**ДОКУМЕНТАЦИЯ НА ПРОЕКТ**

**RESTOURANT “YUMMY”**

**АВТОР:**

**Кристиан Проданов**

**МОДУЛ VII „РАЗРАБОТКА НА СОФТУЕР“ (РС)**

**ЗА 11 КЛАС**

**ПО НП „ОБУЧЕНИЕ ЗА ИТ КАРИЕРА“, ЦЕНТЪР ПЛОВДИВ**

**Място на провеждане на обучението:**

**ППМГ „Гео Милев“, гр. Стара Загора**

**Цели**

Целта ми е да създам прототип на програма за ресторант, с конзолен интерфейс, чрез който да се осъществява добавяне на храни в менюто, поръчки за вкъщи, свободни маси в ресторанта, информация за персонала и резервации за ядене в ресторанта.

**Основни етапи в реализирането на проекта**

1. Обсъждане и събиране на идеи за приложението
2. Търсене на информация от сферата на медицината
3. Обсъждане на дизайн и релации между таблиците в БД
4. Създаване на БД
5. Въвеждане на информация в БД
6. Създаване на проект във Visual Studio
7. Използване на трислойна архитектура
8. Свързване на БД с проекта
9. Конструиране на заявки към БД
10. Въвеждане на информация в БД
11. Писане на документация

**Реализация на приложението**

Проектът е качен в GitHub на линк: https://github.com/KristianProdanov/RESTOURANT.git

Създал съм локална база данни Proekt, като за целта използвах приложение SQL Server Management Studio, а за кодовата част Visual Studio. База се състои от следните таблици:

* MenuItems – съдържа полета: Id на храните, описание на храната, цена, категория спрямо ястието и дали е наличено ястието;
* Orders – съдържа полета: Id на поръчката, храна от MenuItems, количество храна, крайна цена и дата на поръчката;
* Employees – Id на член от персонала, име на човека, титла като длъжност, телефон, имайл и къде живее;
* Tables – таблица с Id, за колко местна е масата и статус на масата;
* Reservations – таблица съдържаща Id на резервацията, резервация на дадено име, за колко човека има резервация, дата за кой ден ще се проведе и последното поле е за специални изисквания;

Таблиците са свързани в релации по следния начин:Graphical user interface, application, website

Description automatically generated

Проектът е качен в GitHub на линк: https://github.com/KristianProdanov/RESTOURANT.git

Създадената база използваме в реализацията на CRUD приложение без DBC в проект на Visual Studio, с език за програмиране C#.

Структурата на проекта е следната:

* Съдържа следната трислойна архитектура:
  + Слой за данни;
  + Бизнес слой;
  + Презентационен слой;

Посочената архитектура е реализирана чрез класове, поместени в следните папки:

* Папка **Models** – реализира общ (споделен) слой, който работи с всички останали. Тя съдържа следните класове:
  + **Employees.cs** - клас, който визуализира в проекта таблица employees от базата данни;
  + **MenuItems.cs** – клас, който визуализира в проекта таблица menuItems от базата данни;
  + **Orders.cs** - клас, който визуализира в проекта таблица orders от базата данни;
  + **ProektContext.cs** – клас, който в проекта е цялата база данни вкарана в програмата;
  + **Reservation.cs** - клас, който визуализира в проекта таблица reservation от базата данни;
  + **Tables.cs** - клас, който визуализира в проекта таблица tables от базата данни;
* Папка **Data** – реализира слоя за данни и включва следните класове:
  + **Database.cs** – статичен клас за връзка с базата данни;
  + **EmployeesData.cs** – клас, в който изпълняваме заявки към БД свързани с таблица employees;
  + **ItemsData.cs** – клас, в който ще изпълняваме заявки към БД свързани с таблица menuItems;
  + **OrdersData.cs** – клас, в който ще изпълняваме заявки към БД свързани с таблица orders;
  + **ReservationsData.cs** – клас, в който ще изпълняваме заявки към БД свързани с таблица reservation;
  + **TablesData.cs** – клас, в който ще изпълняваме заявки към БД свързани с таблица tables;
* Папка **Businesses** – реализира бизнес слой, съдържащ следните класове:
  + **EmployeesBusiness.cs** – описва бизнес логиката свързана с клас Employees;
  + **MenuItemsBusiness.cs** – описва бизнес логиката свързана с клас MenuItems;
  + **OrdersBusiness.cs** – описва бизнес логиката свързана с клас Orders;
  + **ReservationsBusiness.cs** – описва бизнес логиката свързана с клас Reservations;
  + **TablesBusiness.cs** – описва бизнес логиката свързана с клас Tables;
* Папка **Presentation** – реализира презентационен слой и включва следните класове:
  + **DisplayEmployees.cs** – клас, в който реализираме конзолно управление на персонала (добавяне, актуализиране, изтриване, достъп до персонала);
  + **DisplayMenuItems.cs** – клас, в който реализираме конзолно управление на менюто с храни;
  + **DisplayOrders.cs** – клас, в който реализираме конзолно управление на поръчките от клиенти;
  + **DisplayReservations.cs** – клас, в който реализираме конзолно управление на резервациите за деня;
  + **DisplayTables.cs** – клас, в който реализираме конзолно управление на масите и местата в ресторанта;
  + **Display.cs** – клас, в който съчетаваме конзолните управления на учениците и оценките им (съчетаваме класовете DisplayEmployees, DisplayMenuItems, DisplayOrders, DisplayReservations и DisplayTables);
* **Program.cs** – Място, в което създаваме интерфейс на приложението, чрез обект от класа Display.

**Заключение**

Създадох приложение с лесен конзолен интерфейс, чрез което да могат да се реализират основните функции: добавяне на храни, поръчки, резервации, персонал и маси в ресторант, като с това се опитах да му придам завършеност и функционалност. Бих направил за напред програмата реализирана в графичен интерфейс и да добавя повече база данни.