ВОРОНЕЖСКИЙ ИНСТИТУТ ВЫСОКИХ ТЕХНОЛОГИЙ – АНОО ВО

Специальность/Направление 09.03.01 Информатика и вычислительная техника

Отчет по лабораторной работе №2

по дисциплине Прикладное программирование

Тема: Разработка многооконного пользовательского клиент-серверного приложения с использованием технологии Entity Framework

Вариант 1

Выполнил: студент группы ИВТз-201

Беляева Екатерина Юрьевна

Форма обучения: заочная

Проверил: канд. техн. наук

Лавлинская Оксана Юрьевна

подпись руководителя

**Дата «\_\_» \_\_\_\_\_\_\_ 2023 г.**

**Цель работы:** разработать интерфейс пользователя для работы с файловой БД (три окна) одномодальных или одно многомодальное окно с тремя закладками. Организовать обмен данными (ввод-вывод) из файла по теме «Транспортная компания Задачи, решаемые базой данных: ведение базы данных клиентов и перевозчиков»

**Ход работы:** Создаем проект из шаблона Приложение Windows Forms (.NET Framework). Устанавливаем Entity Framework с помощью NuGet:

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

Автоматически созданное описание

Рис. 1 - Установка Entity Framework

Создаем БД с помощью обозревателя серверов:

Изображение выглядит как текст, электроника, снимок экрана, Шрифт

Автоматически созданное описание

Рис. 2 – Подключение к БД

Выполняем SQL запросы для создания нужных таблиц:

*CREATE TABLE [dbo].[Клиент]*

*(*

*[Код\_клиента] INT NOT NULL PRIMARY KEY IDENTITY,*

*[Фамилия] NVARCHAR(50) NULL,*

*[Имя] NVARCHAR(50) NULL,*

*[Отчество] NVARCHAR(50) NULL,*

*[Возраст] NVARCHAR(50) NULL,*

*[Телефон] NVARCHAR(50) NULL*

*)*

*CREATE TABLE [dbo].[Водитель]*

*(*

*[Код\_водителя] INT NOT NULL PRIMARY KEY IDENTITY,*

*[Фамилия] NVARCHAR(50) NULL,*

*[Имя] NVARCHAR(50) NULL,*

*[Отчество] NVARCHAR(50) NULL,*

*[Возраст] NVARCHAR(50) NULL,*

*[Телефон] NVARCHAR(50) NULL*

*)*

*CREATE TABLE [dbo].[Транспорт]*

*(*

*[Код\_транспорта] INT NOT NULL PRIMARY KEY IDENTITY,*

*[Марка] NVARCHAR(50) NULL,*

*[Грузоподъемность] NVARCHAR(50) NULL,*

*[Стоимость] INT NULL*

*)*

Создаем формы для взаимодействия с программой:

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

Автоматически созданное описание

Рис. 3 – Главная форма

При нажатии на кнопку «Формы» открывается новая форма:

Изображение выглядит как снимок экрана, текст, диаграмма, Прямоугольник

Автоматически созданное описание

Рис.4 – Форма «Формы»

При нажатии на кнопки «Клиент», «Водитель» и «Транспорт» открываются соответствующие формы для взаимодействия с базой данных:

Изображение выглядит как снимок экрана, текст, программное обеспечение, Значок на компьютере

Автоматически созданное описание

Рис. 5 – Форма «Клиент»

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

Автоматически созданное описание

Рис. 6 – Форма «Водитель»

Изображение выглядит как текст, снимок экрана, программное обеспечение, Значок на компьютере

Автоматически созданное описание

Рис. 7 – Форма «Транспорт»

Запустим программу и добавим данные в таблицу «Клиент»:

Изображение выглядит как текст, снимок экрана, программное обеспечение, Значок на компьютере

Автоматически созданное описание

Рисунок 8 – Пример заполнения

При нажатии на кнопку «Добавить» появляется сообщение о добавлении данных в базу данных.

Изображение выглядит как текст, снимок экрана, программное обеспечение, Значок на компьютере

Автоматически созданное описание

Рис. 9 – Кнопка «Добавить»

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

Автоматически созданное описание

Рис. 10 – Результат

Также при нажатии на кнопку «Изменить» можно редактировать запись (например, нажмем на ячейку с нужным нам кодом клиента и далее вносим в специальные поля новые данные, затем нажимаем на кнопку «Изменить» и данные в этой строке обновляются):

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

Автоматически созданное описание

Рис. 11 – Результат кнопки «Изменить»

При нажатии на кнопку «Удалить» можно удалить запись (например, нажмем на ячейку с нужным нам кодом клиента и далее нажимаем на кнопку «Удалить» и данные в этой строке удаляются):

Изображение выглядит как текст, снимок экрана, программное обеспечение, Значок на компьютере

Автоматически созданное описание

Рис. 12 - Результат кнопки «Удалить»

Итоговый код разместим на github: <https://github.com/medbedica/vivt/tree/main/apppro>

Предварительно был создан и подключен проект. Файлы были подготовлены к загрузке на персональном ПК. После чего были выполнены команды git add, git commit -m “комментарий» и git push.

**Контрольные вопросы:**

1. Цель и назначение технологии Entity framework

Entity Framework — это платформа (ORM object-relational mapping — отображение данных на реальные объекты) для .NET приложений которая позволяет взаимодействовать с СУБД с помощью сущностей (entity), то есть классов и объектов NET, а не таблиц базы данных, т. е. SQL-запросов непосредственно к базе данных. Фреймворк генерирует команды базы данных для чтения или записи данных в базу данных и предоставляет надежный механизм проверки типов для работы с классами, специфичными для домена. Конечная цель Entity Framework - сократить объем кода, необходимого для доступа к данным, улучшить масштабируемость и повысить производительность.

1. Структура компонентов фреймворка

**Модели Entity Framework**

Все таблицы базы данных определяются в Entity Framework в виде классов моделей или сущностей (entity), как правило, по принципу 1 таблица, например users, – 1 класс в NET, например, User. Такие пары называют условностями, и они определены в классе контекста данных как наборы DbSet и такой подход работает по умолчанию.

**Миграции**

В процессе разработки вполне вероятна ситуация, что класс модели Entity Framework изменился, и приходится удалять и базу данных, чтобы сохранялось соответствие модели. Но при удалении базы данных удаляются и все данные из нее. Чтобы сохранить данные при изменении модели, в Entity Framework существует функция миграции. Она позволяет последовательно применять изменения схемы к базе данных, чтобы синхронизировать ее с моделью данных. В миграции существуют операции, которые позволяют удалять, добавлять столбцы и таблицы, внешние ключи, изменять настройки столбцов, добавлять, удалять и изменять данные, и так далее.

**LINQ**

LINQ — это Language Integrated Query или Внутриязыковой запрос — это такая технология, которая представляет собой набор функций в NET, которые позволяют писать структурированные запросы к базе данных. Для работы с Entity Framework использует технологию LINQ to Entities. LINQ использует похожие на SQL выражения языка C# для получения данных из базы данных. Любая реляционная база данных работает через запросы на языке SQL, и Entity Framework выражения LINQ to Entities транслирует в запросы SQL, которые понятны для используемой базы данных.

1. Альтернативные технологии доступа к базе данных

Подключаться к БД возможно также с помощью технологии ADO.NET или LINQ to SQL.

ADO.NET предоставляет собой технологию работы с данными, которая основана на платформе .NET Framework. Эта технология представляет нам набор классов, через которые мы можем отправлять запросы к базам данных, устанавливать подключения, получать ответ от базы данных и производить ряд других операций.

LINQ to SQL представляет технологию доступа и управления реляционными данными. Данная технология позволяет составлять запросу к бд в удобной форме в с помощью операторов LINQ, которые затем трансформируются в sql-выражения. Ключевыми объектами здесь являются сущности, которые хранятся в базе данных, контекст данных и запрос LINQ.

**Вывод:** в ходе выполнения лабораторной работы мы создали приложение для взаимодействия с базой данной. Изучили технологию Entity Framework для доступа к БД.