделяются между back-end-ом и front-end-ом. Получим следующую структуру решения:

- AccessToDB – подключение к БД, отправка запросов;
- DataStructures – классы данных дублирующие структуру таблиц БД, нужны для ослабления связанности кода и облегченного взаимодействия между БД и приложением;
- UI – пользовательский интерфейс, через UI осуществляется взаимодействие пользователя с программой.

Проект DataStructures должен состоять из следующих компонентов.

- Eatery – класс питальни;
- Meal – класс блюда;
- User – класс пользователя;

Хранящиеся в этих классах поля дублируют соответствующие поля из БД. За исключением пароля пользователя. Пользовательский пароль не хранится в программе.

Проект AccessToDB должен состоять из следующих компонентов.

- Connector - класс коннектора, который напрямую должен осуществлять подключение к базе данных, отправку различных запросов. Все взаимодействие с БД проходит через коннектор.
- Функции работы с данными, а именно:
 - получение;
 - изменение;
 - удаление;
 - проверка данных.

Для удовлетворения требованиям к программе, описанным в разделе 2.2 необходимы следующие функции доступа к данным:

- IsLoginTaken – проверка занят ли логин, нужна для регистрации и авторизации;

- GetUserByLogin – получение информации о пользователе по логину и паролю, нужна для авторизации;
- GetAllEatery – получение списка всех питалень, нужна для отображения;
- GetEateryWhere – получение списка всех питалень, удовлетворяющим заданным параметрам, нужна для фильтрации питалень;
- GetAllMeals – получение списка всех возможных блюд в сети питалень, нужна для просмотра ассортимента блюд;
- GetAllMealsOfEatery – получение списка всех блюд конкретной питальни, нужна для получения информации о меню питальни;
- GetAvailableMealsOfEatery – получение всех блюд в наличии конкретной питальни;
- GetEateriesWhereMealIsAvailable – получение всех питалень, где конкретное блюдо есть в наличии;
- GetAllUsers – получении списка всех пользователей; нужна для повышения привилегий пользователей;
- GetEateryMenuByDay – получить меню конкретного для конкретной питальни (доступное только если у питальни есть периодичное меню);
- InsertUser – добавление нового пользователя в БД, нужна при регистрации;
- InsertMeal – добавление нового блюда в БД, нужна администраторам для нового блюда сети питалень в БД;
- InsertChoosenMeal – добавление избранного блюда в личный список;
- DeleteChoosenMeal – удаление блюда из личного списка;
- GetChoosenMeals – получение списка всех избранных блюд;
- InsetMenu – добавление нового блюда в меню питальни;
- UpdateMenu – изменение данных в меню питальни.

3. Технологическая часть

3.1 Выбор и обоснование языка программирования и среды разработки

В качестве языка программирования был выбран C# т.к.:

- этот язык имеет пакет для работы с Microsoft SQL Server (SqlClient) от Microsoft;
- данный язык программирования объектно-ориентирован, что даст в полной мере
 - использовать наследование, абстрактные классы и т.д.;
 - представлять трехмерные объекты сцены в виде объектов классов, что позволит легко организовать взаимодействие между ними, положительно влияя на читабельность кода.

В качестве среды разработки была выбрана «Visual Studio 2017» по следующим причинам:

- она бесплатна в пользовании студентами;
- она имеет множество удобств, которые облегчают процесс написания и отладки кода;
- она обеспечивает работу с Windows Forms – интерфейсом, который упрощает доступ к элементам интерфейса Microsoft Windows за счет создания обертки для существующего Win32 API в управляемом коде;
- я знакома с данной средой разработки, что сократит время изучения возможностей среды.

3.2 Структура и состав классов

В этом разделе будут рассмотрена структура и состав классов, см. рис. 3.1, рис. 3.2, рис 3.3, рис 3.4.

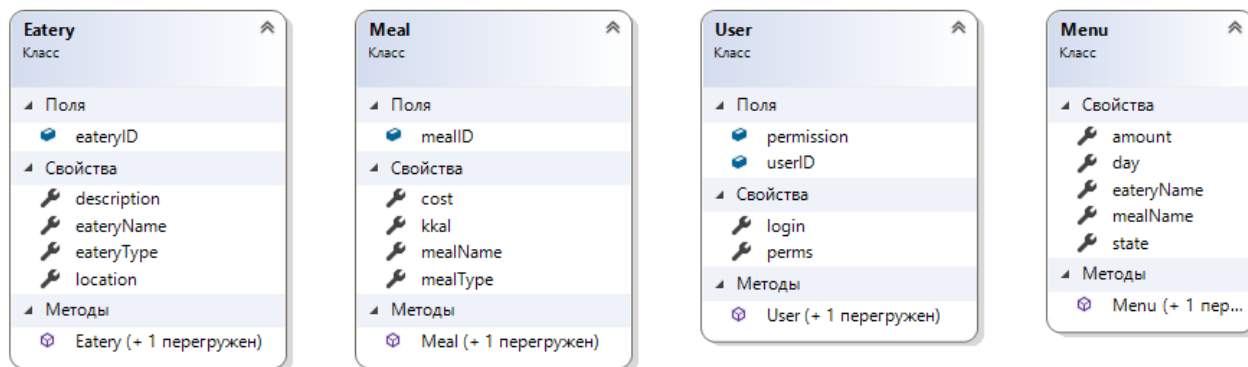


Рис. 3.1 - структура классов данных

Eatery – класс питальни.

Meal – класс блюда.

User – класс пользователя.

Meal – класс меню.

Классы данных могут принимать на вход как сами свойства, так и массив свойств:

Листинг 3.1 – конструкторы класса Eatery

```
public Eatery(string id, string name, string type = "", string loc = "", string
descr = "")
{
    eateryID = id;
    eateryName = name;
    eateryType = type;
    location = loc;
    description = descr;
}

// Создет Eatery из массива данных
// data - массив значений
public Eatery(object[] data)
{
    eateryID = data[0].ToString();
    eateryName = data[1].ToString().TrimEnd();
    if (data[2] != System.DBNull.Value)
        eateryType = data[2].ToString().TrimEnd();
    if (data[3] != System.DBNull.Value)
        location = data[3].ToString().TrimEnd();
    if (data[4] != System.DBNull.Value)
        description = data[4].ToString().TrimEnd();
}
```

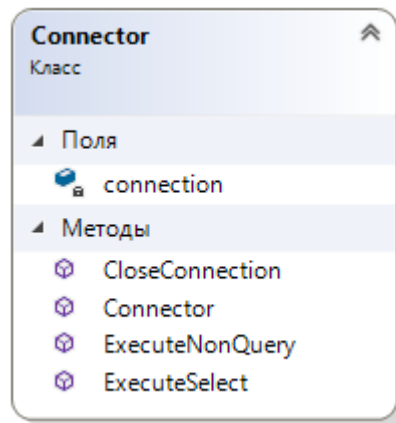


Рис. 3.2 – структура класса Connector

Connector - класс коннектора, который напрямую осуществляет подключение к базе данных, выполняет различные запросы. Все взаимодействие с БД проходит через коннектор.

Листинг 3.2 – конструктор коннектора

```

// Открывает соединение.
// По умолчанию вход в БД через текущего пользователя windows
public Connector(string connectionString =
"Server=LAPTOP;Database=DB_CP;Integrated Security=true;")
{
    connection = new SqlConnection(connectionString);
    connection.Open();
}
  
```

Листинг 3.3 – выполнение запроса, возвращающего таблицу

```

// Выполняет запрос, возвращающий список результатов
// cmdTxt - текст команды
// Возвращает список результатов (массивы значений строк)
public List<object[]> ExecuteSelect(string cmdTxt)
{
    List<object[]> res = new List<object[]>();

    using (var comand = new SqlCommand(cmdTxt, connection))
    {
        var reader = comand.ExecuteReader();
        int nCol = reader.VisibleFieldCount; // количество столбцов

        while (reader.Read()) // получаем очередную строку результата
        {
            object[] tmp = new object[nCol];
            reader.GetValues(tmp); // считать в tmp значения строк
            res.Add(tmp);
        }
        reader.Close();
    }
    return res;
}
  
```


Листинг 3.4 – выполнение инструкции

```
// Выполняет инструкцию
// cmdTxt - текст команды
public int ExecuteNonQuery(string cmdTxt)
{
    int res = 0;
    using (var comand = new SqlCommand(cmdTxt, connection))
    {
        res = comand.ExecuteNonQuery();
    }
    return res;
}
```

Листинг 3.5 – закрытие соединения с БД

```
// Закрывает соединение
public void CloseConnection()
{
    connection.Close();
}
```

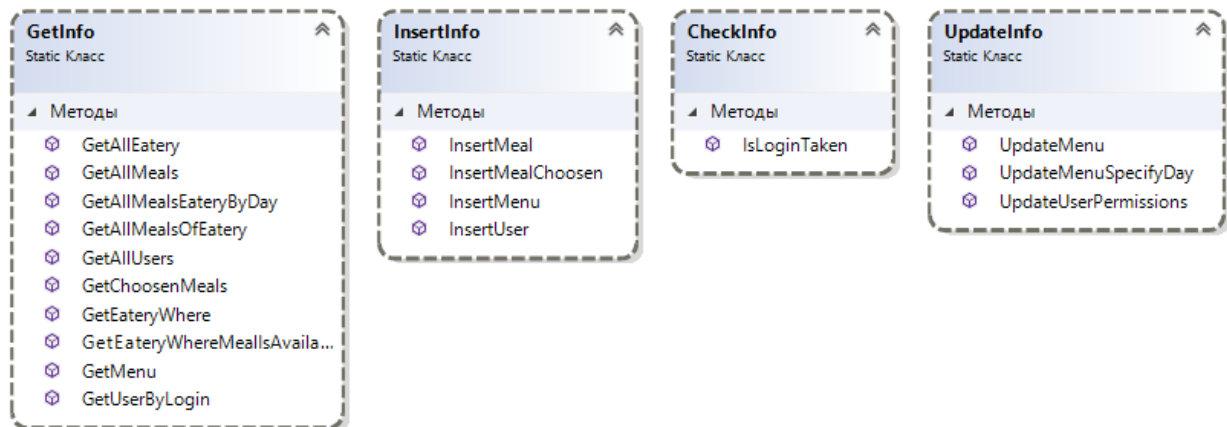


Рис. 3.3 – статические классы доступа к данным

GetInfo – статический класс получения информации из БД.

InsertInfo – статический класс добавления информации в БД.

CheckInfo – статический класс проверки информации.

UpdateInfo – статический класс обновления информации.

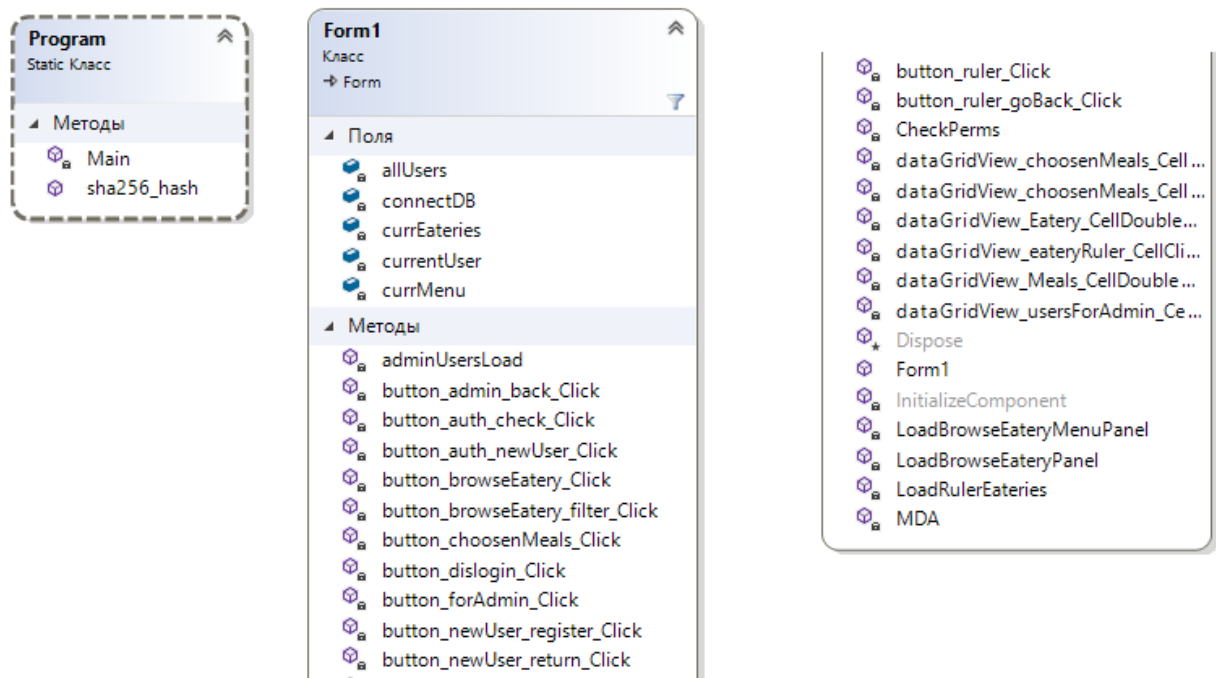


Рис. 3.4 – классы приложения

Program\Main – входная точка в программу.

Sha256_hash – функция хеширования паролей.

Form1 – класс пользовательского интерфейса.

connectDB – объект Connector.

currentUser – текущий пользователь приложения.

currEateries – список питалень, отображенный в приложении.

currMenu – список блюд, текущее меню отображенное в приложении.

allUsers – список пользователей, отображенных в приложении.

```
// Вызывается при нажатии на кнопку авторизации
private void button_auth_check_Click(object sender, EventArgs e)
{
    string login = textBox_auth_login.Text;
    string pass = textBox_auth_pass.Text;
    textBox_auth_pass.Text = ""; // сброс пароля

    bool taken = CheckInfo.IsLoginTaken(connectDB, login);
    if (taken)
    {
        User u = GetInfo.GetUserByLogin(connectDB, login, pass);
        if (u == null)
        {
            label_auth_res.Text = "Wrong password";
            label_auth_res.Visible = true;
        }
    }
    else
    {

```

```

        label_auth_res.Visible = false;
        LoadBrowseEateryPanel(u.login);
    }
    else {
        label_auth_res.Text = "No such login";
        label_auth_res.Visible = true;
    }
}

```

3.3 Сведения о модулях программы

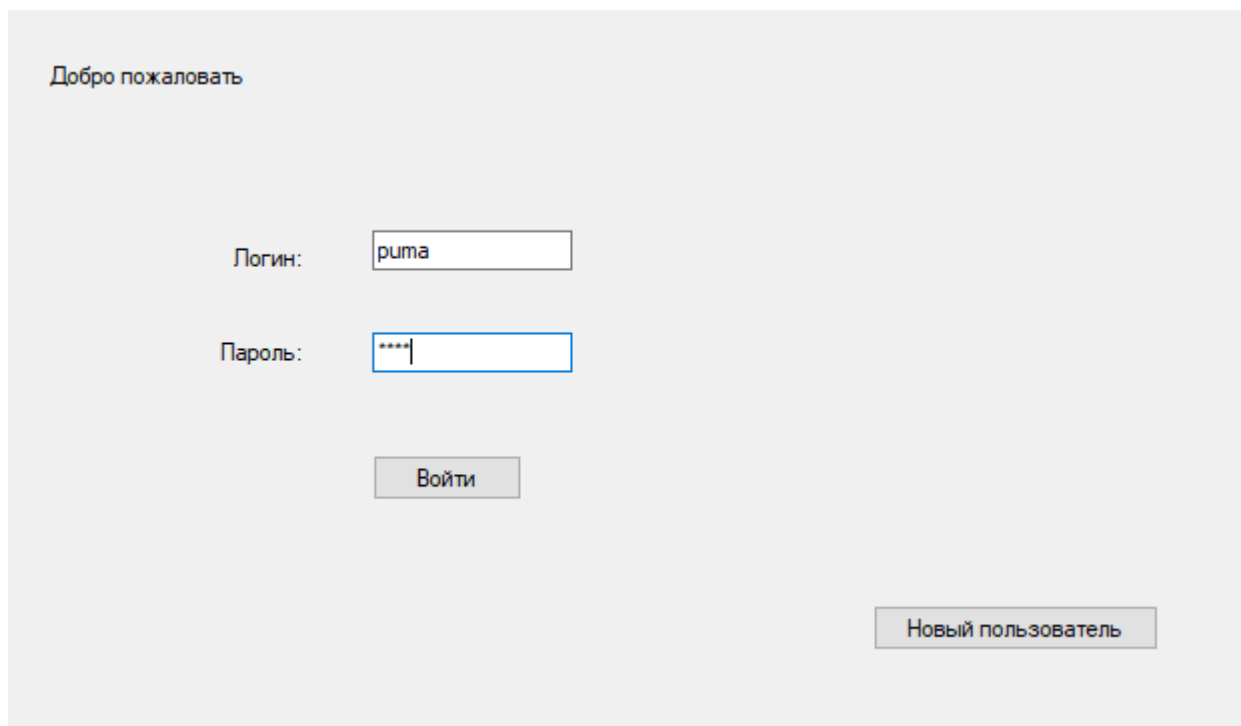
Программа состоит из 3 проектов:

- AccessToDB – подключение к БД, отправка запросов;
- DataStructures – классы данных дублирующие структуру таблиц БД, нужны для ослабления связанности кода и облегченного взаимодействия между БД и приложением;
- UI – пользовательский интерфейс, через UI осуществляется взаимодействие пользователя с программой.

Внутри проектов расположены модули, которые хранят в себе и называются идентично рассмотренным выше классам.

3.4 Интерфейс программы

Интерфейс программы разбит на страницы. Каждая страница предоставляет разную функциональность. На рис.3.5, рис.3.6, рис.3.7, рис.3.8, рис.3.9, рис.3.10 показан интерфейс различных страниц.

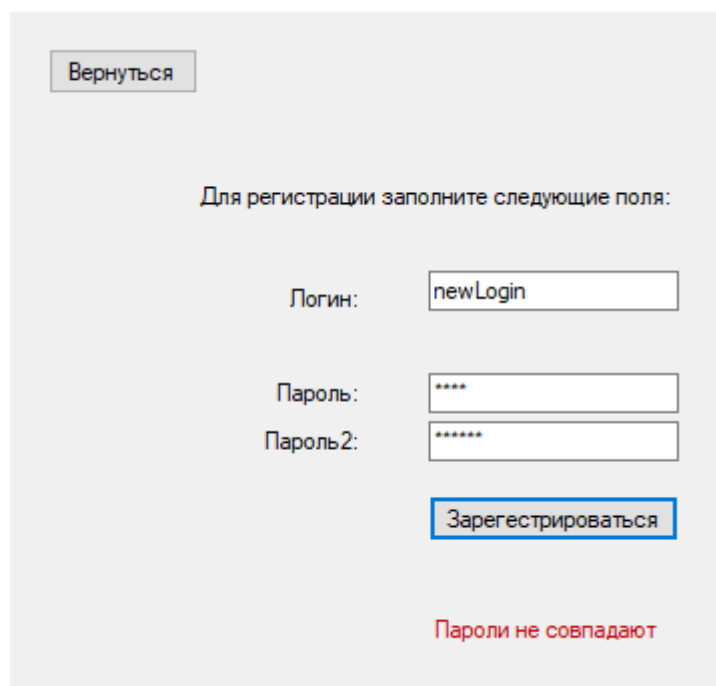


Добро пожаловать

Логин:

Пароль:

Рис. 3.5 – интерфейс страницы авторизации



Для регистрации заполните следующие поля:

Логин:

Пароль:

Пароль2:

Пароли не совпадают

Рис. 3.6 – интерфейс страницы регистрации

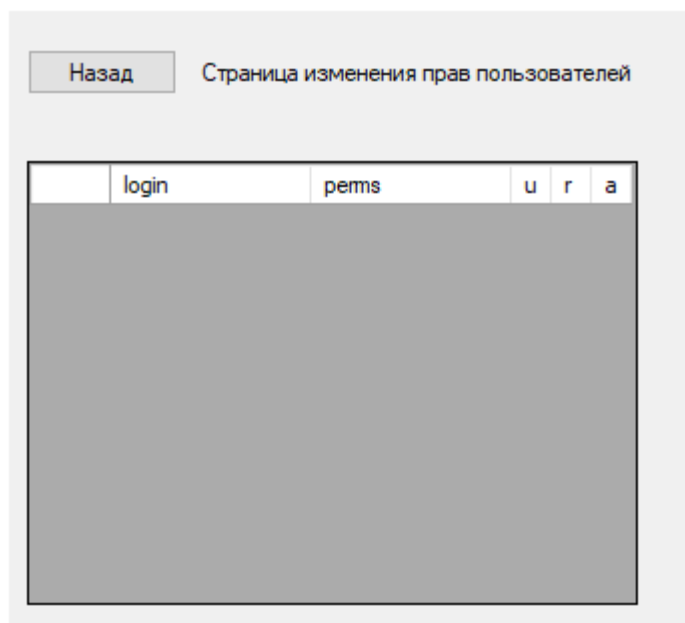


Рис. 3.10 – интерфейс страницы доступа администратора

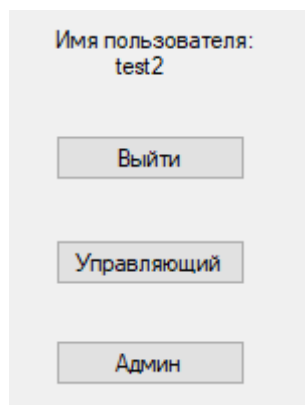


Рис. 3.11 – интерфейс переключения режимов пользователя

На рис. 3.11 показана вкладка переключения режимов пользователя, которая появляется после успешной авторизации. Кнопки «управляющий» и «админ» доступны только пользователям, имеющим соответствующие права.

Заключение

Во время выполнения курсового проекта были рассмотрены существующие виды СУБД, описана структура базы данных и приложения. Было разработано программное обеспечение, предоставляющее интерфейс к базе данных.

Программа реализована таким образом, что пользователь может получать информацию о сети питания, актуальное меню, создавать список избранных блюд и просматривать наличие этих блюд внутри сети питания; модератор сети может просматривать информацию меню и изменять его; администратор может изменять права пользователей.

В ходе выполнения поставленной задачи были изучены возможности языка C# и библиотекой WindowsForms. Получен опыт работы с Microsoft SQL Server и Microsoft SQL Server Management Studio, получены знания в области баз данных.

Список использованной литературы

1. ISO/IEC TR 10032:2003 Information technology — Reference model of data management
2. Дейт К. Дж. Введение в системы баз данных. — 8-е изд. — М.: «Вильямс», 2006.
3. Бородина А. И. Реляционная модель данных [Электронный ресурс] Режим доступа: http://www.bseu.by/it/tohod/lekcii2_3.htm, свободный (дата обращения 10.05.2020)
4. Microsoft Transact-SQL Reference(Database Engine) [Электронный ресурс] Режим доступа: <https://docs.microsoft.com/en-us/sql/t-sql/language-reference?view=sql-server-ver15>, свободный (дата обращения 10.04.2020)
5. .NET API browser System.Data.SqlClient [Электронный ресурс] Режим доступа: <https://docs.microsoft.com/en-us/dotnet/api/system.data.sqlclient?view=dotnet-plat-ext-3.1>, свободный (дата обращения 13.04.2020)