



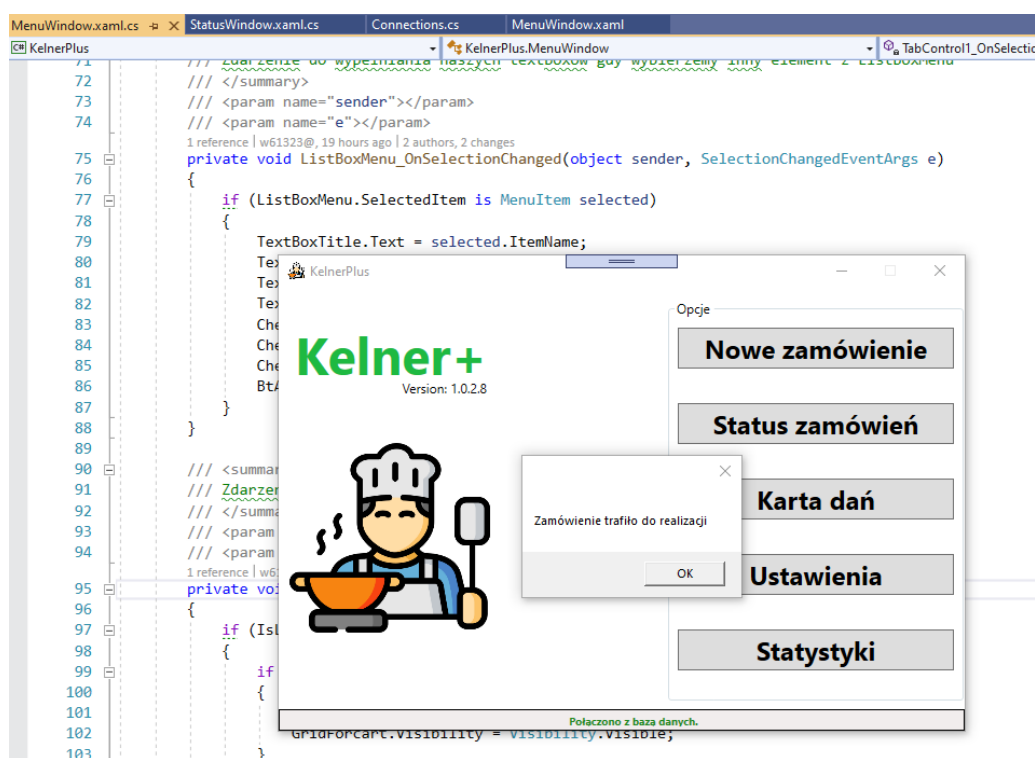
WYŻSZA SZKOŁA INFORMATYKI I ZARZĄDZANIA
z siedzibą w Rzeszowie

Dokumentacja projektu:

Aplikacja KelnerPlus

Przedmiot: Programowanie

Prowadzący: Dr Marek Jaszuk



Patryk Kawalec

nr albumu: 61311

Krzysztof Zastawny

nr albumu: 61326

Krzysztof Chudaś

nr albumu: 61078

4IIZ/2018-GP01

Grupa projektowa nr 22

Opis założeń projektu:

Celem naszego projektu jest stworzenie aplikacji która pozwalać będzie przyjmować zamówienia w restauracji przez kelnerów.

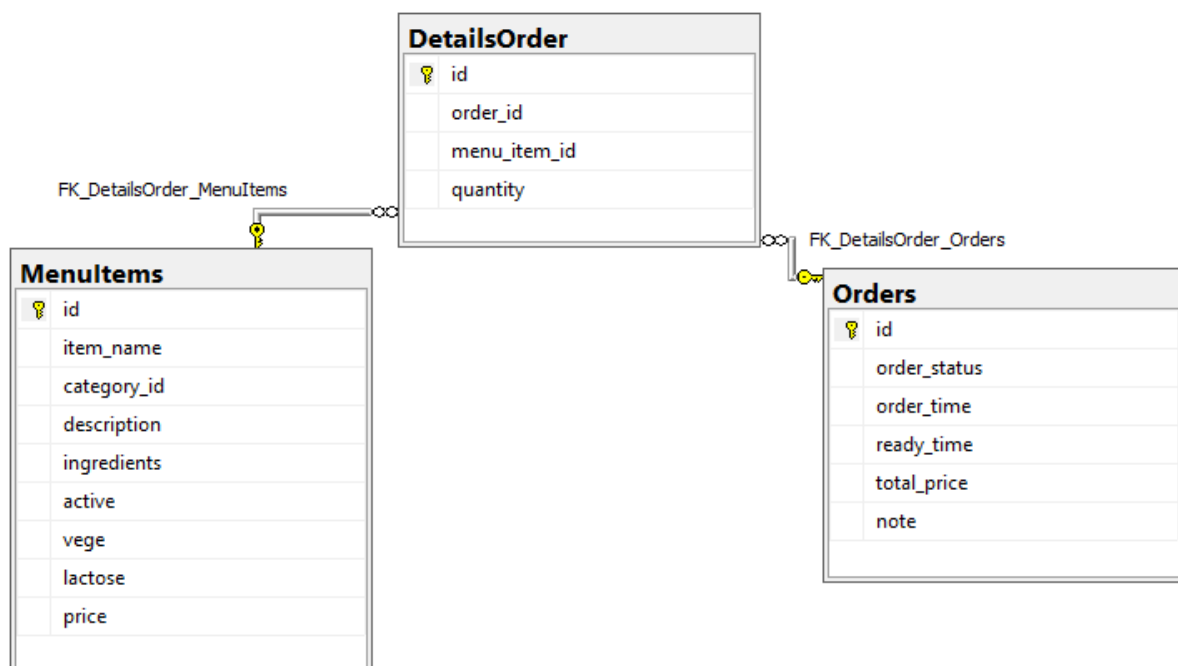
Opis techniczny projektu (struktura kodu programu):

Aplikacja powstawała w środowisku programistycznym Visual Studio 2019, dla platformy .NET Framework 4.7.2 w języku C# z elementami XAML (warstwa graficzna WPF).

Oprócz aplikacji została zaprojektowana baza danych na potrzeby programu, do przechowywania informacji o zamówieniach oraz o dostępnym menu (jadłospisie).

Konfiguracja bazy danych

Baza danych została umieszczona na serwerze lokalnym Microsoft SQL Server przy pomocy Microsoft SQL Server Management Studio (SSMS). Sama struktura jest bardzo prosta, zawiera 3 tabele (MenuItems, DetailsOrder oraz Orders) powiązane między sobą. [Skrypt schematu do pobrania na Githubie.](#)



Rysunek 1. Schemat bazy

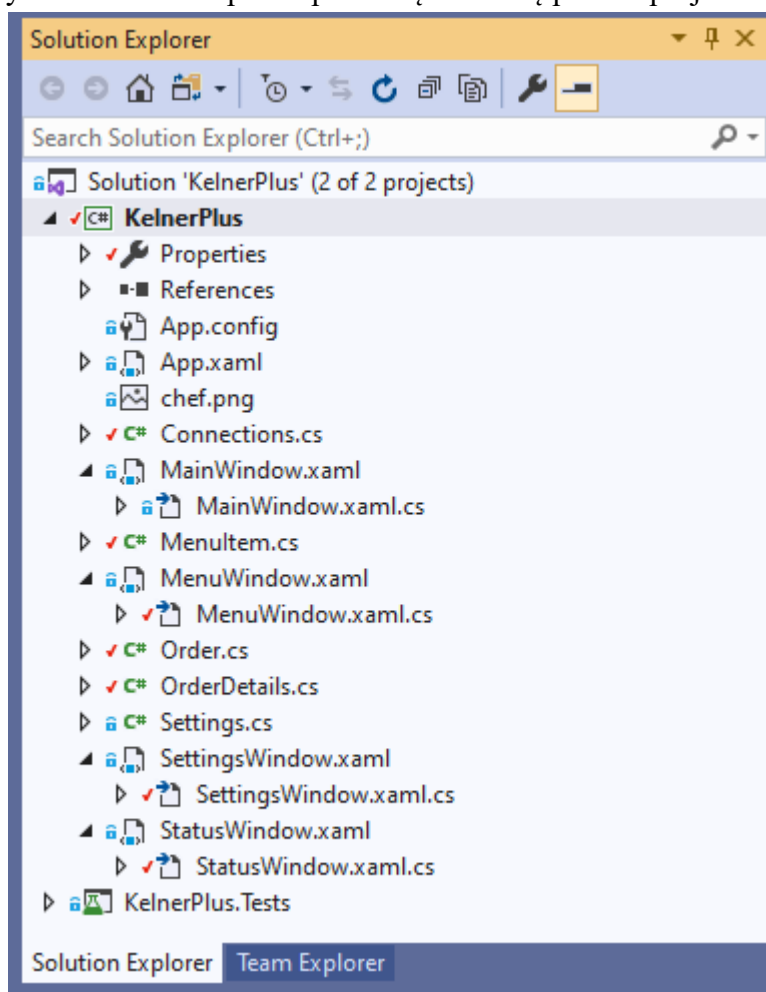
Pierwsze uruchomienie projektu

Wymagania:

- Postawiona baza danych według schematu powyższego lub pobranego z Githuba
- Visual Studio 2019
- Kod z repozytorium

Po spełnieniu powyższych wymagań przechodzimy do katalogu z kodem źródłowym i uruchamiamy poprzez plik **KelnerPlus.sln**.

Powinniśmy zobaczyć w Solution Explorer poniższą strukturę plików projektowych.

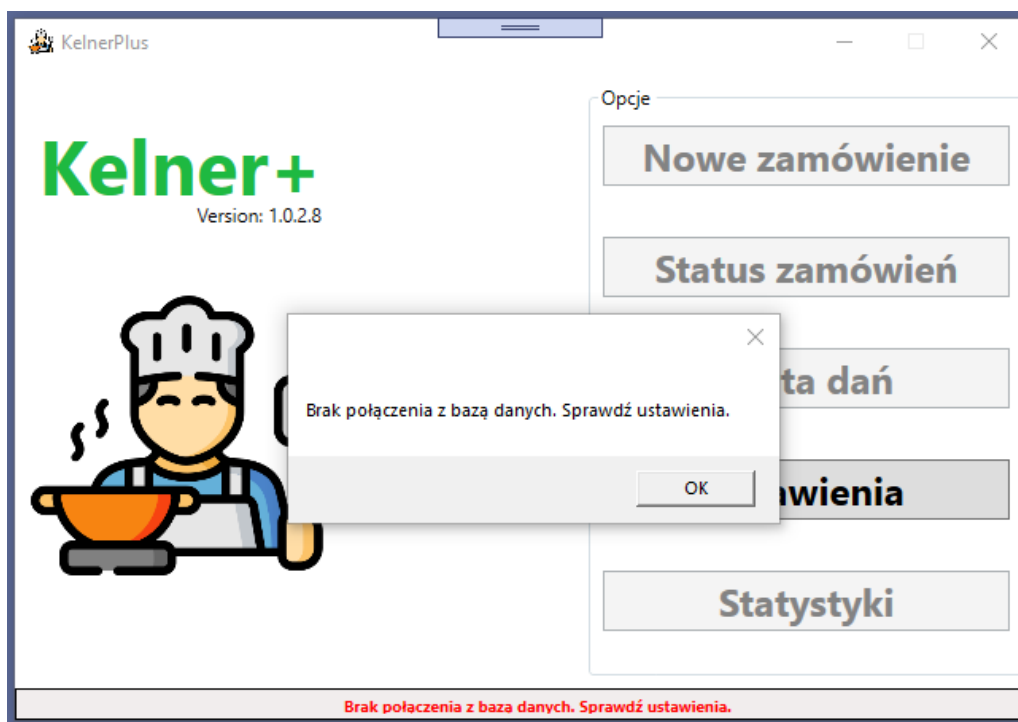


Rysunek 2. Struktura plików projektu

Po uruchomieniu przyciskiem Start lub skrótem Start without Debugging (CTRL+F5) z zakładki Debug powinna uruchomić się aplikacja.

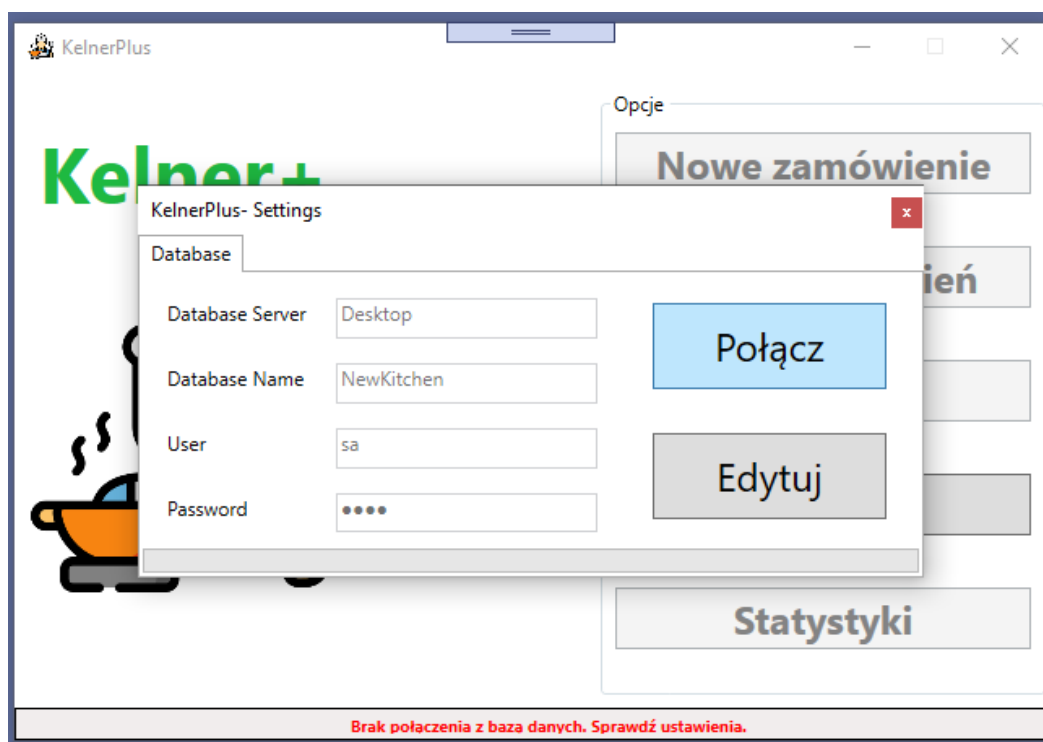
Pierwsze uruchomienie aplikacji

Jeżeli jest to nasze pierwsze uruchomienie to aplikacja wyrzuci komunikat, że nie ma połączenia z bazą danych.



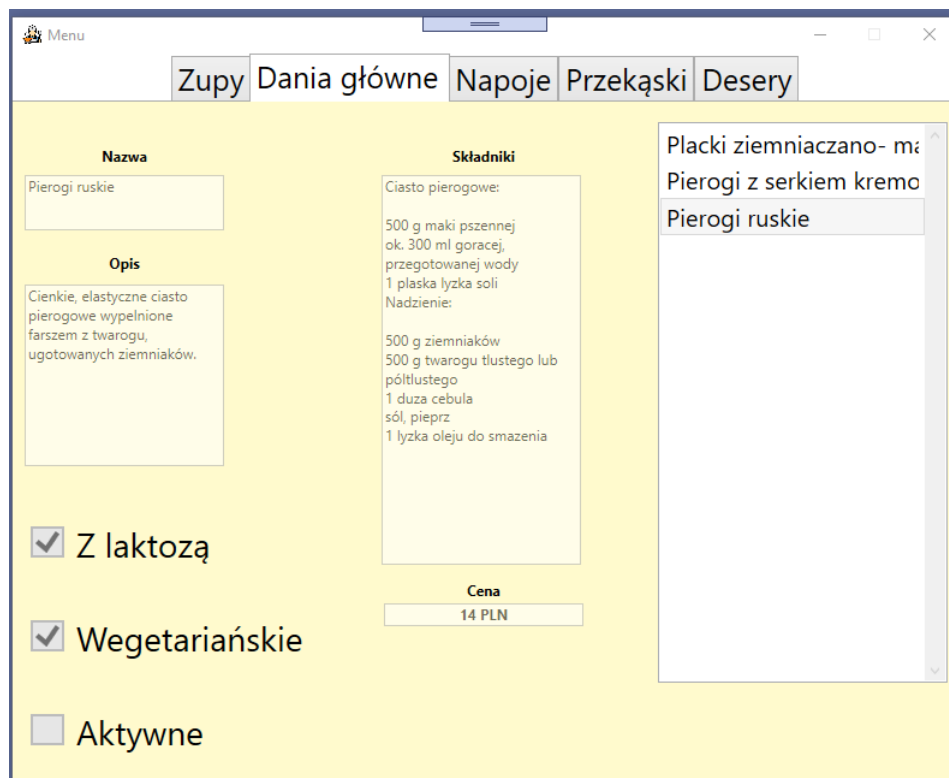
Rysunek 3. Komunikat o braku połączenia z bazą danych

W tym celu musimy przejść do zakładki Ustawienia i wprowadzić informacje o połączeniu:

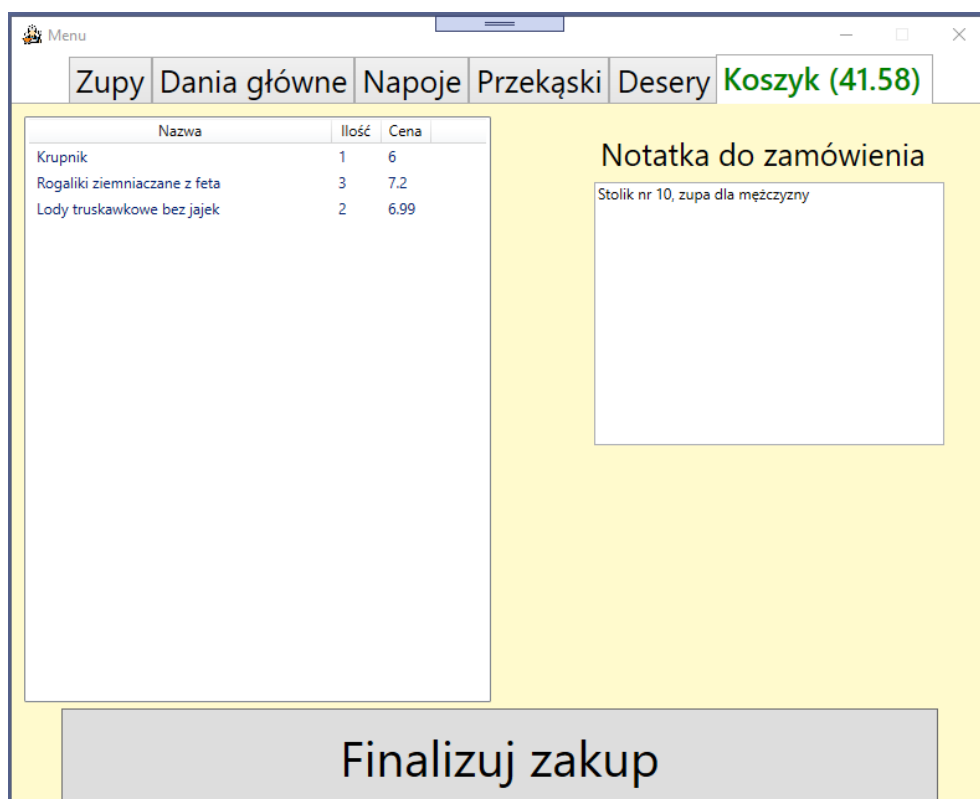


Rysunek 4. Wprowadzanie ustawień

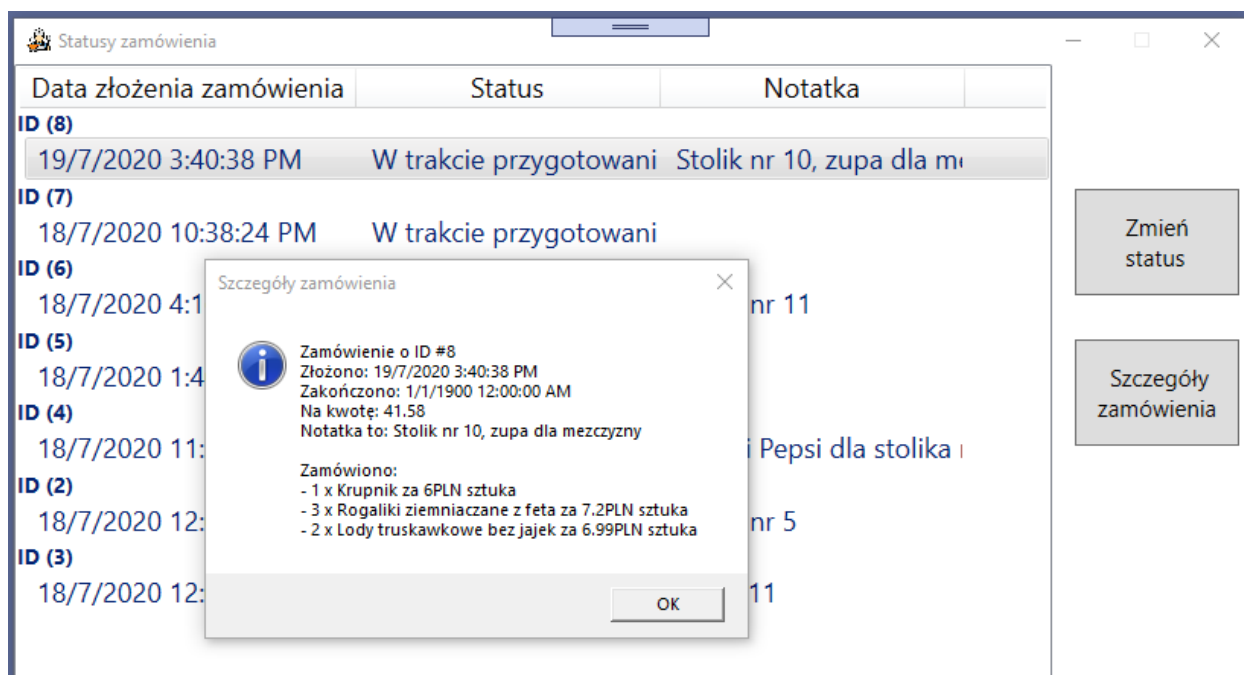
Przykładowe zrzuty ekranu z naszej aplikacji



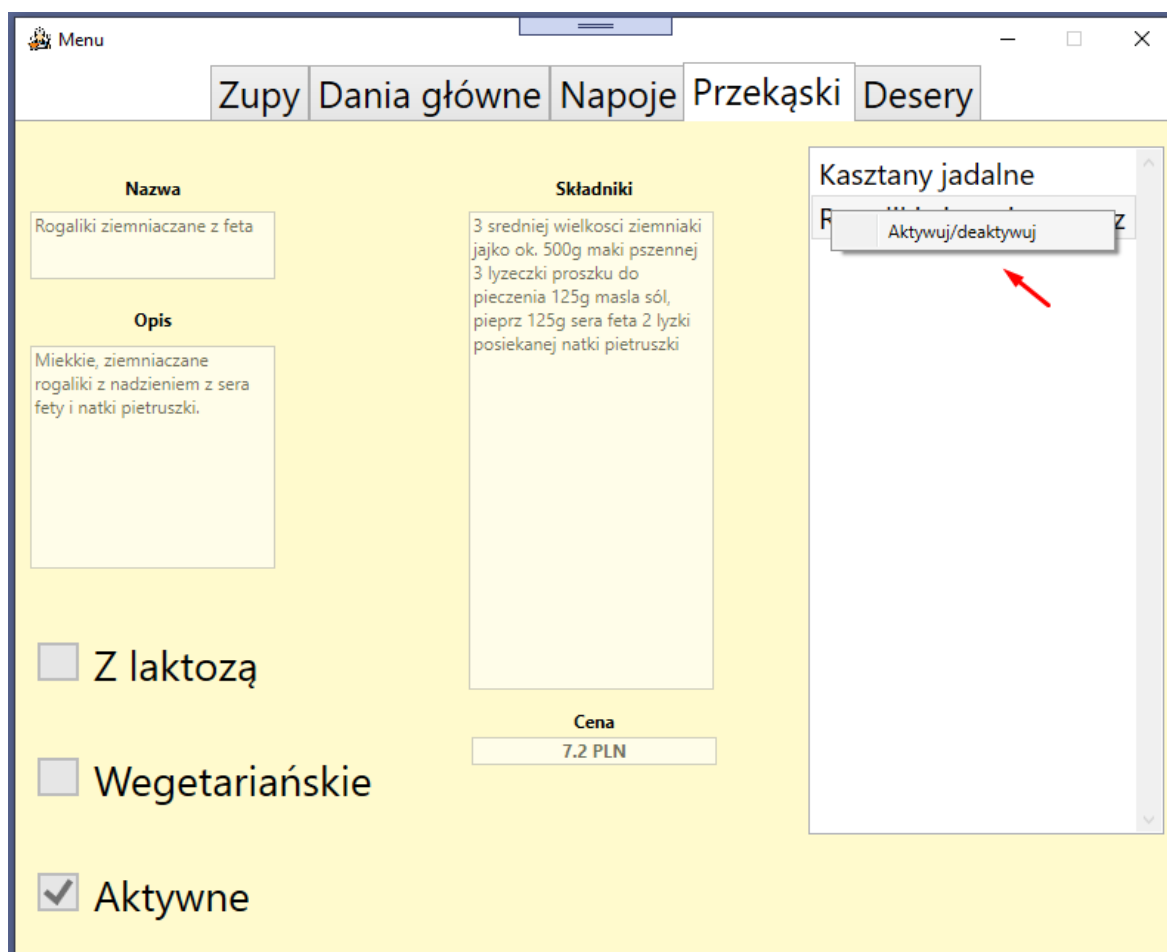
Rysunek 5. Menu dostępne w restauracji



Rysunek 6. Okno składania zamówienia- koszyk



Rysunek 8. Okno statusów zamówień- szczegóły zamówienia



Rysunek 7. Ukryte menu do aktywacji i dezaktywacji przedmiotu z Menu

Przykładowe fragmenty kodu

```
261 /// <summary>
262 /// Zmienia status zamówienia na zakończony
263 /// </summary>
264 /// <param name="myOrderId">id zamówienia, które trzeba zakończyć</param>
265 /// <param name="readyTime">czas zakończenia</param>
266 1 reference | 0 changes | 0 authors, 0 changes
267 public static void ChangeStatus(int myOrderId, string readyTime)
268 {
269     Connections.ExecuteSqlCommand(cmd: $"UPDATE Orders SET order_status=2, ready_time='{readyTime}' WHERE id= {myOrderId}");
270 }
271 }
272
```

string Connections.ExecuteSqlCommand(string cmd)

Wykonanie polecenia na bazie

Returns:

zwraca pierwszy element odpowiedzi z bazy

Rysunek 9. Metoda ChangeStatus

Powyższa metoda jest wykonywana w momencie, kiedy użytkownik zechce zaznaczyć zamówienie jako zakończone (zrealizowane). Następuje wtedy przesłanie dwóch parametrów, które następnie trafią w formie zapytania typu UPDATE do bazy. Pierwszy parametr to ID zamówienia, które użytkownik chce zakończyć, a drugi to data oraz godzina wykonania akcji. Stringu, który zwraca ExecuteSqlCommand nie musimy nigdzie przechowywać ani wyświetlać, ponieważ nie jest to SELECT.

Inaczej wygląda sprawa, kiedy wysyłanym zapytaniem SQL jest Select, wtedy możemy pobrać element zwracający string z metody ExecuteSqlCommand. Ponieważ potrzebujemy wiedzieć do dalszych czynności jakie ID otrzymało nasze zamówienie, musimy odpowiedź z SELECTa zapisać do zmiennej lastOrderId.

```
/// <summary>
/// Odpowiada za przesłanie zamówienia oraz szczegółów zamówienia do bazy
/// </summary>
/// <param name="order">Nasz obiekt order, który zostanie zapisany w bazie</param>
1 reference | w61323@, 1 day ago | 2 authors, 4 changes
private void AddOrderToDB(Order order)
{
    string insertOrder = $"INSERT INTO [dbo].[Orders]([order_status],[order_time],[ready_time],[total_price],[note])VALUES
    ({order.Status}, '{order.OrderTime}', '{order.ReadyTime}', {order.TotalPrice}, '{order.Note}')";
    Connections.ExecuteSqlCommand(insertOrder);
    var lastOrderId = Convert.ToInt32(Connections.ExecuteSqlCommand(cmd: "SELECT IDENT_CURRENT('Orders')"));
    string insertOrderDetails = "";
    foreach (OrderDetails orderDetails in order.Items)
    {
        insertOrderDetails += $"INSERT INTO [dbo].[Details]([order_id],[order_time],[ready_time],[total_price],[note])VALUES({lastOrderId},
        {orderDetails.OrderTime}, '{orderDetails.ReadyTime}', {orderDetails.TotalPrice}, '{orderDetails.Note}')";
    }
}
```

string Connections.ExecuteSqlCommand(string cmd)

Wykonanie polecenia na bazie

Returns:

zwraca pierwszy element odpowiedzi z bazy

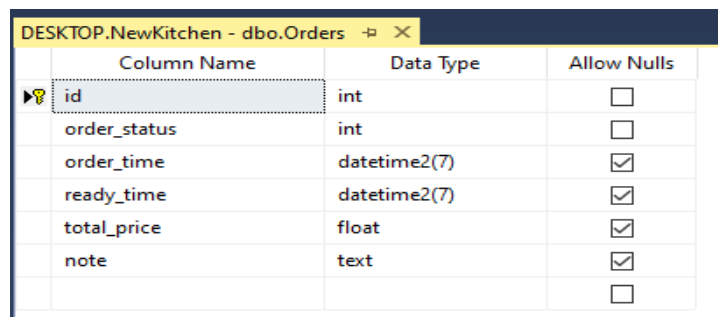
Rysunek 10. Inny sposób wykorzystania naszej metody ExecuteSqlCommand

Problemy napotkane podczas tworzenia programu

Problemy związane ze źle zaprojektowaną bazą

Podczas wykonywania projektu napotkaliśmy kilka problemów związanych ze źle zaprojektowaną bazą danych na początku, przez co kilkurotnie musieliśmy ją przeprojektowywać. Przykładem błędu, które uniemożliwił nam zapisywanie daty i czasu zamówienia było ustawienie pól `[order_time]` oraz `[ready_time]` jako **data**, następnie jako **datetime**, a finalnie doszliśmy do wniosku, że powinien być **datetime2**.

Kolejnym problem okazały się pola tekstowe, które na początku mieliśmy zadeklarowane w bazie danych jako **nvarchar(50)**. Szybko okazało się, że przy dłuższych opisach notatek lub składników dań jest to niewystarczający typ danych i musieliśmy zmienić na **text**.

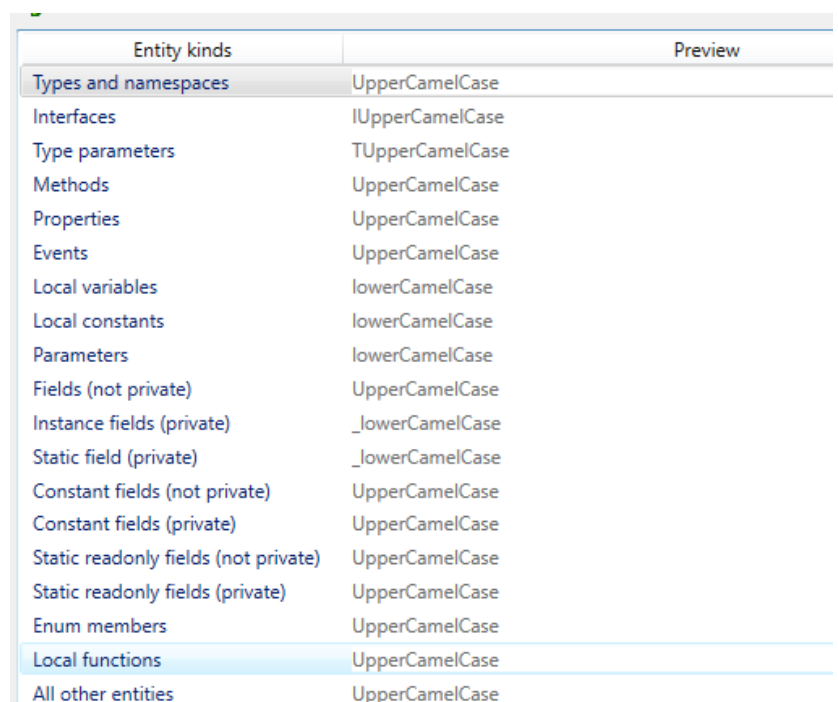


	Column Name	Data Type	Allow Nulls
▶	id	int	<input type="checkbox"/>
	order_status	int	<input type="checkbox"/>
	order_time	datetime2(7)	<input checked="" type="checkbox"/>
	ready_time	datetime2(7)	<input checked="" type="checkbox"/>
	total_price	float	<input checked="" type="checkbox"/>
	note	text	<input checked="" type="checkbox"/>
			<input type="checkbox"/>

Rysunek 11. Prawidłowe typy danych dla tabeli Orders

Problemy związane z konwencją nazewnictwa

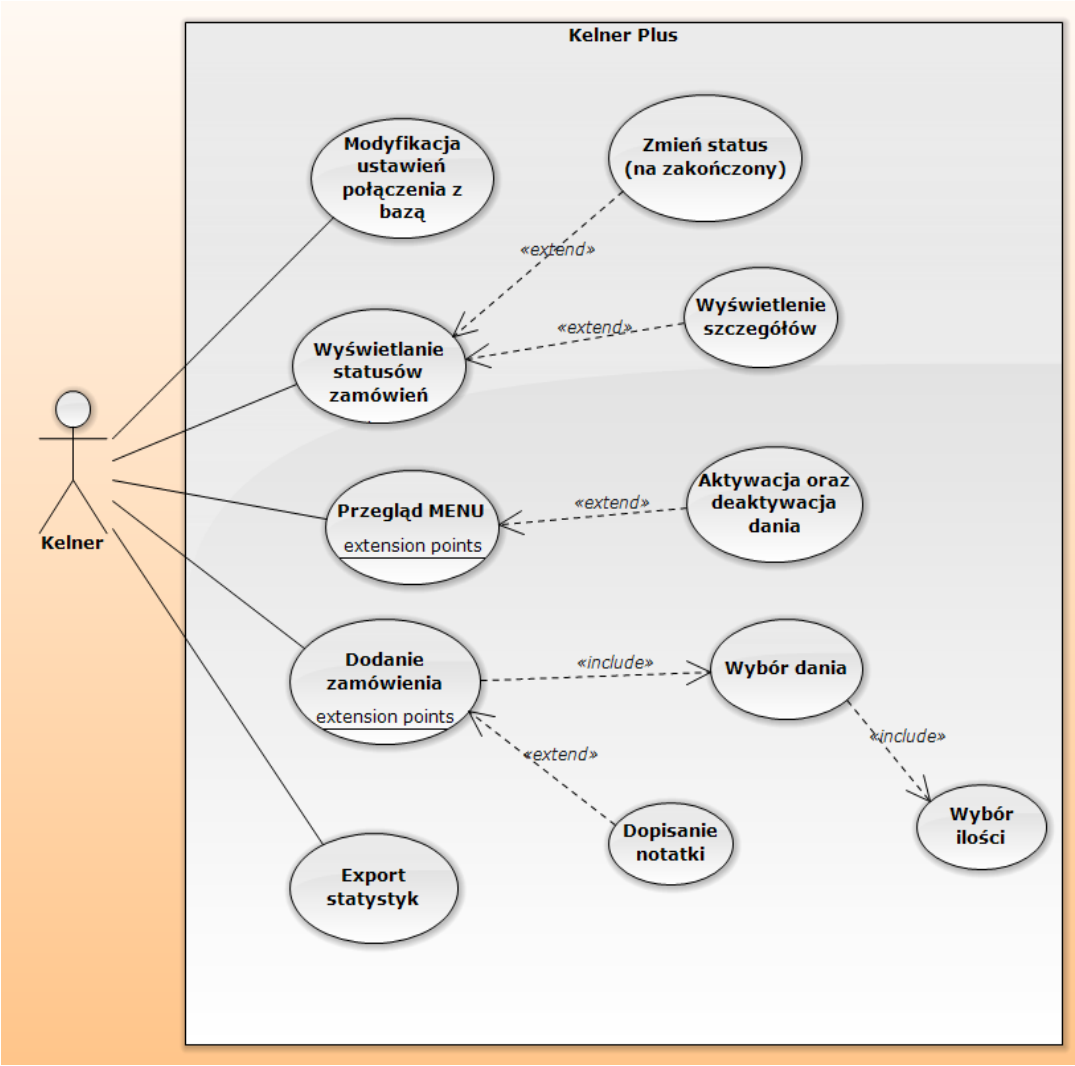
Ponieważ pracowaliśmy w grupie 3 osobowej pojawił się problem z nazewnictwem zmiennych, klas oraz metod. Dopiero na ostatnim etapie projektu, gdy robiliśmy ogólne poprawki zdecydowaliśmy się na ujednolicenie nazw i przyjęliśmy konwencję według domyślnych ustawień dodatku ReSharper



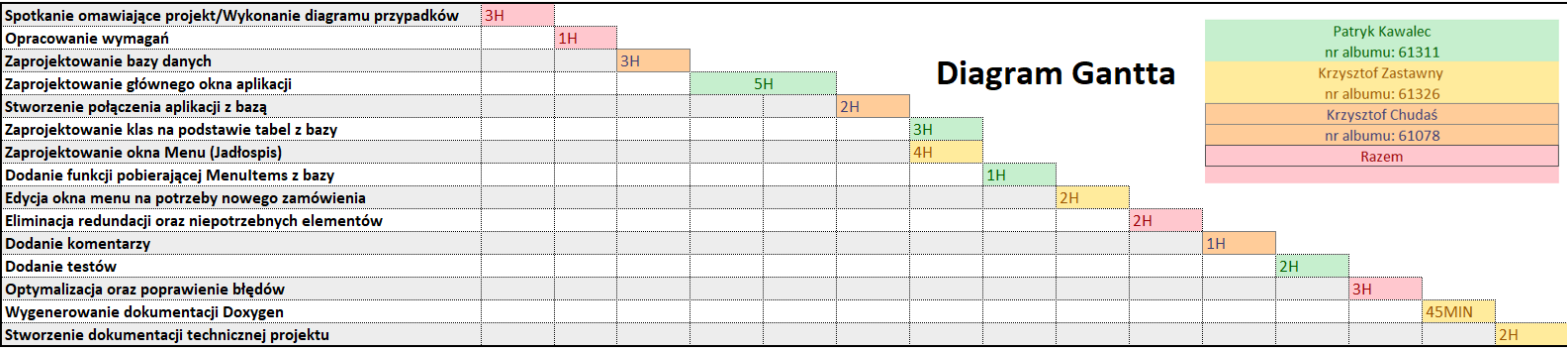
Entity kinds	Preview
Types and namespaces	UpperCamelCase
Interfaces	IUpperCamelCase
Type parameters	TUpperCamelCase
Methods	UpperCamelCase
Properties	UpperCamelCase
Events	UpperCamelCase
Local variables	lowerCamelCase
Local constants	lowerCamelCase
Parameters	lowerCamelCase
Fields (not private)	UpperCamelCase
Instance fields (private)	_lowerCamelCase
Static field (private)	_lowerCamelCase
Constant fields (not private)	UpperCamelCase
Constant fields (private)	UpperCamelCase
Static readonly fields (not private)	UpperCamelCase
Static readonly fields (private)	UpperCamelCase
Enum members	UpperCamelCase
Local functions	UpperCamelCase
All other entities	UpperCamelCase

Rysunek 12. Konwencja nazewnictwa z dodatku ReSharper

Diagramy



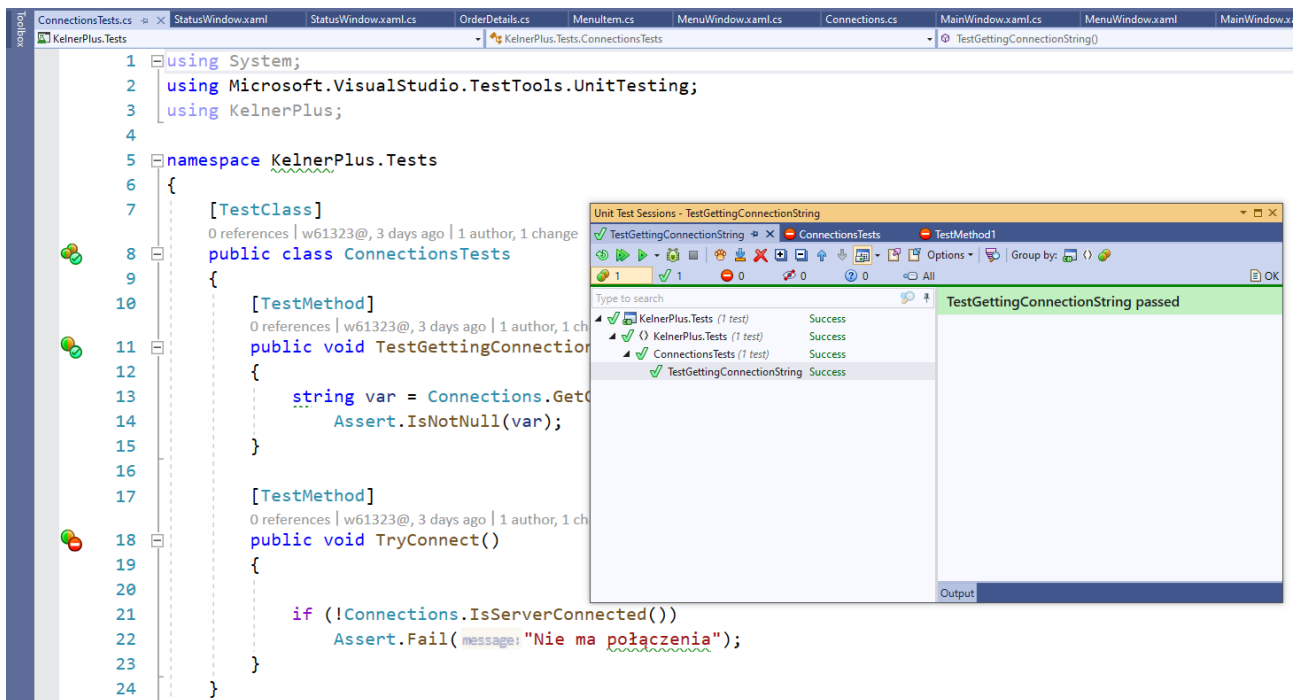
Rysunek 13. Diagram przypadków użycia



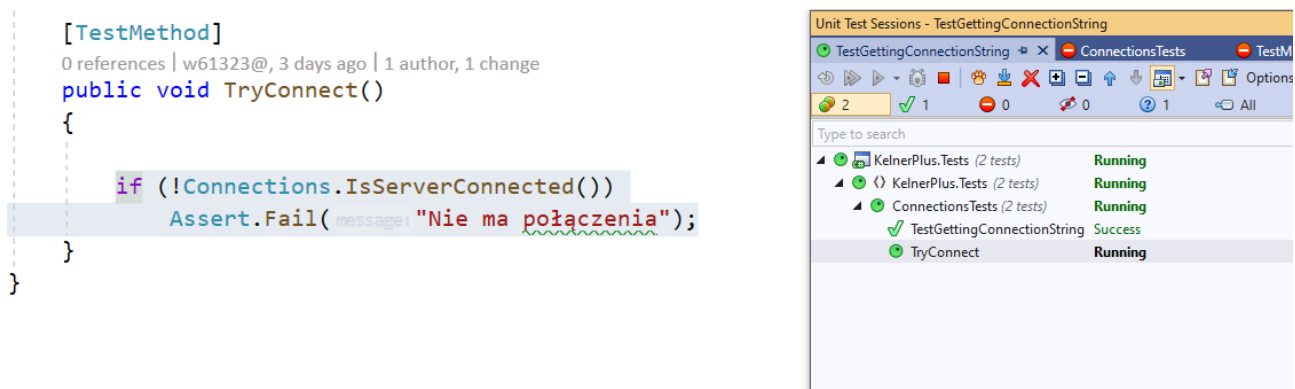
Rysunek 14. Diagram Gantt

Testy

Ponieważ nasza aplikacja oparta jest w głównej mierze na połączeniu z bazą danych, nie byliśmy w stanie wykonać testów jednostkowych.



Rysunek 15. Test jednostkowy funkcji `GetConnectionString`



Rysunek 16. Test połączenia do bazy

Materialy źródłowe:

- <https://docs.microsoft.com/pl-pl/dotnet/api/system.data.sqlclient.sqlconnection?view=dotnet-plat-ext-3.1>
- <http://csharp.net-informations.com/data-providers/csharp-sql-server-connection.htm>
- <https://www.youtube.com/watch?v=Vjldip84CXQ>
- https://www.tutorialspoint.com/ms_sql_server/ms_sql_server_create_database.htm
- http://c-sharp.ue.katowice.pl/ksiazka/c_sharp_wer2_0.pdf

Github

- <https://github.com/w61326/KelnerPlus>

Dokumentacja Doxygen

- <https://w61326.github.io/KelnerPlus/index.html>