**Order Manager Application / Aplikacja do zarządzania zamówieniami**

1. **Wytyczne dotyczące obsługi i instalacji**

Order Manager to aplikacja napisana w C# z użyciem Windows Forms, która służy do zarządzania zamówieniami oraz płatnościami w firmie. Aplikacja korzysta z pięciu tabel w bazie danych zarządzanej przy użyciu Microsoft SQL Server: Orders, Payments, OrderDetails, Customers i Products. Aplikacja umożliwia zarządzanie danymi w czterech głównych widokach.

W widoku Klienci użytkownicy mogą dodawać, edytować i usuwać informacje o klientach, co pozwala na utrzymanie aktualnych danych kontaktowych i adresowych.

Widok Produkty pozwala na zarządzanie asortymentem poprzez dodawanie nowych produktów, edytowanie informacji o istniejących produktach oraz usuwanie tych, które są już nieaktualne.

Widok Zamówienia umożliwia tworzenie nowych zamówień, dodawanie pozycji do zamówień, usuwanie zamówień oraz ich wysyłanie.

W widoku Płatności można tworzyć nowe płatności, wycofywać je w razie potrzeby oraz opłacać zrealizowane zamówienia, co ułatwia zarządzanie finansami i monitorowanie stanu płatności.

Wymagania:

1. VisualStudio 2022
2. .NET SDK 8.0
3. SQL Server 2019
4. SQL Server Management Studio (SSMS)

Instalacja:

1. Wczytaj skrypt scriptCreatingDB.sql aplikacji z folderu projektu OrderManagerApp\DataAccess\Data do SSMS.
2. Skonfiguruj połączenie z bazą danych w pliku konfiguracyjnym App.config w projekcie OrderManagerApp (OrderManagerApp\OrderManagerApp\App.config). Należy w <connectionString> podmienić: data source = „nazwaServera” i database = „nazwaBazy’.
3. Skompiluj projekt i uruchom aplikację.
4. **Diagram bazy danych**

Obraz zawierający tekst, diagram, zrzut ekranu, Plan

Opis wygenerowany automatycznie

**∞**

**∞**

**1**

**1**

**1**

**1**

**1**

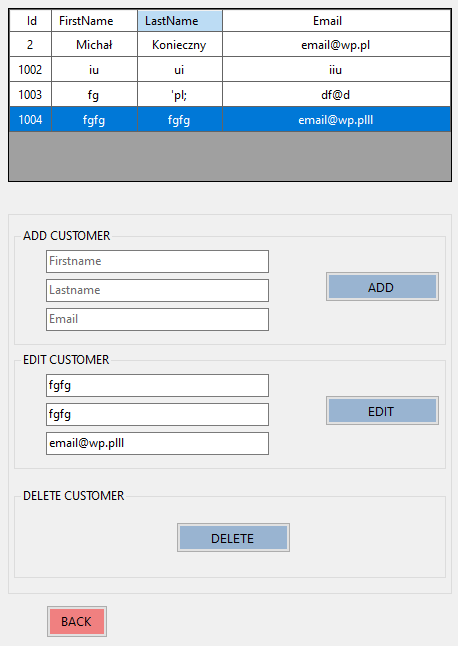
**1**

1. **Przewodnik po aplikacji**
2. **MainForm** to główny widok aplikacji, który służy jako centralny punkt nawigacyjny. Użytkownik może wybierać spośród czterech głównych funkcji aplikacji: zarządzania klientami, produktami, zamówieniami i płatnościami. Każdy z tych obszarów jest reprezentowany przez odpowiednio oznaczone przyciski.

Obraz zawierający tekst, zrzut ekranu, Czcionka, numer

Opis wygenerowany automatycznie

1. **CustomersForm** to widok dedykowany zarządzaniu danymi klientów. Umożliwia użytkownikowi dodawanie nowych klientów poprzez wprowadzenie imienia, nazwiska oraz adresu e-mail, a następnie kliknięcie przycisku "ADD". Widok ten pozwala także na edytowanie danych istniejących klientów, poprzez wprowadzenie zaktualizowanych informacji w odpowiednie pola i kliknięcie przycisku "EDIT". Dodatkowo, widok zawiera funkcję usuwania klientów za pomocą przycisku "DELETE". Na górze formularza znajduje się miejsce na wyświetlanie listy klientów. Przycisk "BACK" umożliwia powrót do głównego widoku aplikacji.



1. **ProductsForm** to widok w aplikacji Order Manager przeznaczony do zarządzania produktami. Umożliwia użytkownikowi dodawanie nowych produktów poprzez wprowadzenie odpowiednich informacji o produkcie, a następnie kliknięcie przycisku "ADD". Widok ten pozwala także na edytowanie danych istniejących produktów poprzez wprowadzenie zaktualizowanych informacji i kliknięcie przycisku "EDIT". Dodatkowo, użytkownik może usuwać produkty z bazy danych za pomocą przycisku "DELETE".

Obraz zawierający tekst, zrzut ekranu, numer, Czcionka

Opis wygenerowany automatycznie

1. **OrderForm** to widok, który umożliwia zarządzanie zamówieniami. Użytkownik może dodawać produkty do zamówienia, określając ich ilość. Formularz oferuje także opcję usunięcia całego zamówienia za pomocą odpowiedniego przycisku "DELETE". Po zakończeniu kompletowania zamówienia, użytkownik może wysłać zamówienie klikając przycisk "SEND". Dodatkowo, widok zawiera funkcję zamówienia dostawy.

Obraz zawierający tekst, zrzut ekranu, wyświetlacz, oprogramowanie

Opis wygenerowany automatycznie

1. **PaymentsForm** to widok w aplikacji przeznaczony do zarządzania płatnościami. Umożliwia użytkownikowi generowanie nowych płatności dla zamówień oraz opłacanie istniejących płatności poprzez wprowadzenie odpowiedniej kwoty w polu NumericUpDown i kliknięcie przycisku "PAY". Dodatkowo, użytkownik może wycofać płatność, korzystając z przycisku "WITHDRAW".

Obraz zawierający tekst, zrzut ekranu, numer, wyświetlacz

Opis wygenerowany automatycznie

1. **Opis wykorzystanych komponentów bazodanowych**

**DbConnectionFactory**: Jest to klasa pomocnicza, która zapewnia mechanizm tworzenia połączenia z bazą danych. Jest używana w całej wszystkich repozytoriach w projekcie DataAccess do utworzenia obiektu połączenia, który będzie używany do komunikacji z bazą danych.

Obraz zawierający tekst, Oprogramowanie multimedialne, zrzut ekranu

Opis wygenerowany automatycznie

**SqlConnection**: Ten komponent reprezentuje połączenie z bazą danych SQL Server. Jest wykorzystywany przez DbConnectionFactory do utworzenia połączenia z bazą danych przed wykonaniem każdego zapytania w projekcie.

**SqlCommand**: Jest to obiekt, który reprezentuje polecenie SQL, które ma zostać wykonane w bazie danych i znajduje się w każdej metodzie związanej z manipulacją danymi w bazie.

**SqlDataAdapter**: Ten komponent jest wykorzystywany do przesyłania danych między bazą danych a DataSet. W metodzie GetAllCustomers używasz SqlDataAdapter, aby wykonać zapytanie SQL na połączeniu i wypełnić dane klientów z bazy danych do DataSet.

**SqlDataAdapter**: Ten komponent jest wykorzystywany do przesyłania danych między bazą danych a DataSet. Wykorzystywany we wszystkich metodach w repozytoriach. Przykładowo w metodzie GetAllCustomers w CustomerRepository SqlDataAdapter użyty jest, aby wykonać zapytanie SQL na połączeniu i wypełnić dane klientów z bazy danych do DataSet.

Obraz zawierający tekst, zrzut ekranu, Czcionka, linia

Opis wygenerowany automatycznie

**DataSet**: Reprezentuje pamięć podręczną danych w formie zbioru tablic i relacji. Używany w zdecydowanej większości funkcji służących do zarządzania danymi w aplikacji. Przykładowo w metodzie GetAllCustomers w CustomerRepository użyty, aby przechować dane klientów pobrane z bazy danych przed ich przetworzeniem (screen powyżej).

**DataRow**: Reprezentuje wiersz danych w tabeli DataSet. Stosowany w zdecydowanej więkoszści metod aplikacji wykonujących polecenia bazodanowe : SELECT, UPDATE, INSERT.

Obraz zawierający tekst, Czcionka, zrzut ekranu

Opis wygenerowany automatycznie

**DataTable**: Reprezentuje kolekcję wierszy i kolumn, w której przechowywane są dane z bazy danych. Wykorzystywane w metodzie GetMissingOrderIdsInPayments w OrderRepository – tworzone są dwie tabele: dla wyników zapytania o id zamówień ze statusem „Sent” z tabeli Orders i dla wyników zapytań o id zamówień z tabeli Payments .

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

1. **Zbiór reguł biznesowych**

- Klient może zostać utworzony oraz jego dane mogą zostać zmodyfikowane tylko przy podaniu wszystkich możliwych informacji (imię, nazwisko, email),

- Klient może zostać usunięty tylko wtedy, gdy nie posiada jeszcze żadnych aktywnych zamówień,

- Płatność może zostać wygenerowana tylko dla zamówień posiadających status Sent – Wysłne (są to zamówienia z kompletną listą produktów),

- Płatność za zamówienie może zostać częściowo/całkowicie opłacona tylko wtedy gdy zamówienie jest kompletne (nie posiada

statusu Processed – Przetwarzane)

- Jednorazowo płatność nie może przewyższyć kwoty (Amount) potrzebnej do opłacenia zamówienia – nie ma możliwości nadpłaty,

- Anulować płatność można tylko dla tych zamówień, które nie zostały jeszcze dostarczone (nie posiadają statusu Delivered)