



Katedra/Zakład:
Kierunek studiów: Fizyka Techniczna
Specjalność: Informatyka stosowana
Rodzaj studiów: stacjonarne
Imię i nazwisko: Paweł Gawroński
Numer albumu: 126412

Katedra/Zakład:
Kierunek studiów: Fizyka Techniczna
Specjalność: Informatyka stosowana
Rodzaj studiów: stacjonarne
Imię i nazwisko: Dariusz Kohlandt
Numer albumu: 123738

PROJEKT DYPLOMOWY INŻYNIERSKI

Temat projektu:

Strona internetowa wykonana w technologii ASP.NET MVC 3 z wykorzystaniem WCF

Zakres projektu:

Aplikacja umożliwia klientom przeglądanie ofert dostępnych restauracji oraz składanie zamówień. Pozwala menadżerom na zarządzanie personelem restauracji oraz treścią stron z ofertą lokali. Dzięki intuicyjnemu panelowi punktu sprzedaży możliwe jest proste w obsłudze przyjmowanie zamówień, a także zmiana statusu informująca klientów o procesie realizacji zamówienia.

Potwierdzenie przyjęcia projektu:

Opiekun projektu

.....
Tytuł, imię i nazwisko

Kierownik Katedry/Zakładu

.....
Tytuł, imię i nazwisko

Gdańsk, 02.01.2013r.

OŚWIADCZENIE

Imię i nazwisko
Wydział
Kierunek
Rodzaj studiów

Paweł Gawroński
Wydział Fizyki Technicznej i Matematyki Stosowanej
Fizyka Techniczna
stacjonarne

wyrażam/ ~~nie wyrażam~~ zgodę/y * do korzystania z mojej pracy dyplomowej:
Strona internetowa wykonana w technologii ASP.NET MVC 3 z wykorzystaniem WCF
do celów naukowych lub dydaktycznych¹.

Świadomy(a) odpowiedzialności karnej z tytułu naruszenia przepisów ustawy z dnia 4 lutego 1994r. o prawie autorskim i prawach pokrewnych (Dz. U. Nr 80, poz. 904 z 2000r ze zmianami) i konsekwencji dyscyplinarnych określonych w ustawie Prawo o szkolnictwie wyższym (Dz. U. Nr 164, poz. 1365 z 2005r.) ², a także odpowiedzialności cywilno-prawnej oświadczam, że przedkładana praca dyplomowa ~~została opracowana przeze mnie samodzielnie /~~ została opracowana w zakresie: ³ * opisanym w Podsumowaniu na stronie 140 i nie była wcześniej podstawą żadnej innej urzędowej procedury związanego z nadaniem dyplomu szkoły wyższej uczelni lub tytułów zawodowych.

Wszystkie informacje umieszczone w pracy, uzyskane ze źródeł pisanych i elektronicznych oraz inne informacje, zostały udokumentowane w wykazie literatury odpowiednimi odnośnikami zgodnie z art. 34 ustawy o prawie autorskim i prawach pokrewnych. Jednocześnie wyrażam zgodę na dołączenie tekstu pracy do bazy prac systemu antyplagiatowego.

Gdańsk, dnia 02.01.2013

.....
podpis studenta

*) niepotrzebne skreślić

OŚWIADCZENIE

Imię i nazwisko	Dariusz Kohlandt
Wydział	Wydział Fizyki Technicznej i Matematyki Stosowanej
Kierunek	Fizyka Techniczna
Rodzaj studiów	stacjonarne

wyrażam/ ~~nie wyrażam~~ zgodę/y * do korzystania z mojej pracy dyplomowej:
Strona internetowa wykonana w technologii ASP.NET MVC 3 z wykorzystaniem WCF
do celów naukowych lub dydaktycznych¹.

Świadomy(a) odpowiedzialności karnej z tytułu naruszenia przepisów ustawy z dnia 4 lutego 1994r. o prawie autorskim i prawach pokrewnych (Dz. U. Nr 80, poz. 904 z 2000r ze zmianami) i konsekwencji dyscyplinarnych określonych w ustawie Prawo o szkolnictwie wyższym (Dz. U. Nr 164, poz. 1365 z 2005r.) ², a także odpowiedzialności cywilno-prawnej oświadczam, że przedkładana praca dyplomowa ~~została opracowana przeze mnie samodzielnie /~~ została opracowana w zakresie: ³ * opisanym w Podsumowaniu na stronie 140 i nie była wcześniej podstawą żadnej innej urzędowej procedury związanego z nadaniem dyplomu szkoły wyższej uczelni lub tytułów zawodowych.
Wszystkie informacje umieszczone w pracy, uzyskane ze źródeł pisanych i elektronicznych oraz inne informacje, zostały udokumentowane w wykazie literatury odpowiednimi odnośnikami zgodnie z art. 34 ustawy o prawie autorskim i prawach pokrewnych. Jednocześnie wyrażam zgodę na dołączenie tekstu pracy do bazy prac systemu antyplagiatowego.

Gdańsk, dnia 14.01.2013

.....

podpis studenta

*) niepotrzebne skreślić

Pragniemy podziękować Panu dr inż. Patrykowi Jasikowi za merytoryczne wsparcie i pomoc podczas pracy nad projektem.

SPIS TREŚCI

SPIS RYSUNKÓW	14
SPIS FRAGMENTÓW KODU	16
WSTĘP (Dariusz Kohlandt)	17
1. NARZĘDZIA.....	18
1.1. Microsoft Visual Studio (Paweł Gawroński).....	18
1.2. phpMyAdmin (Dariusz Kohlandt).....	19
1.3. XAMPP (Dariusz Kohlandt)	19
1.4. Maszyny wirtualne (Dariusz Kohlandt)	19
2. TECHNOLOGIE	21
2.1. C# (Paweł Gawroński, Dariusz Kohlandt)	21
2.2. .NET Framework (Paweł Gawroński)	21
2.3. ASP.NET MVC (Paweł Gawroński, Dariusz Kohlandt).....	22
2.4. WCF(Paweł Gawroński).....	23
2.5. MySQL (Paweł Gawroński, Dariusz Kohlandt).....	24
2.6. JavaScript oraz jQuery (Dariusz Kohlandt).....	24
2.7. Chmury danych (Dariusz Kohlandt).....	25
2.8. HTML i CSS (Dariusz Kohlandt).....	25
3. PROJEKT APLIKACJI (Paweł Gawroński, Dariusz Kohlandt)	27
3.1. Opis.....	27
3.1.1. Opis aplikacji	27
3.1.2. Opis aktorów	27
3.2. Diagramy	28
3.2.1. Przypadków użycia	28
3.2.2. Diagramy maszyny stanowej	47
3.2.3. Diagramy czynności.....	48
3.2.4. Diagram bazy danych	61

3.2.5. Inne	63
3.3. Wymagania aplikacji	65
4. PROCES WYTWARZANIA APLIKACJI	78
4.1. Opis głównych funkcjonalności	78
4.1.1. Logowanie (Paweł Gawroński).....	78
4.1.2. Rejestracja (Paweł Gawroński).....	79
4.1.3. Odzyskiwanie hasła (Paweł Gawroński)	81
4.1.4. Zgłaszanie błędów (Paweł Gawroński)	81
4.1.5. Pomoc (Paweł Gawroński)	82
4.1.6. Wyszukiwanie restauracji (Paweł Gawroński)	83
4.1.7. Strona restauracji (Paweł Gawroński)	84
4.1.8. Edycja danych konta (Paweł Gawroński)	85
4.1.9. Zmiana hasła (Paweł Gawroński)	87
4.1.10. Komentarze (Paweł Gawroński)	88
4.1.11. Zamówienia (Paweł Gawroński).....	90
4.1.12. Dodawanie restauracji (Paweł Gawroński).....	95
4.1.13. Edycja danych restauracji (Paweł Gawroński)	97
4.1.14. Zarządzanie restauracją (Paweł Gawroński).....	98
4.1.15. POS (Dariusz Kohlandt)	103
4.1.16. Panel Administratora (Dariusz Kohlandt)	104
4.2. Budowa aplikacji	106
4.2.1. Środowisko pracy (Dariusz Kohlandt).....	106
4.2.2. Serwis WCF oparty na usłudze Windows (Paweł Gawroński)	106
4.2.3. Model dostawców (Paweł Gawroński)	112
4.2.4. Przekazywanie danych (Dariusz Kohlandt).....	118
4.2.5. Widoki (Dariusz Kohlandt).....	121
4.2.6. Autoryzacja użytkowników (Dariusz Kohlandt)	122

4.2.7. Połączenie z chmurą danych (Dariusz Kohlandt).....	123
4.2.8. Użyte komponenty (Dariusz Kohlandt)	124
4.3. Problemy w czasie wytwarzania.....	125
4.3.1. Rozsyłanie zamówień (Dariusz Kohlandt)	125
4.3.2. Lista miast (Dariusz Kohlandt)	126
4.3.3. Tworzenie menu (Paweł Gawroński).....	127
5. INSTRUKCJE URUCHOMIENIA ORAZ OBSŁUGI APLIKACJI.....	129
5.1. Instrukcja uruchomienia	129
5.1.1. Instrukcja uruchomienia usługi EresService (Paweł Gawroński).....	129
5.1.1.1. Instalacja usługi	129
5.1.1.2. Uruchamianie usługi	131
5.1.2. Import bazy danych (Paweł Gawroński).....	132
5.1.3. IIS (Dariusz Kohlandt).....	133
5.1.4. Konta dostępowe (Dariusz Kohlandt).....	135
5.2. Instrukcja obsługi (Dariusz Kohlandt)	136
5.2.1. Instrukcja dla klienta (Dariusz Kohlandt).....	136
5.2.2. Instrukcja dla menadżera (Paweł Gawroński)	136
5.2.3. Instrukcja dla restauracji (Paweł Gawroński)	136
5.2.4. Instrukcja dla administratora (Dariusz Kohlandt).....	136
6. Testy (Dariusz Kohlandt).....	137
PODSUMOWANIE (Dariusz Kohlandt).....	139
LITERATURA	142
ZAŁĄCZNIKI	144
Załącznik 1 – Manual – Klient.....	145
Wstęp	147
1. Strona główna	148
1.1. Niezalogowany użytkownik.....	149

1.1.1. Funkcjonalności Strony Głównej.....	149
1.1.2. Szukaj (Szukanie zaawansowane)	153
1.1.3. Konto.....	155
1.2. Zalogowany użytkownik.....	158
1.2.1. Edycja danych.....	159
1.2.2. Zmiana hasła	160
1.2.3. Historia zamówień	161
1.2.4. Komentarze	161
1.2.5. Koszyk	162
1.2.6. Aktualne Zamówienia	163
2. Strona Restauracji.....	164
2.1. Strona główna restauracji.....	164
2.2. Menu	165
2.3 Dowóz, Imprezy Okolicznościowe, Kontakt.....	165
2.4. Galeria.....	166
2.5. Komentarze	167
Załącznik 2 – Manual – Menadżer.....	168
Wstęp	170
1. Zakładanie konta Menagera.....	171
2. Logowanie do systemu	172
3. Panel Menadżera.....	173
3.1. Twoje Restauracje.....	173
3.1.1. Edycja danych.....	175
3.1.2. Zmiana hasła	176
3.1.3. Zarządzanie	176
3.1.3.1. Strona główna	177
3.1.3.2. Menu	178

3.1.3.3. Dowóz, Imprezy Okolicznościowe, Kontakt	179
3.1.3.4. Galeria	179
3.1.3.5. Komentarze	180
3.2. Pracownicy.....	181
3.2.1. Edycja danych.....	182
3.2.2. Zmiana hasła	182
3.3. Konto.....	183
3.4. Zgłaszanie błędów	184
Załącznik 3 – Manual – POS	185
Wstęp	187
1. Restauracja.....	188
1.1. Panel Restauracji.....	188
1.2. Aktywne zamówienia.....	189
1.3. Wszystkie zamówienia.....	190
Załącznik 4 – Manual – Admin	191
Wstęp	193
1. Administrator.....	194
1.1. Tworzenie nowego użytkownika	194
1.2. Zarządzanie użytkownikami	194
1.3. Dodawanie/usuwanie ról użytkowników	195
1.4. phpMyAdmin	196

SPIS RYSUNKÓW

Rys. 3.2.1.1. Diagram logowania do systemu (opracowanie własne).....	29
Rys. 3.2.1.2. Diagram wylogowania z systemu (opracowanie własne)	30
Rys. 3.2.1.3. Diagram wystawiania komentarzy (opracowanie własne).....	31
Rys. 3.2.1.4. Diagram wykonania zamówienia (opracowanie własne).....	32
Rys. 3.2.1.5. Diagram rejestracji w systemie (opracowanie własne).....	33
Rys. 3.2.1.6. Diagram dodawania nowej restauracji (opracowanie własne).....	34
Rys. 3.2.1.7. Diagram edycji bazy danych (opracowanie własne)	35
Rys. 3.2.1.8. Diagram wyszukiwania miasta lub restauracji (opracowanie własne).	36
Rys. 3.2.1.9. Diagram wyświetlania informacji o wybranej restauracji (opracowanie własne)	37
Rys. 3.2.1.10. Diagram wyświetlania informacji o aktualnym zamówieniu (opracowanie własne).....	38
Rys. 3.2.1.11. Diagram zgłoszania błędów (opracowanie własne)	39
Rys. 3.2.1.12. Diagram przywracania hasła (opracowanie własne).....	40
Rys. 3.2.1.13. Diagram edycji danych użytkownika (opracowanie własne)	41
Rys. 3.2.1.14. Diagram edycji danych restauracji (opracowanie własne)	42
Rys. 3.2.1.15. Diagram edycji strony restauracji (opracowanie własne)	43
Rys. 3.2.1.16. Diagram podglądu zamówienia (opracowanie własne)	44
Rys. 3.2.1.17. Diagram zmiany statusu zamówienia (opracowanie własne)	45
Rys. 3.2.1.18. Diagram podglądu wszystkich zamówień (opracowanie własne)	46
Rys. 3.2.2.1. Diagram przydzielania uprawnień po zalogowaniu (opracowanie własne)	47
Rys. 3.2.2.2. Diagram rozpoznawania uprawnień przy wylogowaniu (opracowanie własne)	47
Rys. 3.2.2.3. Diagram przedstawiający stan zamówienia (opracowanie własne)	48
Rys. 3.2.3.1. Diagram dodania nowej restauracji (opracowanie własne)	48
Rys. 3.2.3.2. Diagram czynności dla logowania (opracowanie własne)	49
Rys. 3.2.3.3. Diagram czynności dla rejestracji użytkownika (opracowanie własne)	50
Rys. 3.2.3.4. Diagram czynności dla wykonania zamówienia (opracowanie własne)	51
Rys. 3.2.3.5. Diagram czynności dla wystawienia komentarza (opracowanie własne)	52
Rys. 3.2.3.6. Diagram czynności dla wyszukania miasta (opracowanie własne)	53
Rys. 3.2.3.7. Diagram czynności dla wyszukania restauracji (opracowanie własne)	54
Rys. 3.2.3.8. Diagram czynności dla zgłoszania błędów (opracowanie własne)	54
Rys. 3.2.3.9. Diagram czynności dla przywracania hasła (opracowanie własne)	55
Rys. 3.2.3.10. Diagram czynności dla drukowania zamówienia (opracowanie własne)	56
Rys. 3.2.3.11. Diagram czynności dla edycji bazy danych (opracowanie własne)	56
Rys. 3.2.3.12. Diagram czynności dla edycji zawartości strony restauracji (opracowanie własne)	57
Rys. 3.2.3.13. Diagram czynności dla edycji danych użytkownika (opracowanie własne)	57
Rys. 3.2.3.14. Diagram czynności dla edycji danych restauracji (opracowanie własne)	58
Rys. 3.2.3.15. Diagram czynności dla wyświetlania informacji o zamówieniu (opracowanie własne)	58
Rys. 3.2.3.16. Diagram czynności dla wyświetlania wszystkich zamówień (opracowanie własne)	59
Rys. 3.2.3.17. Diagram czynności dla zmiany statusu zamówienia (opracowanie własne)	59
Rys. 3.2.3.18. Diagram czynności dla drukowania rachunku (opracowanie własne)	60
Rys. 3.2.3.19. Diagram czynności dla wylogowania (opracowanie własne)	60
Rys. 3.2.3.20. Diagram czynności dla wyświetlania informacji o restauracji (opracowanie własne)	60
Rys. 3.2.4.1. Diagram bazy danych (opracowanie własne)	61
Rys. 3.2.5.1. Diagram pakietów (opracowanie własne)	64
Rys. 3.2.5.2. Diagram komponentów (opracowanie własne)	64
Rys. 3.2.5.3. Diagram wdrożenia (opracowanie własne)	65
Rys. 4.1.2.1. Mapa dopasowanych miast oraz kodów pocztowych (opracowanie własne)	80
Rys. 4.1.4.1. Wiadomość email z treścią zgłoszenia (opracowanie własne)	82
Rys. 4.1.5.1. Podstawowy panel zarządzania dokumentem pdf (opracowanie własne)	82
Rys. 4.1.5.2. Zaawansowany panel zarządzania dokumentem pdf (opracowanie własne)	82
Rys. 4.1.6.1. Podstawowy panel wyszukiwania restauracji (opracowanie własne)	83
Rys. 4.1.6.2. Lista restauracji w danym mieście (opracowanie własne)	83
Rys. 4.1.6.3. Zaawansowane wyszukiwanie restauracji (opracowanie własne)	84
Rys. 4.1.6.4. Wyniki zaawansowanego wyszukiwania restauracji po nazwie miasta (opracowanie własne)	84
Rys. 4.1.7.1. Podgląd produktu w danej kategorii wybranej restauracji (opracowanie własne)	85
Rys. 4.1.8.1. Mapa dopasowanych miast oraz kodów pocztowych (opracowanie własne)	86
Rys. 4.1.10.1. Wystawianie komentarza (opracowanie własne)	88
Rys. 4.1.10.2. Podgląd własnego komentarza (opracowanie własne)	89

Rys. 4.1.10.3. Zgłaszanie komentarza do administracji (opracowanie własne).....	89
Rys. 4.1.11.1. Podgląd koszyka (opracowanie własne).....	90
Rys. 4.1.11.2. Realizacja zamówienia – wybór sposobu zapłaty (opracowanie własne).....	91
Rys. 4.1.11.3. Podgląd własnego komentarza (opracowanie własne)	91
Rys. 4.1.11.4. Płatność PayPal (źródło: https://www.sandbox.paypal.com)	92
Rys. 4.1.11.5. Podgląd historii zamówień (opracowanie własne)	95
Rys. 4.1.12.1. Mapa dopasowanych miast oraz kodów pocztowych (opracowanie własne).....	96
Rys. 4.1.13.1. Mapa dopasowanych miast oraz kodów pocztowych (opracowanie własne).....	98
Rys. 4.1.14.1. Edycja strony z informacjami o dowozie (opracowanie własne)	99
Rys. 4.1.14.2. Przykładowe główne logo restauracji (opracowanie własne)	99
Rys. 4.1.14.3. Usunięte zdjęcie w galerii restauracji (opracowanie własne)	101
Rys. 4.1.14.4. Podgląd kategorii (opracowanie własne).....	102
Rys. 4.1.14.5. Podgląd produktu (opracowanie własne).....	102
Rys. 4.1.14.6. Dodawanie kategorii (opracowanie własne).....	103
Rys. 4.1.14.7. Dodawanie produktu (opracowanie własne)	103
Rys. 4.1.16.1. phpMyAdmin (źródło phpMyAdmin)	105
Rys. 4.2.2.1. Usługa EresService (opracowanie własne).....	106
Rys. 4.2.2.2. Contract i podłączona referencja do usługi w projekcie (opracowanie własne)	107
Rys. 4.2.2.3. Fragment pliku Queries.resx (opracowanie własne).....	110
Rys. 4.2.2.4. Dziennik zdarzeń (opracowanie własne)	111
Rys. 4.2.2.5. Instalator usługi (opracowanie własne)	112
Rys. 4.2.3.1. Klasa CustomMembershipProvider (opracowanie własne)	113
Rys. 4.2.3.2. Klasa CustomRoleProvider (opracowanie własne)	116
Rys. 4.2.4.1. Model ProduktModel (opracowanie własne).....	119
Rys. 4.3.3.1. Dodawanie nowej kategorii (opracowanie własne)	127
Rys. 4.3.3.2. Dodawanie nowego produktu (opracowanie własne)	128
Rys. 5.1.1.1.1. Instalacja w Visual Studio	129
Rys. 5.1.1.1.3. Wybór katalogu do instalacji	130
Rys. 5.1.1.1.2. Wstęp do instalacji	130
Rys. 5.1.1.1.5. Instalowanie.....	130
Rys. 5.1.1.1.4. Potwierdzenie instalacji.....	130
Rys. 5.1.1.1.6. Zakończenie instalacji	130
Rys. 5.1.1.2.1. Uruchamianie usługi.....	131
Rys. 5.1.1.2.2. Zatrzymanie lub ponowne uruchomienie usługi.....	132
Rys. 5.1.2.1. Okno programu XAMPP	132
Rys. 5.1.2.2. Okno importu bazy danych	133
Rys. 5.1.3.1. Menedżer internetowych usług informacyjnych	134
Rys. 6.1. Debugger w kontrolerze Home (opracowanie własne).....	137
Rys. 6.2. WCF Test Client (Visual Studio 2010)	138

SPIS FRAGMENTÓW KODU

Kod 4.1.1.1. Logowanie użytkownika oraz przekierowanie do odpowiadającej roli stronie.	79
Kod 4.1.2.1. Walidacja loginu, hasła oraz adresu poczty elektronicznej.	80
Kod 4.1.2.2. Przypisanie użytkownika do roli.	81
Kod 4.1.3.1. Generowanie nowego hasła.	81
Kod 4.1.5.1. Ładowanie dokumentu pdf do atrybutu src elementu iframe.	82
Kod 4.1.8.1. Walidacja oraz wywołanie metody edytującej dane.	87
Kod 4.1.9.1. Walidacja danych.	88
Kod 4.1.10.1. Wysyłanie wiadomości email.	90
Kod 4.1.11.1. Ograniczenie ilości zamawianych produktów.	90
Kod 4.1.11.2. Sprawdzanie dostępności restauracji przed wysłaniem zamówienia.	92
Kod 4.1.11.3. Weryfikacja płatności PayPal.	94
Kod 4.1.11.4. Usuwanie zamówienia z koszyka.	95
Kod 4.1.12.1. Walidacja danych.	96
Kod 4.1.12.2. Dodawanie restauracji do systemu.	97
Kod 4.1.14.1. Zapis pliku w chmurze Dropbox.	100
Kod 4.1.14.2. Usuwanie pliku z chmury Dropbox.	100
Kod 4.1.14.3. Część odpowiedzialna za odczyt standardowego pliku z logiem restauracji.	101
Kod 4.1.15.1. Wylogowanie pracownika.	104
Kod 4.1.16.1. Tworzenie nowego użytkownika.	105
Kod 4.1.16.2. Usuwanie użytkownika.	105
Kod 4.1.16.3. Tworzenie nowej roli.	105
Kod 4.1.16.4. Usuwanie roli.	106
Kod 4.2.2.1. Fragment pliku IResService.cs.	107
Kod 4.2.2.2. Fragment pliku EresService.cs.	108
Kod 4.2.2.3. Fragment pliku Email.cs.	108
Kod 4.2.2.4. Fragment pliku Database.cs.	110
Kod 4.2.2.5. Obsługa wyjątków, zapisywanie informacji do dziennika zdarzeń.	110
Kod 4.2.2.6. Połączenie z serwisem.	111
Kod 4.2.2.7. Konfiguracja punktu końcowego serwisu WCF wraz z konfiguracją.	112
Kod 4.2.3.1. Fragment pliku CustomMembershipProvider.cs.	114
Kod 4.2.3.2. Fragment pliku Web.config.	115
Kod 4.2.3.3. Fragment pliku CustomRoleProvider.cs.	116
Kod 4.2.3.4. Fragment pliku Web.config.	117
Kod 4.2.3.5. Fragment pliku CustomMembershipUser.cs.	118
Kod 4.2.4.1. Model ProductModel.	119
Kod 4.2.4.2. Przekazanie modelu z danymi do widoku.	119
Kod 4.2.4.3. Odniesienie do danych modelu w widoku.	120
Kod 4.2.4.4. Wypełnienie JSON'a danymi.	120
Kod 4.2.4.5. Przypisanie danych z JSON'a do elementu HTML.	120
Kod 4.2.4.6. Przekazanie danych poprzez ViewData w kontrolerze.	120
Kod 4.2.4.7. Wypisanie danych z ViewData w widoku.	121
Kod 4.2.5.1. Html helper w kodzie html wyświetlający adres poczty elektronicznej z modelem.	121
Kod 4.2.5.2. Fragment klasy CSS nadającej styl elementowi w widoku.	121
Kod 4.2.5.3. Cykliczny skrypt wywołujący metodę IsOnline z kontrolera Home, która zwraca status restauracji na stronie głównej.	122
Kod 4.2.6.1. Autoryzacja użytkowników w pliku web.config dla kontrolera Find.	122
Kod 4.2.6.2. Nadpisana metoda atrybutu autoryzacji.	123
Kod 4.2.7.1. Otwieranie połącznie za pomocą biblioteki SharpBox.	123
Kod 4.2.7.2. Tworzenie nowego katalogu.	123
Kod 4.3.1.1. Zapis zamówienia do bazy danych.	125
Kod 4.3.1.2. Odczyt zamówień dla konkretnej restauracji.	125
Kod 4.3.2.1. Pobieranie współrzędnych z serwera Google.	126
Kod 5.1.2.1. Tworzenie nowego użytkownika w narzędziu phpMyAdmin.	133

WSTĘP (Dariusz Kohlandt)

„ASP.NET 4 jest wspaniałą technologią, która może być stosowana do tworzenia aplikacji sieciowych. Kiedy ASP.NET 1.0 zostało po raz pierwszy pokazane w roku 2000, wiele osób uznało tę technologię za duży, rewolucyjny krok naprzód w obszarze aplikacji sieciowych. ASP.NET 2.0 był równie rewolucyjny, a ASP.NET 4 kontynuuje ten marsz i udostępnia najlepszą ze znanych platform do tworzenia aplikacji. Technologia ASP.NET 4 została zbudowana w oparciu o podstawy przygotowane przez poprzednie wydania ASP.NET 1.0/2.0/3.5 i skupia się głównie na zwiększeniu produktywności programisty” [1].

Celem niniejszej pracy jest stworzenie witryny internetowej wykorzystując technologię ASP.NET 4 wraz ze wzorcem MVC 3. W zamierzeniu komunikację i przesył danych ma zapewnić nasza usługa systemu Windows napisana do tej aplikacji. Witryna ma spełniać funkcję portalu, który skupia w sobie restauracje, posiada możliwość obsługi klientów, menadżerów restauracji oraz pracowników tych restauracji. Portal będzie posiadał funkcję składania zamówień przez klientów, które poprzez bazę danych będą kierowane do odpowiednich restauracji. Dla menadżerów zostały przewidziane funkcje tworzenia i edycji stron własnych restauracji, włączając w to menu posiłków, galerię zdjęć oraz inne opisy najważniejszych informacji o danym lokalu. Restauracje i ich pracownicy dostaną natomiast funkcjonalności niezbędne do obsługi przychodzących zamówień. W systemie zostaną zaimplementowane płatności internetowe PayPal, za pomocą których klient będzie mógł w łatwy i szybki sposób opłacić swoje zamówienie. Zostały przewidziane również powiadomienia o statusie zamówienia oraz weryfikacje płatności dokonywanych przy użyciu systemu PayPal.

W kolejnych rozdziałach pracy zostały opisane zaimplementowane funkcjonalności, diagramy przepływu danych, opisy najważniejszych i fragmentów w aplikacji oraz opisy problemów napotkanych podczas tworzenia portalu wraz z ich rozwiązaniami.

1. NARZĘDZIA

1.1. Microsoft Visual Studio (Paweł Gawroński)

Microsoft Visual Studio jest zintegrowanym środowiskiem programistycznym opracowanym przez firmę Microsoft w 1995 roku [2]. Za jego pomocą można tworzyć aplikacje konsolowe, „okienkowe”, internetowe, usługi i aplikacje sieciowe oraz wiele innych. W skład narzędzi programistycznych Visual Studio wchodzą: Microsoft Visual C# (od wersji 7.0), Microsoft Visual C++, Microsoft Visual Basic, Microsoft Visual J# (wersje 7.0-8.0), Microsoft Visual Web Developer ASP.NET (od wersji 8.0), Microsoft Visual F# (od wersji 10) [3].

Przełomowa okazała się wersja 7.0 z 2002 roku, która wprowadziła platformę .NET Framework oraz dedykowany jej obiektowy język C#. Kolejnymi przełomowymi zmianami były komplikacja aplikacji do kodu pośredniego (Common Intermediate Language) zamiast do kodu maszynowego oraz dostosowanie języków programowania do platformy .NET.

Microsoft Visual Studio w wersji 10 z 2010 roku wykorzystuje platformę .NET Framework w wersji 4.0. Poza zmienionym graficznym interfejsem (GUI) wprowadzono nowy język programowania F# oraz mechanizm znajdowania błędów składniowych kodu w czasie rzeczywistym [4].

Wykorzystywany przez nas Microsoft Visual Studio 2010 Ultimate (oraz inne wersje oprócz Express) opiera się na licencji zamkniętego oprogramowania, dlatego powstało kilka narzędzi, głównie darmowych, które pozwalają na tworzenie oprogramowania opartego o platformę .NET. Najszybciej rozwijającym się jest zintegrowane środowisko programistyczne MonoDevelop, pozwala na pisanie aplikacji w językach takich jak C, C++, C#, Visual Basic czy w mniej popularnym języku Vala. Umożliwia pisanie aplikacji „okienkowych”, a także aplikacji internetowych opartych o ASP.NET w systemach Windows, Linux, a także Mac OSX. Główną zaletą MonoDevelop jest możliwość pracy z projektami stworzonymi w Visual Studio na innych systemach operacyjnych niż Windows [5]. Nie wykorzystaliśmy MonoDevelop z kilku powodów. Nasza aplikacja jest docelowo przeznaczona tylko do wykorzystywania w systemach operacyjnych Windows, poza tym mamy możliwość wykorzystania licencji na

Visual Studio wykupionej przez uczelnię oraz MonoDevelop nie zapewnia pełnego pokrycia bibliotek z .NET Framework.

1.2. phpMyAdmin (Dariusz Kohlandt)

phpMyAdmin jest darmowym oraz otwartoźródłowym narzędziem napisanym w języku PHP, służącym do administrowania bazami danych MySQL poprzez przeglądarkę. Potrafi obsługiwać zadania takie jak tworzenie, modyfikacja lub kasowanie baz danych, tabel, pól albo wierszy. Wykonuje zapytania SQL oraz zarządza użytkownikami i rolami dostępu [6]. Alternatywą dla tego narzędzia jest SQL Server, który współpracuje z bazami MSSQL i pozwala na wykonywanie bardziej złożonych operacji na bazach danych, poza wymienionymi wyżej. Wybraliśmy aplikację phpMyAdmin z prostego powodu, jest on obsługiwany przez każdą przeglądarkę, a co za tym idzie można go używać na prawie każdej platformie.

1.3. XAMPP (Dariusz Kohlandt)

XAMPP jest darmowym oraz otwartoźródłowym rozwiązaniem do hostowania stron internetowych. Składa się głównie z serwera HTTP Apache, bazy danych MySQL oraz z interpreterów dla skryptów napisanych w PHP i Perl'u [7]. Postanowiliśmy wykorzystać to narzędzie, ponieważ za jego pomocą mamy bezpośredni i bardzo szybki dostęp do plików bazy danych znajdujących się na maszynie wirtualnej. Alternatywą jest program WAMP, który oferuje dokładnie takie same możliwości, co XAMPP.

1.4. Maszyny wirtualne (Dariusz Kohlandt)

Maszyna wirtualna (Virtual Machine - VM) jest symulacją maszyny (abstrakcyjnej lub realnej), która jest z reguły odmienna od maszyny, na której jest uruchamiana. Maszyny wirtualne mogą bazować na specyfikacjach hipotetycznego komputera albo emulować architekturę i funkcjonowanie realnego komputera. VM jest programową implementacją maszyny, która uruchamia programy jak fizyczna maszyna. Maszyny wirtualne dzieli się na dwie główne kategorie, w zależności od ich wykorzystania i stopnia komunikacji z prawdziwą maszyną [8]. Wirtualna maszyna, która była używana podczas pisania projektu była oparta o system Windows Server 2008.

Udostępniona była ona przez uczelnię w systemie usług PLATON [9]. Na niej znajdowała się baza danych MySQL. Inne lokalne maszyny wirtualne były wykorzystane przez nas do testowania podstawowych funkcjonalności aplikacji w celu wykrycia błędów kompatybilności z innymi systemami i przeglądarkami.

2. TECHNOLOGIE

2.1. C# (Paweł Gawroński, Dariusz Kohlandt)

C# jest językiem programowania, który został wprowadzony w 2002 razem z Microsoft Visual Studio 7.0. Struktura języka C# obejmuje silne typowanie, imperatywy, deklaracje, funkcje, generyczność i obiektowo zorientowane programowanie. W zamierzeniach miał to być prosty, nowoczesny i ogólny obiektowy język programowania. Jego aktualna wersja jest oznaczona numerem 5.0 [10]. Od momentu pojawienia się tego języka na rynku dodano do jego implementacji między innymi typy generyczne, słowo kluczowe yield, typy częściowe, metody anonimowe, typy Nullable, wyrażenia lambda oraz typy domniemane i anonimowe [11].

Java jest najlepszym wyborem, jeżeli chodzi o pisanie i uruchamianie aplikacji na różnego rodzaju platformach. Natomiast w C# te same problemy, co w Java można rozwiązać przy użyciu wydajniejszego środowiska i bardziej intuicyjnych technik. C# pozwala na odnoszenie się do referencji za pomocą słów kluczowych ref i out. W Java natomiast, parametry są przekazywane przez wartość. Interfejsy w C# nie mogą zawierać składowych, natomiast w Java tak. Java nie potrafi wykryć przepelnienia na przykład podczas obliczeń matematycznych. C# może zostać zmuszony do wykrywania takich przepelnień przy użyciu bloku oznaczonego, jako checked. C# pozwala na pisanie kodu korzystającego ze wskaźników [12].

2.2. .NET Framework (Paweł Gawroński)

.NET Framework to platforma programistyczna przeznaczona do tworzenia aplikacji. Stworzona została przez firmę Microsoft w 2002 roku i jest oparta na licencji freeware. Umożliwia budowanie, wdrażanie oraz uruchamianie aplikacji desktopowych, internetowych i aplikacji na urządzenia przenośne, takich jak tablety i smartfony oraz serwisów internetowych.

Platforma .NET składa się głównie ze środowiska uruchomieniowego (Common Language Runtime) zarządzającego pamięcią oraz innymi usługami systemowymi, a także z biblioteki klas zawierającej kod wielokrotnego użytku ułatwiający pracę programistom [13].

Używany przez nas .NET Framework w wersji 4.0 z 2010 roku zawiera poprawiony system zabezpieczeń, lepsze niż w poprzednich wersjach narzędzia diagnostyczne, lepsze wsparcie systemów 64-bitowych oraz poprawioną wydajność [14].

Firma Microsoft nie udostępnia kodu źródłowego do platformy .NET, jednak jej część jest zgłoszona, jako standard ECMA (European Computer Manufacturers Association), w tym specyfikacja języka C# oraz specyfikacja CLI (Common Language Infrastructure). Dokładny opis tych dwóch elementów pozwolił na powstanie alternatywy dla .Net Framework. Jest nią środowisko Mono, czyli implementacja technologii .Net, oparta na licencji wolnego oprogramowania. W skład Mono wchodzą między innymi kompilator języka C#, VES (Virtual Execution System), a w nim kompilator Just-In-Time i garbage collector, a także implementacja bibliotek.NET. Mono współpracuje z wieloma systemami operacyjnymi w tym Linux, UNIX, Mac OS X i Windows [15]. Nie wykorzystaliśmy Mono, ponieważ używamy tylko systemu Windows, drugim powodem jest brak pełnego odwzorowania bibliotek .Net przez Mono.

2.3. ASP.NET MVC (Paweł Gawroński, Dariusz Kohlandt)

Active Server Pages to technologia stworzona przez firmę Microsoft w 1996 roku. Przeznaczona jest ona do szybkiego i łatwego tworzenia dynamicznych stron internetowych. ASP pozwala na sterowanie aplikacją po stronie serwera oraz stosowanie instrukcji VBScript oraz JScript, dzięki czemu można wykonywać operacje przed wysłaniem strony na serwer. ASP rozwijało się wraz z rozkwitem Internetu i szybko stało się jedną z wiodących technologii na rynku.

W 2002 roku opublikowana została technologia ASP.NET, która pojawiła się wraz z platformą .NET Framework 1.0. Wprowadziła ona obiektowość do aplikacji sieciowych oraz rozdzieliła warstwę prezentacji z kodem statycznym, od warstwy logiki z kodem zapewniającym funkcjonalność, pisany głównie w C# i VisualBasic. Te cechy znacznie zwiększyły produktywność programistów, przejrzystość kodu oraz pozwoliły na wielokrotne wykorzystywanie kodu [1].

Wzorzec ASP.NET MVC został wydany w 2007 roku, jako dodatek do Visual Studio w wersji 9. Połączyl on rozwiązania znane z ASP.NET z wzorcem architektonicznym MVC (Model-View-Controller). Wzorzec MVC zapewnia podział aplikacji na trzy główne składowe: model, widok i kontroler, oraz ich separację. Model

jest to zestaw klas opisujących dane wykorzystywane w aplikacji, a także reguły opisujące manipulację tymi danymi. Widok to interfejs użytkownika aplikacji. Kontroler jest zestawem klas zawierających metody odpowiedzialne za komunikację z użytkownikiem, przepływem danych oraz logiką aplikacji [1].

Inne wzorce projektowe to na przykład Model-View-ViewModel (MVVM) lub Model-View-Presenter (MVP). MVVM oddziela widok, czyli warstwę prezentacji zawierającą interfejs użytkownika od modelu, czyli logiki programu. Wykorzystuje do tego ViewModel, czyli obiekt będący łącznikiem między widokiem a modelem, zawierający informacje o zależnościach między nimi oraz pilnujący, aby warstwa widoku zawierała dane aktualne względem modelu [16]. MVP podobnie jak wyżej opisywane wzorce ma na celu oddzielenie interfejsu użytkownika od logiki biznesowej. Rolę pośrednika między widokiem i modelem pełni tu prezenter [17].

Bardziej popularne i darmowe, aczkolwiek oferujące mniejsze możliwości, jest zastosowanie języka PHP do obsługi połączeń oraz zarządzania danymi po stronie serwera. Nie zdecydowaliśmy się na to rozwiązanie z powodu braku znajomości środowiska tworzenia aplikacji i samego języka oraz, według nas, jego ograniczonych możliwości.

2.4. WCF(Paweł Gawroński)

Windows Communication Foundation to mechanizm przeznaczony do tworzenia usług sieciowych wprowadzony w 2006 roku wraz z .NET Framework w wersji 3.0. WCF jest połączeniem takich technologii jak: usługi sieciowe ASP.NET (nazywane również ASMX), .NET Remoting zapewniający szybką wymianę wiadomości między aplikacjami .NET, Enterprise Services zawierający narzędzia do projektowania aplikacji rozproszonych, WSE (Web Services Enhancements) pozwalające na tworzenie usług korzystających z protokołów komunikacyjnych z grupy WS-*, czy MSMQ (Microsoft Message Queuing) zapewniająca, że wiadomość dotrze do odbiorcy, dzięki czemu zapewnia pełną obsługę komunikacji między różnymi platformami i frameworkami [18]. Windows Communication Foundation zawarty w czwartej wersji .NET Framework pozwala między innymi na uproszczoną konfigurację, dynamiczne nawiązywanie połączeń pomiędzy klientem a hostem oraz rozsyłanie wiadomości do poszczególnych punktów końcowych na podstawie jej zawartości bądź struktury [19].

WCF powstało z połączenia wielu technologii specjalizujących się w konkretnych zadaniach, dlatego nie ma drugiego tak rozbudowanego narzędzia do tworzenia usług sieciowych.

2.5. MySQL (Paweł Gawroński, Dariusz Kohlandt)

MySQL jest jednym z najpopularniejszych systemów do zarządzania relacyjnymi bazami danych opartych na licencji GPL. MySQL został stworzony w roku 1995 przez Michaela Wideniusa w ramach projektu Open Source. Następnie technologia ta trafiła w ręce programistów z firm MySQL AB i Sun Microsystems, aby ostatecznie w 2010 roku zostać wykupiona przez firmę Oracle. Wykorzystywany w naszym projekcie MySQL w wersji 5.5 zapewnia większą wydajność oraz stabilność w porównaniu do poprzednich wersji [20].

Wyborem oczywistym byłaby tutaj baza danych MSSQL, która ma lepsze środowisko do administracji i obsługi, jest lepiej zintegrowana z technologią, której używamy do programowania, lecz zależało nam na możliwości udostępnienia użytkownikom aplikacji darmowego narzędzia do zarządzania bazą danych obsługiwanej z poziomu przeglądarki internetowej. Baza MSSQL oraz przeznaczony do zarządzania nią Microsoft SQL Server nie pozwalał na zarządzanie bazą z poziomu przeglądarki, więc zdecydowaliśmy się na MySQL w połączeniu z phpMyAdminem.

2.6. JavaScript oraz jQuery (Dariusz Kohlandt)

JavaScript (JS) jest prototypowym językiem skryptowym, który jest dynamiczny oraz słabo typowany. Został stworzony przez firmę Netscape i stosuje się go najczęściej na stronach internetowych. Skrypty utworzone w JS i umieszczone na stronach internetowych zazwyczaj zapewniają interaktywność poprzez dynamiczne reagowanie na zdarzenia, sprawdzanie poprawności wprowadzonych danych w formularzach lub tworzenie elementów nawigacyjnych [21].

jQuery jest biblioteką programistyczną stworzoną dla języka JavaScript, która w znaczny sposób ułatwia posługiwanie się językiem JavaScript. Została ona wydana w 2006 roku i jest najpopularniejszą biblioteką JavaScript w czasach obecnych. Składnia jQuery jest zaprojektowana w taki sposób by łatwiej nawigować po dokumencie, manipulować elementami DOM'u (Document Object Model), tworzyć animacje,

przechwytywać zdarzenia oraz budować aplikacje AJAX (Asynchronous JavaScript and XML). Biblioteka jQuery jest udostępniana na licencji GPL (General Public License) oraz X11 inaczej zwaną MIT (Massachusetts Institute of Technology) [22].

2.7. Chmury danych (Dariusz Kohlandt)

Chmura danych (cloudstorage) jest modelem sieciowego miejsca internetowego, gdzie dane są składowane w wirtualizowanych kontenerach, które są udostępniane głównie przez osoby trzecie. Osoby, które potrzebują miejsca na dane, wykupują je od firm hostingowych. Operatorzy centrów danych w tle wirtualizują zasoby w zależności od potrzeb klienta i wystawiają je publicznie, jako kontenery danych, w których klienci mogą umieszczać swoje pliki lub inne dane. Fizycznie zasoby mogą rozprzestrzeniać się na wiele serwerów [23].

Obecnym standardem trzymania plików są serwery FTP. Z powodu ograniczeń stawianych przez usługodawców nie byliśmy w stanie wykorzystać tej technologii (brak darmowego publicznego anonimowego dostępu do danych). Samo FTP jest protokołem transferu plików typu klient - serwer, który wykorzystuje protokół TCP pozwalający na transfer plików w dwóch kierunkach [24].

2.8. HTML i CSS (Dariusz Kohlandt)

HTML5 jest językiem znaczników przeznaczonym do tworzenia oraz prezentowania zawartości dla Światowej Sieci Internetowej (WWW). Jest to główna technologia Internetu zaproponowana przez Opera Software. Technologia ta jest rozwinięciem HTML4 oraz XHTML. Oficjalna specyfikacja tego języka została wydana kilkanaście dni temu. Przeglądarki internetowe muszą wspierać nową wersję HTML w celu poprawnego wyświetlania stron internetowych używających funkcjonalności HTML5 [25].

Kaskadowe arkusze stylów (Cascading Style Sheets) są językiem arkuszy stylów używanym do opisu wyglądu oraz formatowania dokumentu napisanego przy użyciu języka znaczników. Jego najczęstszym zastosowaniem jest stylizacja stron internetowych napisanych w HTML oraz XHTML. Jednak może być również zastosowany dla każdego rodzaju dokumentu XML włączając w to SVG (Scalable Vector Graphics) oraz XUL (XML-based User-interface Language).

CSS jest zaprojektowany w celu odseparowania zawartości dokumentu od części związanej ze stylem prezentacji treści dokumentu. Taki podział poprawia dostęp do treści, zapewnia większą elastyczność i kontrolę, umożliwia korzystanie przez większość stron z jednego pliku, w którym zdefiniowano styl oraz redukuje złożoność zawartości struktury dokumentu [26].

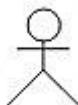
3. PROJEKT APLIKACJI (Paweł Gawroński, Dariusz Kohlandt)

3.1. Opis

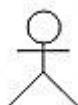
3.1.1. Opis aplikacji

Aplikacja e-Restauracja jest uniwersalnym systemem przeznaczonym do prezentacji ofert różnych restauracji oraz do tworzenia i zarządzania zamówieniami w restauracjach. Przeznaczona jest zarówno dla klientów indywidualnych jak również dla obsługi restauracji, a w szczególności dla menadżerów. Docelowo nasz system ma obsługiwać wszystkie restauracje oferujące dowóz dania do klienta. Dodatkowo system umożliwia śledzenie stanu zamówienia. Klient może sprawdzić czy jego zamówienie zostało przyjęte oraz jaki jest status jego realizacji. Poza tym aplikacja oferuje elastyczny wybór sposobu płatności, jak płatności i przelewy internetowe lub płatność gotówką przy odbiorze. Aplikacja przewiduje pięć warstw uprawnień dostępu: dla osób niezalogowanych, klientów zalogowanych i administratora strony. Dwie kolejne warstwy przewidzieliśmy dla obsługi restauracji: dla menadżerów restauracji i pracowników.

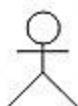
3.1.2. Opis aktorów



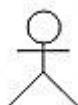
Niezalogowany użytkownik – aktor główny, aktywny. Klient przed zalogowaniem, przeglądający listę restauracji.



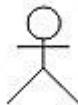
Zalogowany użytkownik – aktor główny, aktywny. Klient składający zamówienia.



Restauracja – aktor główny, pasywny. Restauracja zmienia status zamówienia i edytuje zamówienia.



Menadżer – aktor drugorzędny, aktywny. Menadżer zarządzający restauracją.



Administrator – aktor drugorzędny. Administrator systemu.

3.2. Diagramy

3.2.1. Przypadków użycia

Diagramy przypadków użycia graficznie przedstawiają możliwości, które system świadczy aktorom. Na rysunkach zawarto opis czynności, a strzałki pokazują przepływ wykonywania.

1. Przypadek użycia:

Logowanie do systemu – Rys. 3.2.1.1.;

Aktorzy:

Niezalogowany użytkownik systemu;

Zdarzenie inicjujące:

Wpisanie danych do formularza logowania oraz wybranie opcji “Zaloguj”;

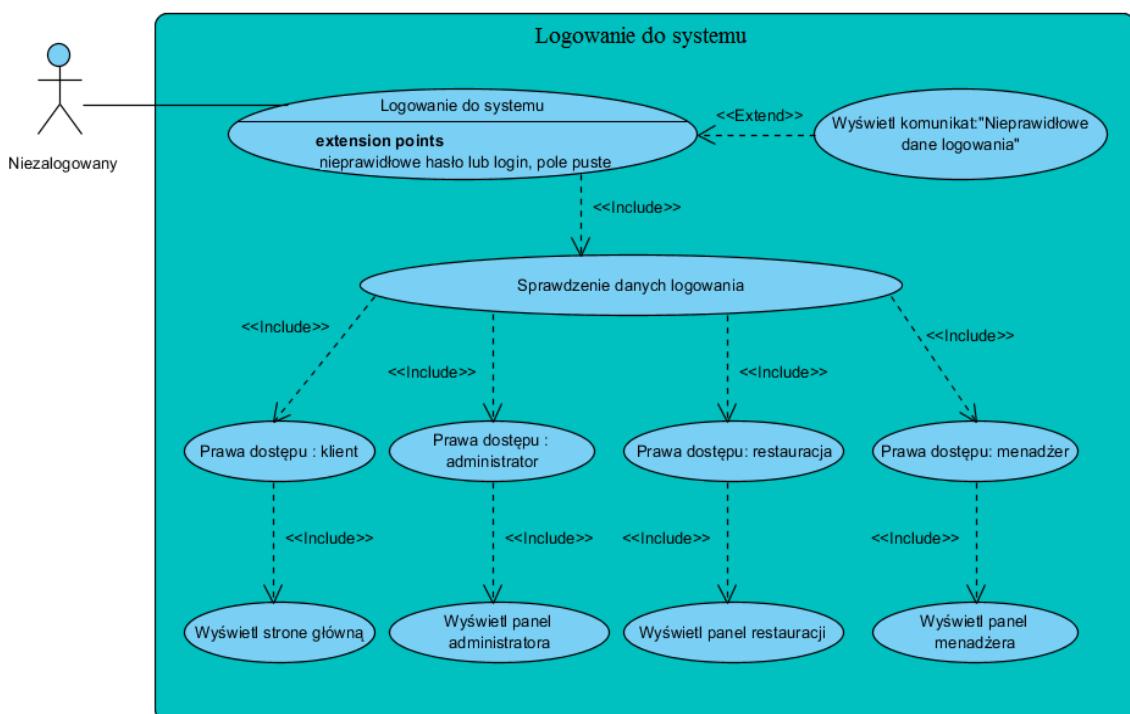
Opis:

Sprawdź dane do logowania. Gdy dane są poprawne, zidentyfikuj użytkownika i sprawdź przydzielone mu prawa dostępu. Jeżeli prawa dostępu wskazują na klienta to wyświetl stronę główną, jeżeli na administratora systemu wyświetl panel administratora, jeżeli na restaurację wyświetl panel restauracji, jeżeli na menadżera wyświetl panel menadżera;

Sytuacje wyjątkowe:

Niewypełnione pole loginu lub hasła: wyświetl informację o niewypełnieniu pola login lub hasło;

Niepoprawny login lub hasło: wyświetl informację o niepoprawnym logowaniu;



Rys. 3.2.1.1. Diagram logowania do systemu (opracowanie własne).

2. Przypadek użycia:

Wylogowanie z systemu – Rys. 3.2.1.2.;

Aktorzy:

Zalogowany i niezalogowany użytkownik systemu, restauracja, administrator;

Zdarzenie inicjujące:

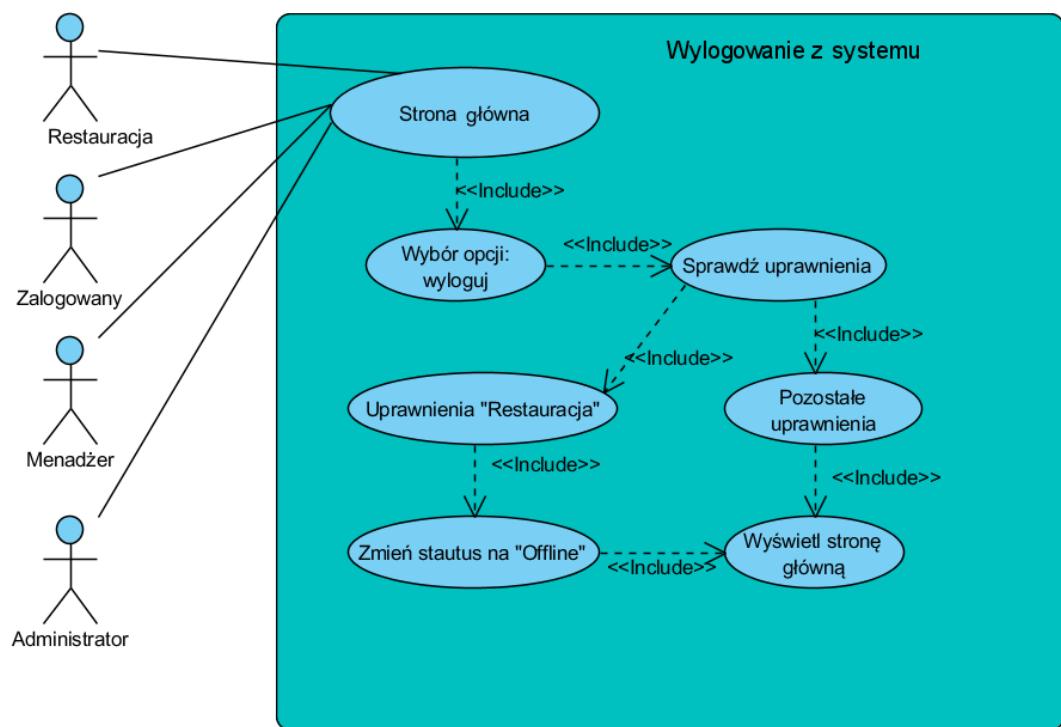
Wybranie opcji „Wyloguj”;

Opis:

Jeżeli użytkownik jest zalogowany i wybierze opcję „Wyloguj”, to wyloguj użytkownika z systemu i wyświetl stronę główną;

Sytuacje wyjątkowe:

Brak;



Rys. 3.2.1.2. Diagram wylogowania z systemu (opracowanie własne).

3. Przypadek użycia:

Wystawianie komentarzy – Rys. 3.2.1.3.;

Aktorzy:

Zalogowany użytkownik systemu;

Zdarzenie inicjujące:

Wpisanie tekstu w pole komentarza oraz wybranie opcji “Wystaw komentarz”;

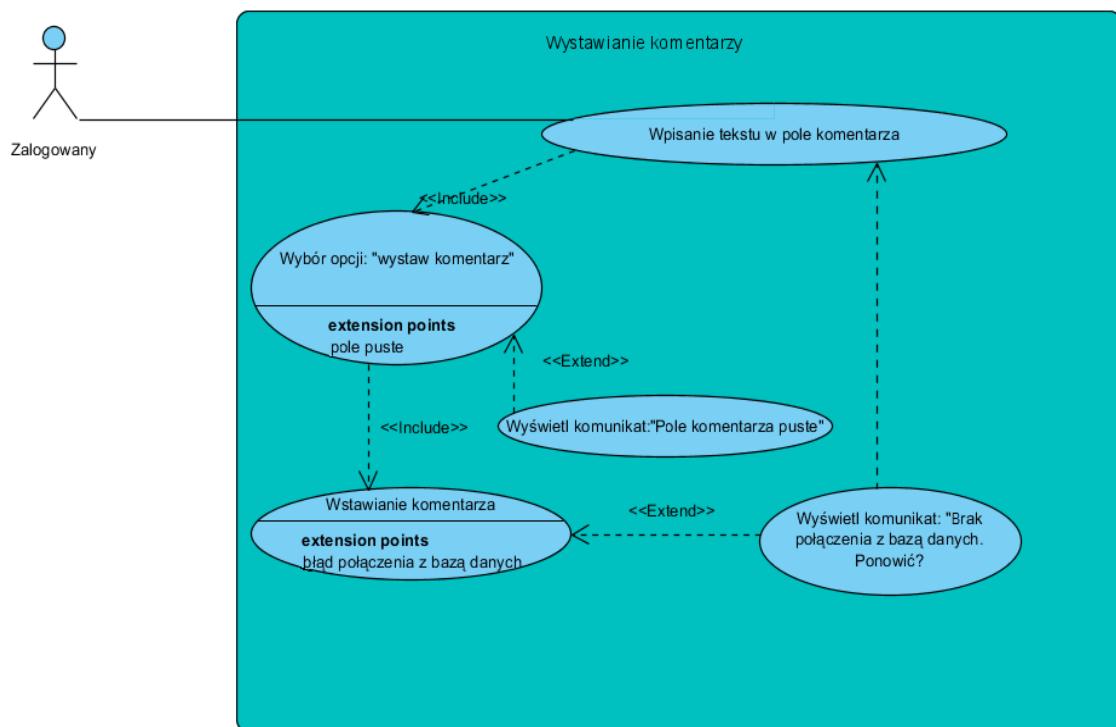
Opis:

Jeżeli pole komentarza nie jest puste i została wybrana opcja “Wystaw komentarz”, to wystaw komentarz;

Sytuacje wyjątkowe:

Błąd połączenia z bazą danych: wyświetl prośbę o ponowne przesłanie formularza;

Niewypełnione pole komentarza: wyświetl informację o niewypełnieniu zawartości pola komentarza;



Rys. 3.2.1.3. Diagram wystawiania komentarzy (opracowanie własne).

4. Przypadek użycia:

Wysłanie zamówienia – Rys. 3.2.1.4.;

Aktorzy:

Zalogowany użytkownik systemu;

Zdarzenie inicjujące:

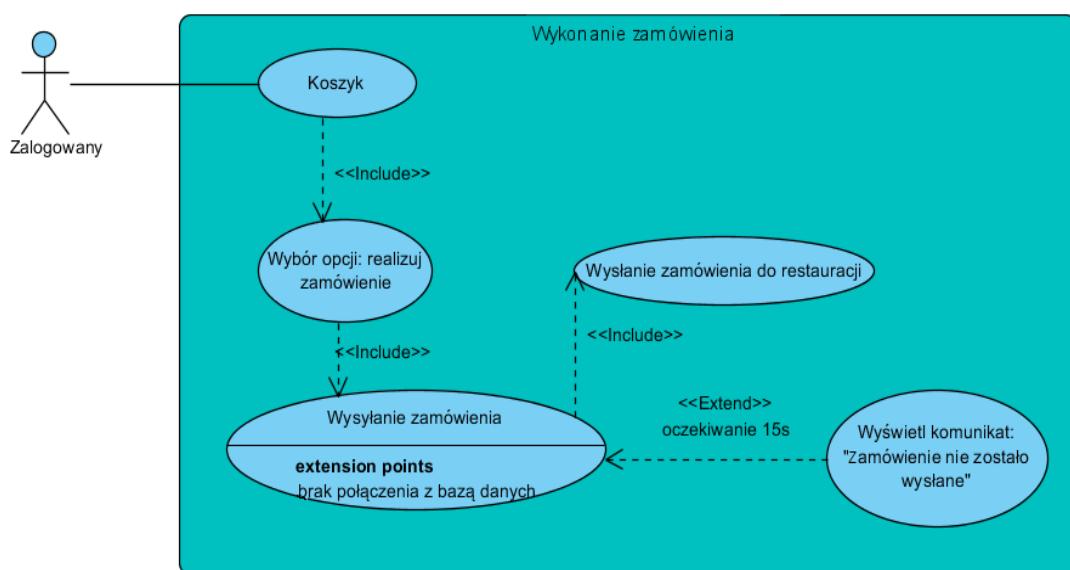
Wysłanie formularza z zamówieniem na serwer poprzez wybranie odpowiedniej opcji w koszyku;

Opis:

Jeżeli wybrana zostanie opcja “Realizuj zamówienie”, to wyslij zamówienie do odpowiedniej restauracji;

Sytuacje wyjątkowe:

Błąd połączenia z bazą danych: po 15 sekundach oczekiwania wyświetl komunikat o niepowodzeniu operacji;



Rys. 3.2.1.4. Diagram wykonania zamówienia (opracowanie własne).

5. Przypadek użycia:

Rejestracja w systemie – Rys. 3.2.1.5.;

Aktorzy:

Niezarejestrowany użytkownik systemu;

Zdarzenie inicjujące:

Wypełnienie formularza rejestracji oraz wybranie opcji “Zarejestruj”;

Opis:

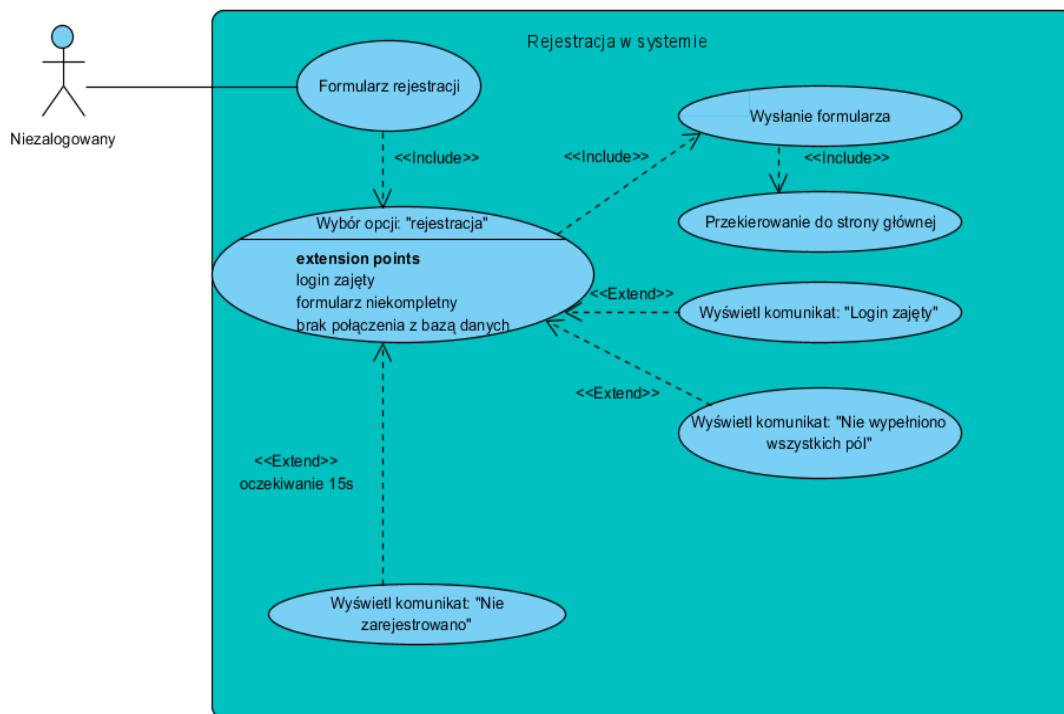
Jeżeli pola formularza zostały wypełnione i wybrana została opcja “Zarejestruj”, to wyślij potwierdzenie rejestracji na podany adres poczty elektronicznej;

Sytuacje wyjątkowe:

Błąd połączenia z bazą danych: po 15 sekundach oczekiwania wyświetl komunikat o niepowodzeniu operacji;

Niewypełnione pole formularza: wyświetl informację o niewypełnieniu obowiązkowego pola formularza;

Zajęty login: wyświetl informację o niedostępności loginu;



Rys. 3.2.1.5. Diagram rejestracji w systemie (opracowanie własne).

6. Przypadek użycia:

Dodanie nowej restauracji – Rys. 3.2.1.6.;

Aktorzy:

Menadżer;

Zdarzenie inicjujące:

Wypełnienie formularza oraz wybranie opcji “Dodaj restaurację”;

Opis:

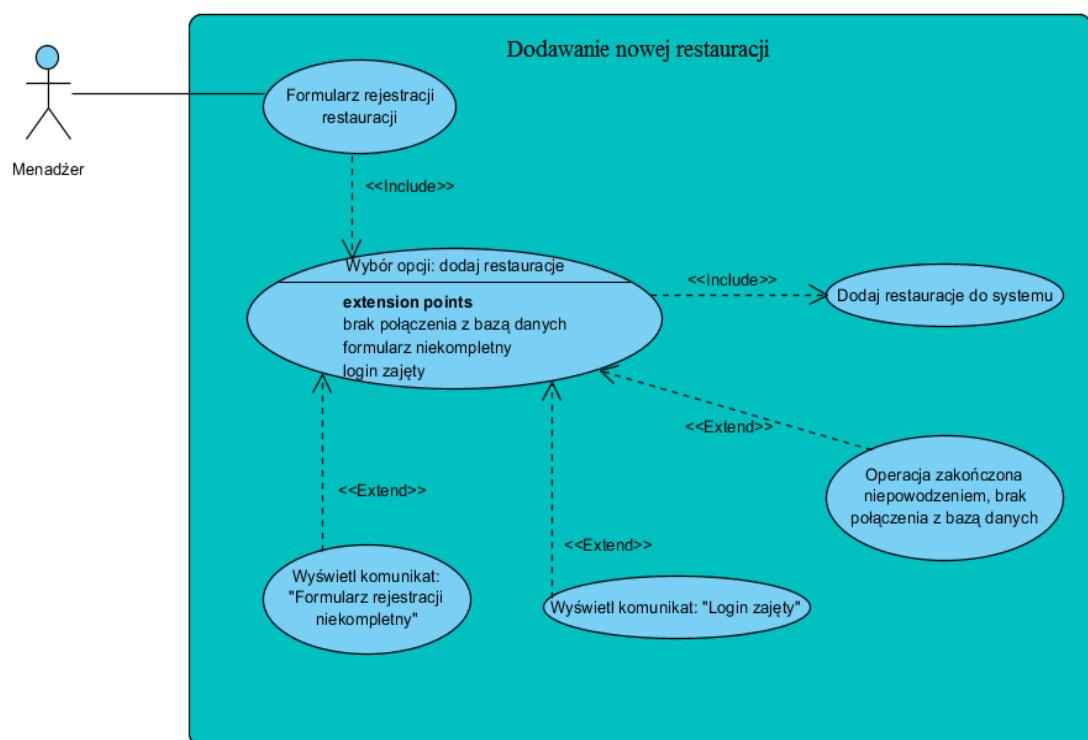
Jeżeli wszystkie pola formularza są wypełnione i wybrana została opcja “Dodaj restaurację”, to dodaj restaurację do systemu;

Sytuacje wyjątkowe:

Błąd połączenia z bazą danych: po 15 sekundach oczekiwania wyświetl komunikat o niepowodzeniu operacji;

Niewypełnione pole formularza: wyświetl informację o niewypełnieniu obowiązkowego pola formularza;

Zajęty login restauracji: wyświetl informację o niedostępności danego logina;



Rys. 3.2.1.6. Diagram dodawania nowej restauracji (opracowanie własne).

7. Przypadek użycia:

Edycja bazy danych – Rys. 3.2.1.7.;

Aktorzy:

Administrator;

Zdarzenie inicjujące:

Wybranie z panelu administracyjnego opcji “Dodaj”, “Usuń” lub “Modyfikuj”;

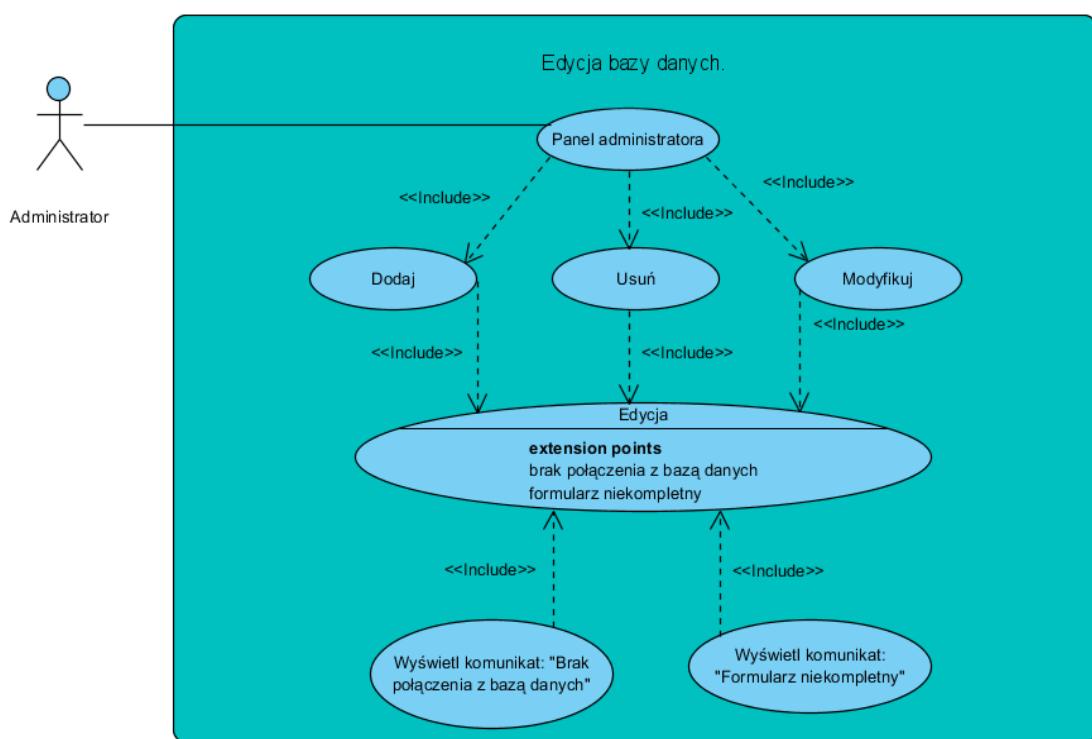
Opis:

Jeżeli wybrana została opcja “Dodaj”, “Usuń” lub “Modyfikuj” oraz pola formularza zostały wypełnione poprawnie, to po wciśnięciu przycisku „Wykonaj” wykonaj operację modyfikacji danych;

Sytuacje wyjątkowe:

Błąd połączenia z bazą danych: po 15 sekundach oczekiwania wyświetl komunikat o niepowodzeniu operacji;

Niewypełnione pole formularza: wyświetl informację o niewypełnieniu zawartości pola formularza;



Rys. 3.2.1.7. Diagram edycji bazy danych (opracowanie własne).

8. Przypadek użycia:

Wyszukiwanie miasta lub restauracji – Rys. 3.2.1.8.;

Aktorzy:

Niezalogowany użytkownik systemu, zalogowany użytkownik systemu;

Zdarzenie inicjujące:

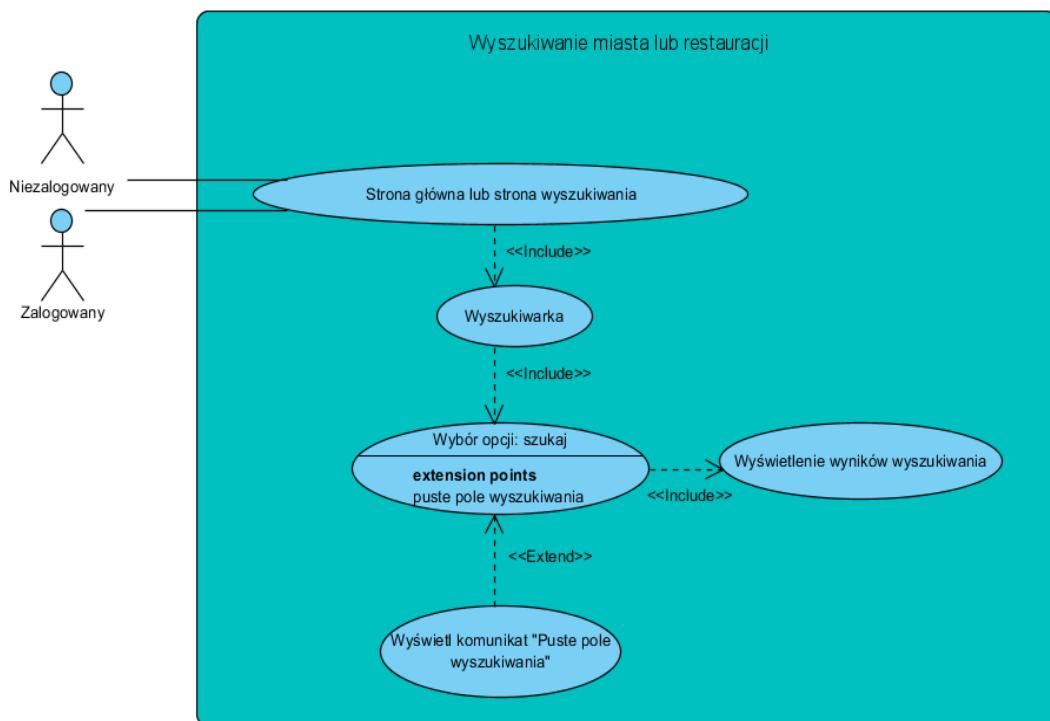
Wpisanie nazwy miasta lub restauracji do odpowiedniego pola wyszukiwania oraz wybranie opcji “Szukaj”;

Opis:

Sprawdź dane do wyszukiwania. Gdy dane są poprawne, to sprawdź czy szukane miasto lub restauracja istnieją w systemie, jeżeli tak to wyświetl wyniki wyszukiwania. Jeżeli nie to wyświetl informacje o braku wyników;

Sytuacje wyjątkowe:

Niewypełnione pole wyszukiwania: wyświetl informację o niepowodzeniu;



Rys. 3.2.1.8. Diagram wyszukiwania miasta lub restauracji (opracowanie własne).

9. Przypadek użycia:

Wyświetlanie informacji o wybranej restauracji – Rys. 3.2.1.9.;

Aktorzy:

Niezalogowany użytkownik systemu, zalogowany użytkownik systemu;

Zdarzenie inicjujące:

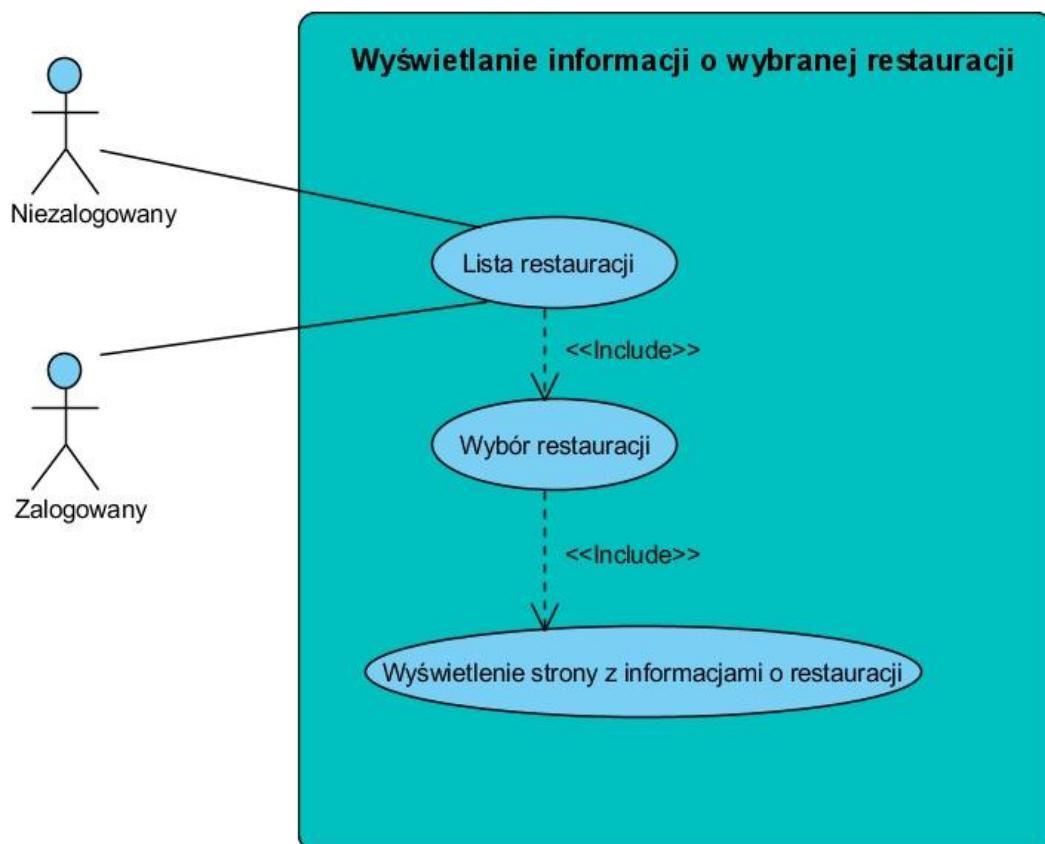
Wybranie restauracji poprzez kliknięcie w pole oznaczone przez nazwę restauracji;

Opis:

Jeżeli restauracja wybrana została prawidłowo, to wyświetl informację o wybranej restauracji;

Sytuacje wyjątkowe:

Brak;



Rys. 3.2.1.9. Diagram wyświetlania informacji o wybranej restauracji (opracowanie własne).

10. Przypadek użycia:

Wyświetlanie informacji o aktualnym zamówieniu – Rys. 3.2.1.10.;

Aktorzy:

Zalogowany użytkownik systemu;

Zdarzenie inicjujące:

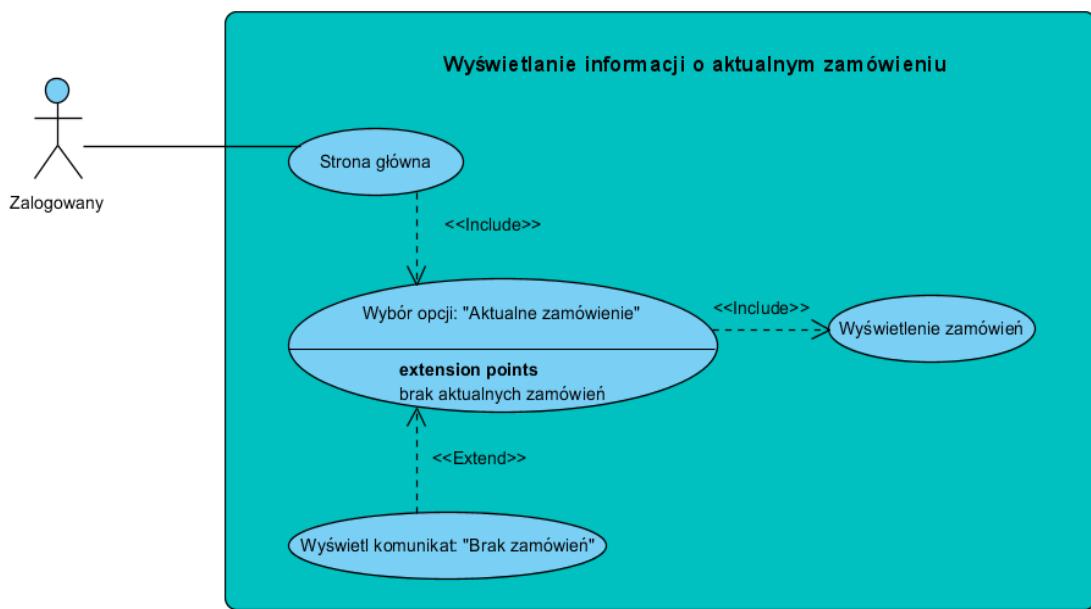
Wybranie opcji „Aktualne zamówienie”;

Opis:

Jeżeli zamówienie użytkownika jest w trakcie realizacji i została wybrana opcja „Aktualne zamówienie”, to wyświetl informację o aktualnym zamówieniu;

Sytuacje wyjątkowe:

Brak aktualnych zamówień: wyświetl informację o braku aktualnych zamówień;



Rys. 3.2.1.10. Diagram wyświetlania informacji o aktualnym zamówieniu (opracowanie własne).

11. Przypadek użycia:

Zgłaszanie błędów – Rys. 3.2.1.11.;

Aktorzy:

Niezalogowany i zalogowany użytkownik systemu, menadżer;

Zdarzenie inicjujące:

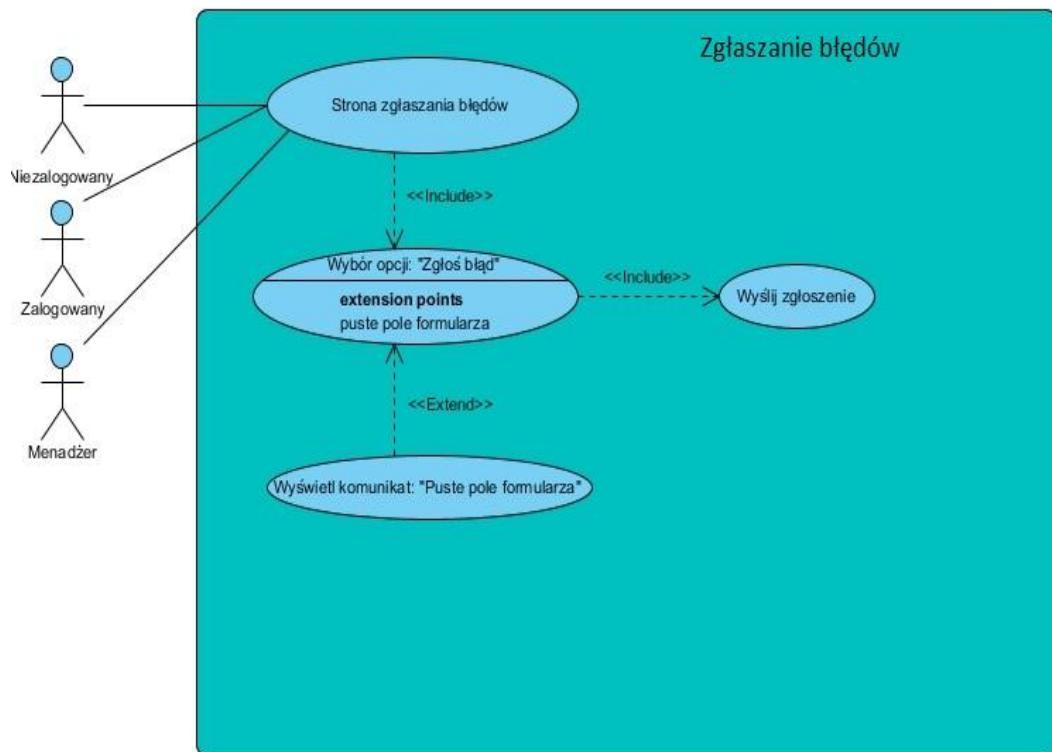
Wypełnienie formularza do zgłaszania błędów oraz wybranie opcji “Wyślij”;

Opis:

Jeśli pola formularza są wypełnione prawidłowo, to wyślij zgłoszenie na adres poczty elektronicznej systemu;

Sytuacje wyjątkowe:

Niewypełnione pole formularza: wyświetl informację o niewypełnieniu obowiązkowego pola;



Rys. 3.2.1.11. Diagram zgłaszanego błędów (opracowanie własne).

12. Przypadek użycia:

Przywracanie hasła – Rys. 3.2.1.12.;

Aktorzy:

Niezalogowany użytkownik systemu;

Zdarzenie inicjujące:

Podanie loginu, wciśnięcie przycisku „Dalej” oraz udzielenie prawidłowej odpowiedzi na pytanie od przywracania hasła i ponowne wciśnięcie przycisku „Dalej”;

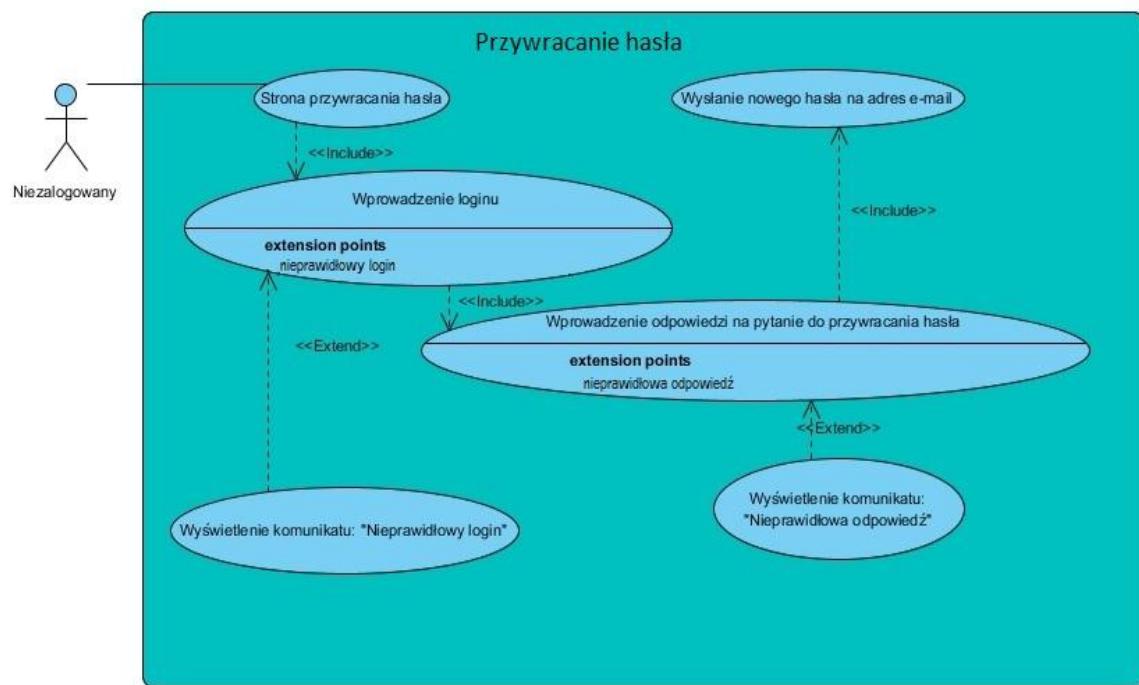
Opis:

Jeśli pola formularza są wypełnione prawidłowo, to wyślij nowe hasło na adres email użytkownika;

Sytuacje wyjątkowe:

Niewypełnione pole formularza: wyświetl informację o niewypełnieniu obowiązkowego pola;

Niepoprawnie wypełnione pola formularza: wyświetl informację o nieprawidłowych danych;



Rys. 3.2.1.12. Diagram przywracania hasła (opracowanie własne).

13. Przypadek użycia:

Edycja danych użytkownika – Rys. 3.2.1.13.;

Aktorzy:

Zalogowany użytkownik systemu;

Zdarzenie inicjujące:

Wypełnienie formularza do edycji danych oraz wybranie opcji „Zapisz”;

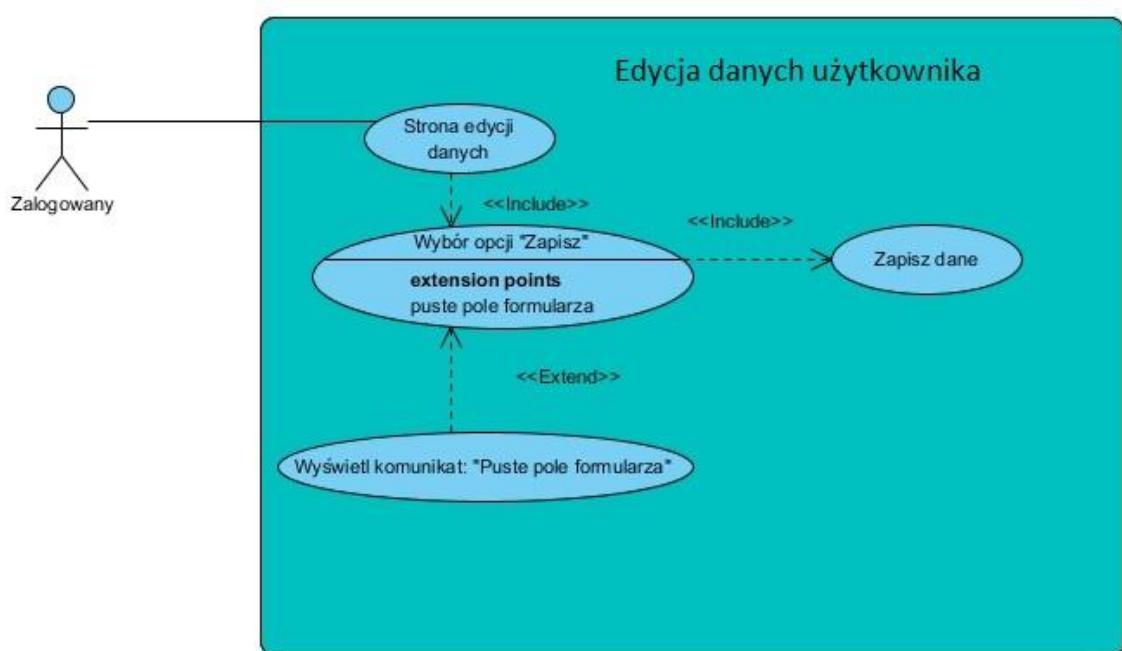
Opis:

Jeśli pola formularza są wypełnione prawidłowo, to zapisz nowe dane do bazy danych;

Sytuacje wyjątkowe:

Niewypełnione pole formularza: wyświetl informację o niewypełnieniu obowiązkowego pola;

Niepoprawnie wypełnione pola formularza: wyświetl informację o nieprawidłowych danach;



Rys. 3.2.1.13. Diagram edycji danych użytkownika (opracowanie własne).

14. Przypadek użycia:

Edycja danych restauracji – Rys. 3.2.1.14.;

Aktorzy:

Menadżer;

Zdarzenie inicjujące:

Wypełnienie formularza do edycji danych oraz wybranie opcji „Zapisz”;

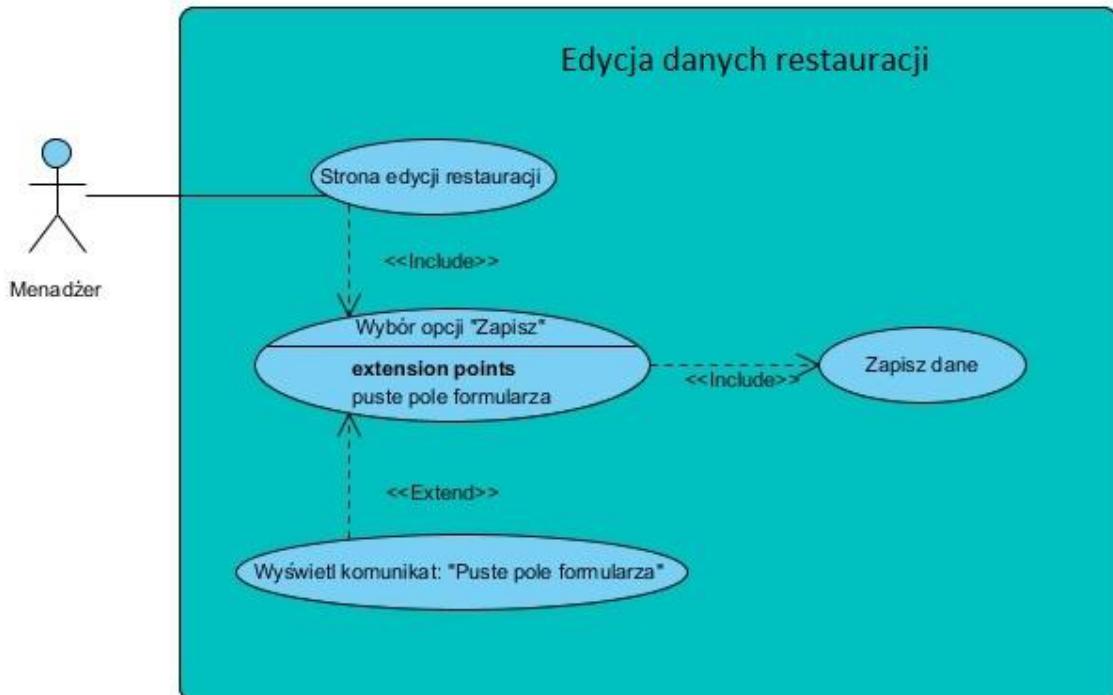
Opis:

Jeśli pola formularza są wypełnione prawidłowo, to zapisz nowe dane restauracji do bazy danych;

Sytuacje wyjątkowe:

Niewypełnione pole formularza: wyświetl informację o niewypełnieniu obowiązkowego pola;

Niepoprawnie wypełnione pola formularza: wyświetl informację o nieprawidłowych danych;



Rys. 3.2.1.14. Diagram edycji danych restauracji (opracowanie własne).

15. Przypadek użycia:

Edycja strony restauracji – Rys. 3.2.1.15.;

Aktorzy:

Menadżer;

Zdarzenie inicjujące:

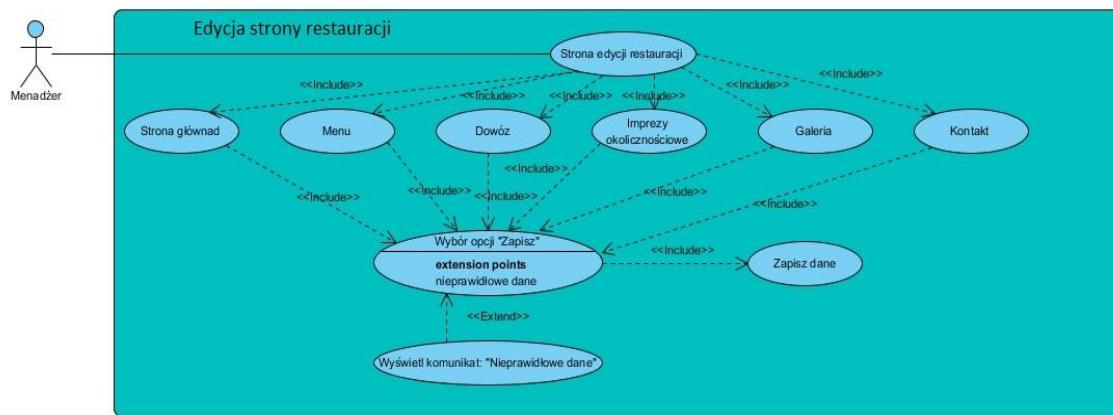
Wypełnienie formularza do edycji strony restauracji oraz wybranie opcji „Zapisz”;

Opis:

Jeśli pola formularza są wypełnione prawidłowo, to zapisz nowe dane do bazy danych;

Sytuacje wyjątkowe:

Niepoprawnie wypełnione pola formularza: wyświetli informację o nieprawidłowych danych;



Rys. 3.2.1.15. Diagram edycji strony restauracji (opracowanie własne).

16. Przypadek użycia:

Podgląd zamówienia – Rys. 3.2.1.16.;

Aktorzy:

Restauracja;

Zdarzenie inicjujące:

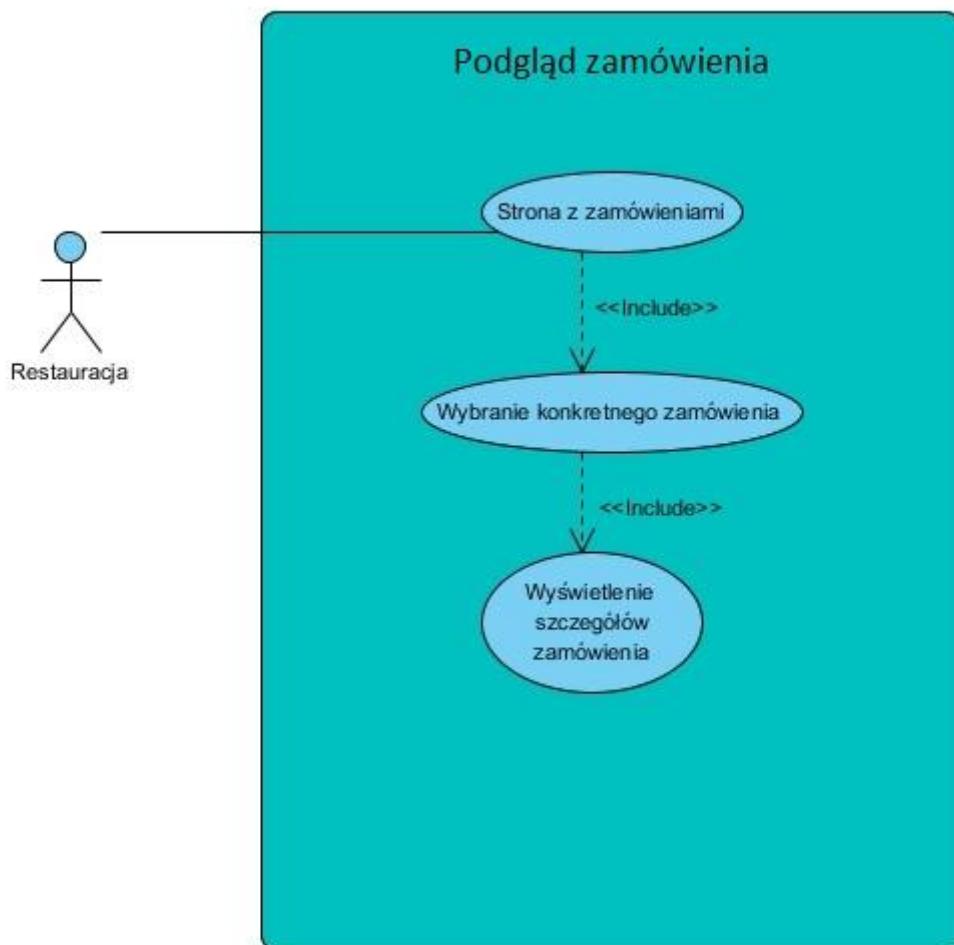
Wybranie zamówienia z listy aktywnych zamówień poprzez kliknięcie na zamówienie;

Opis:

Wyświetlenie szczegółów zamówienia po kliknięciu na zamówienie;

Sytuacje wyjątkowe:

Brak;



Rys. 3.2.1.16. Diagram podglądu zamówienia (opracowanie własne).

17. Przypadek użycia:

Zmiana statusu zamówienia – Rys. 3.2.1.17.;

Aktorzy:

Restauracja;

Zdarzenie inicjujące:

Wybranie zamówienia z listy aktywnych zamówień poprzez kliknięcie na zamówienie, a następnie wybór odpowiedniego statusu zamówienia z listy;

Opis:

Nadanie nowego statusu dla aktualnie wybranego zamówienia;

Sytuacje wyjątkowe:

Brak;



Rys. 3.2.1.17. Diagram zmiany statusu zamówienia (opracowanie własne).

18. Przypadek użycia:

Podgląd wszystkich zamówień – Rys. 3.2.1.18.;

Aktorzy:

Restauracja;

Zdarzenie inicjujące:

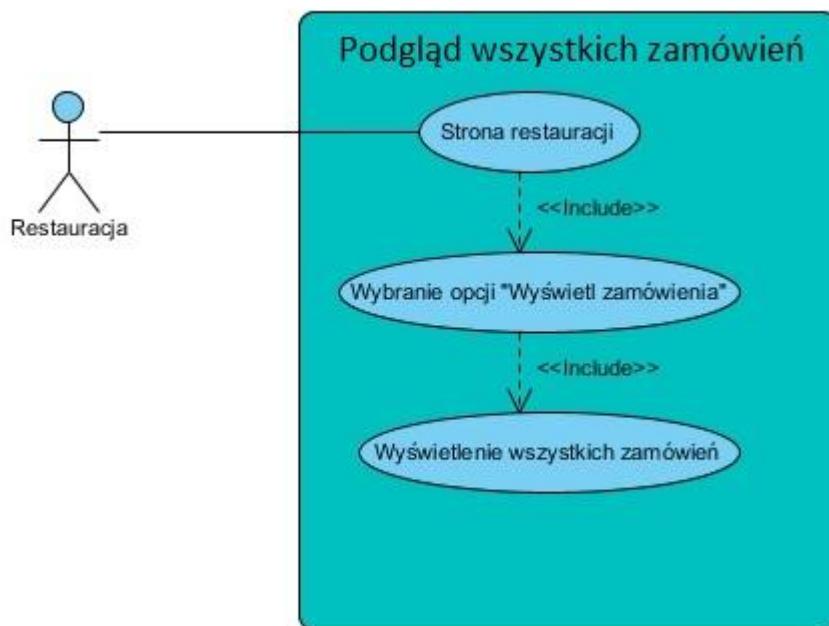
Z menu głównego wybranie opcji "Wszystkie zamówienia";

Opis:

Wyświetlenie strony z wszystkimi zamówieniami;

Sytuacje wyjątkowe:

Brak;

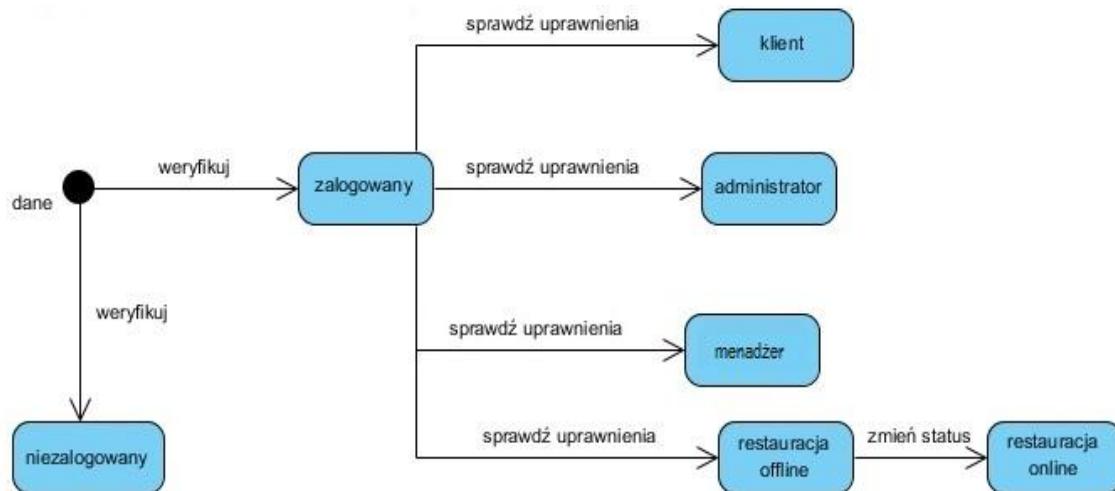


Rys. 3.2.1.18. Diagram podglądu wszystkich zamówień (opracowanie własne).

3.2.2. Diagramy maszyny stanowej

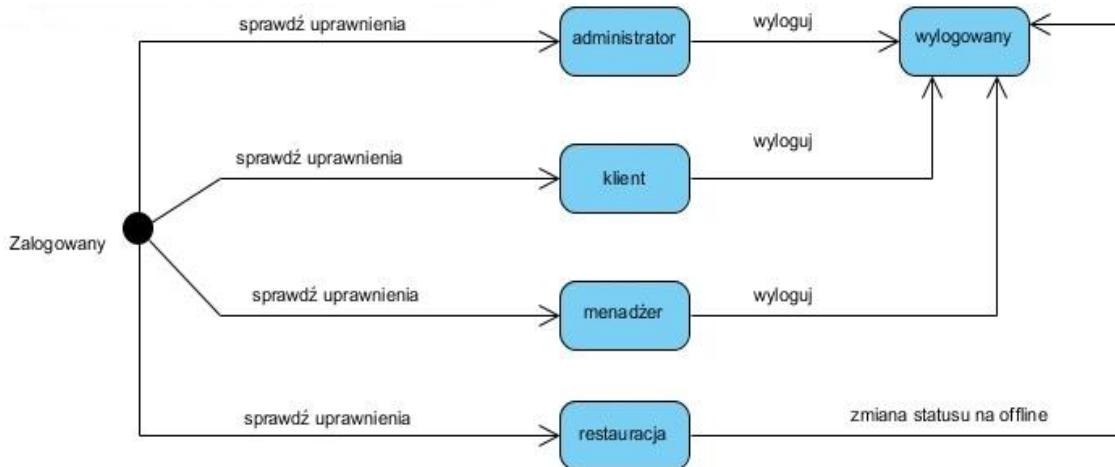
Diagramy maszyny stanowej ukazują wszystkie możliwe stany dla danego obiektu oraz przejścia pomiędzy tymi stanami.

1. Diagram przedstawiający stan przydzielonego uprawnienia/roli przy logowaniu (Rys. 3.2.2.1.).



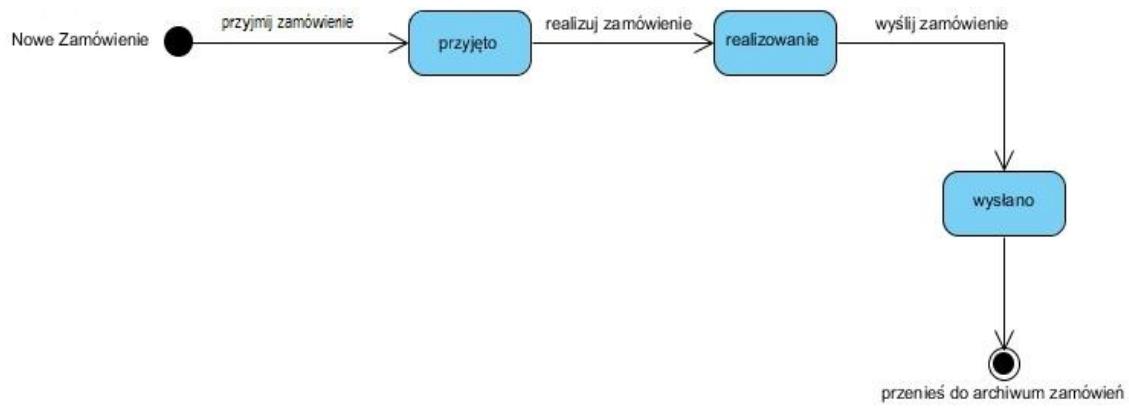
Rys. 3.2.2.1. Diagram przydzielania uprawnień po zalogowaniu (opracowanie własne).

2. Diagram przedstawiający stan przydzielonego uprawnienia/roli przy wylogowaniu (Rys. 3.2.2.2.).



Rys. 3.2.2.2. Diagram rozpoznawania uprawnień przy wylogowaniu (opracowanie własne).

3. Diagram przedstawiający stan zamówienia (Rys. 3.2.2.3).

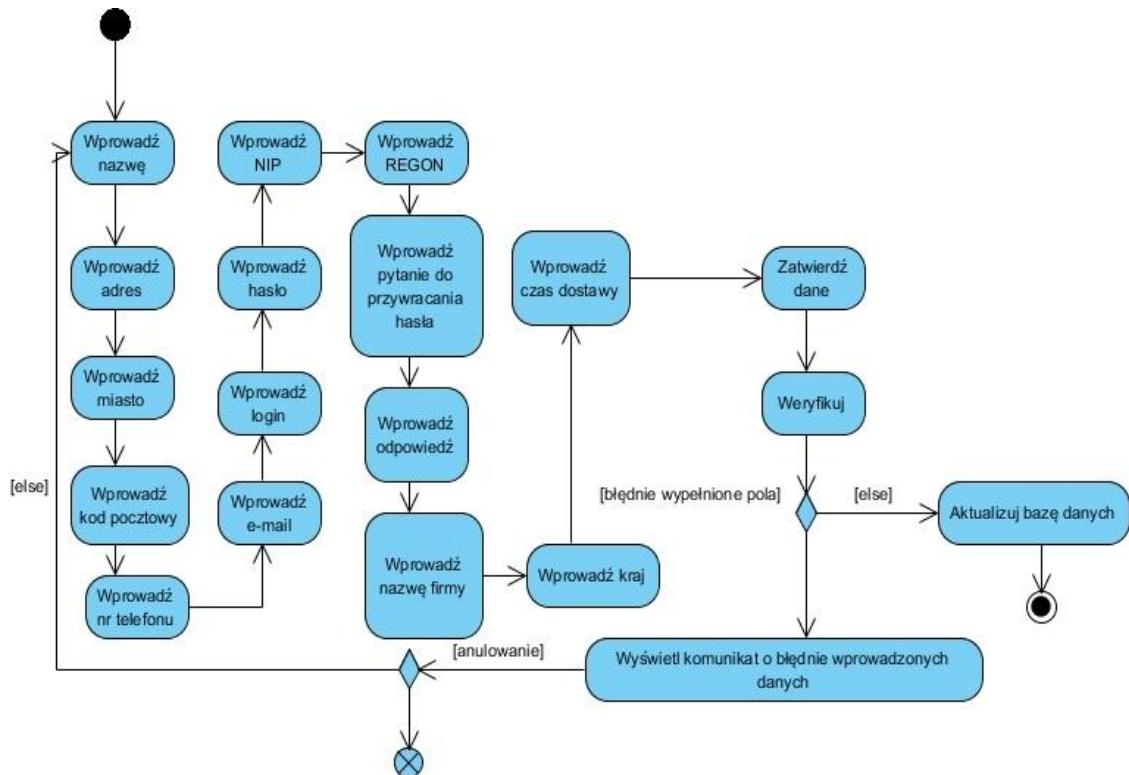


Rys. 3.2.2.3. Diagram przedstawiający stan zamówienia (opracowanie własne).

3.2.3. Diagramy czynności

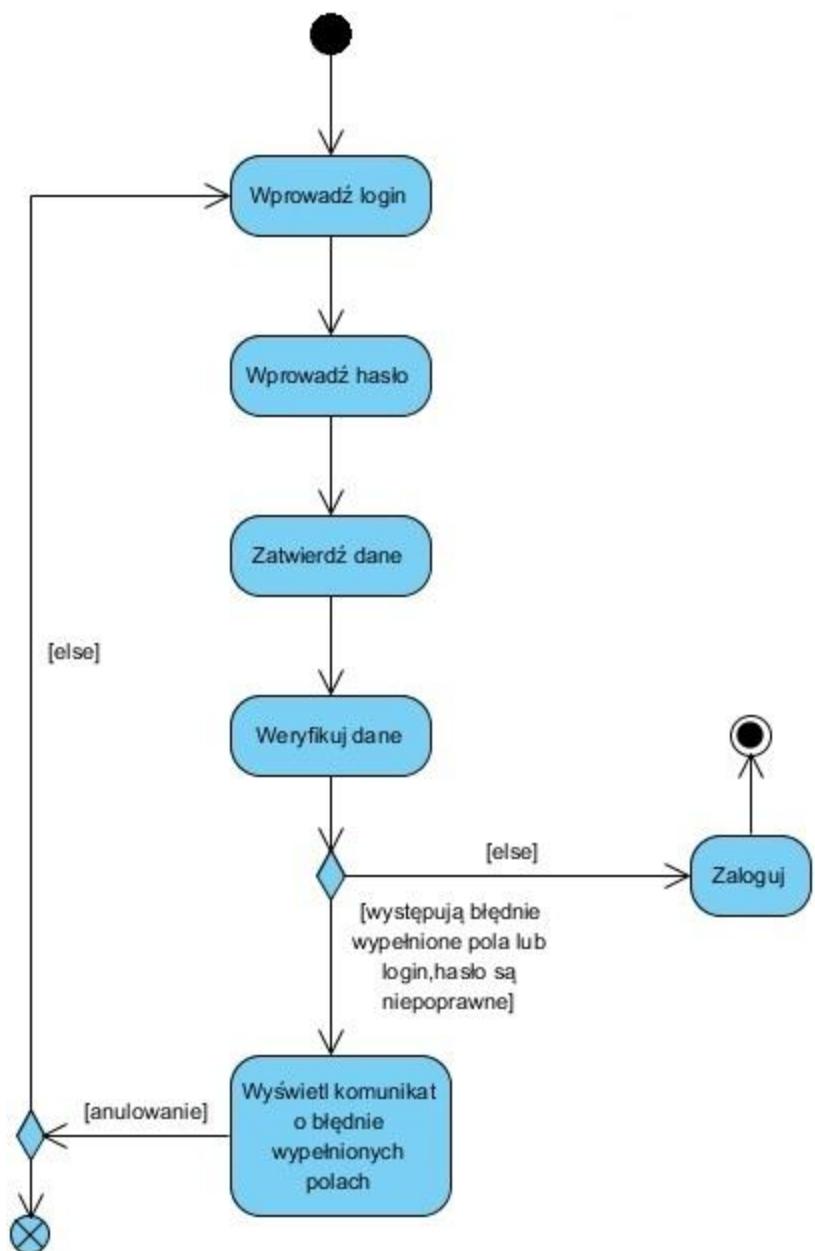
Diagramy czynności obrazują przepływ wykonywanych czynności, zaczynając od początkowego stanu akcji (wypełnione koło) do końcowego stanu (wypełnione koło w okręgu). Po drodze rozróżniamy stany akcji (prostokąt z zaokrąglonymi rogami), przejścia przepływu sterowania (strzałka ciągła), decyzje (romby) oraz punkty końcowe (koło z krzyżem w środku).

1. Diagram czynności dla dodania nowej restauracji (Rys. 3.2.3.1).



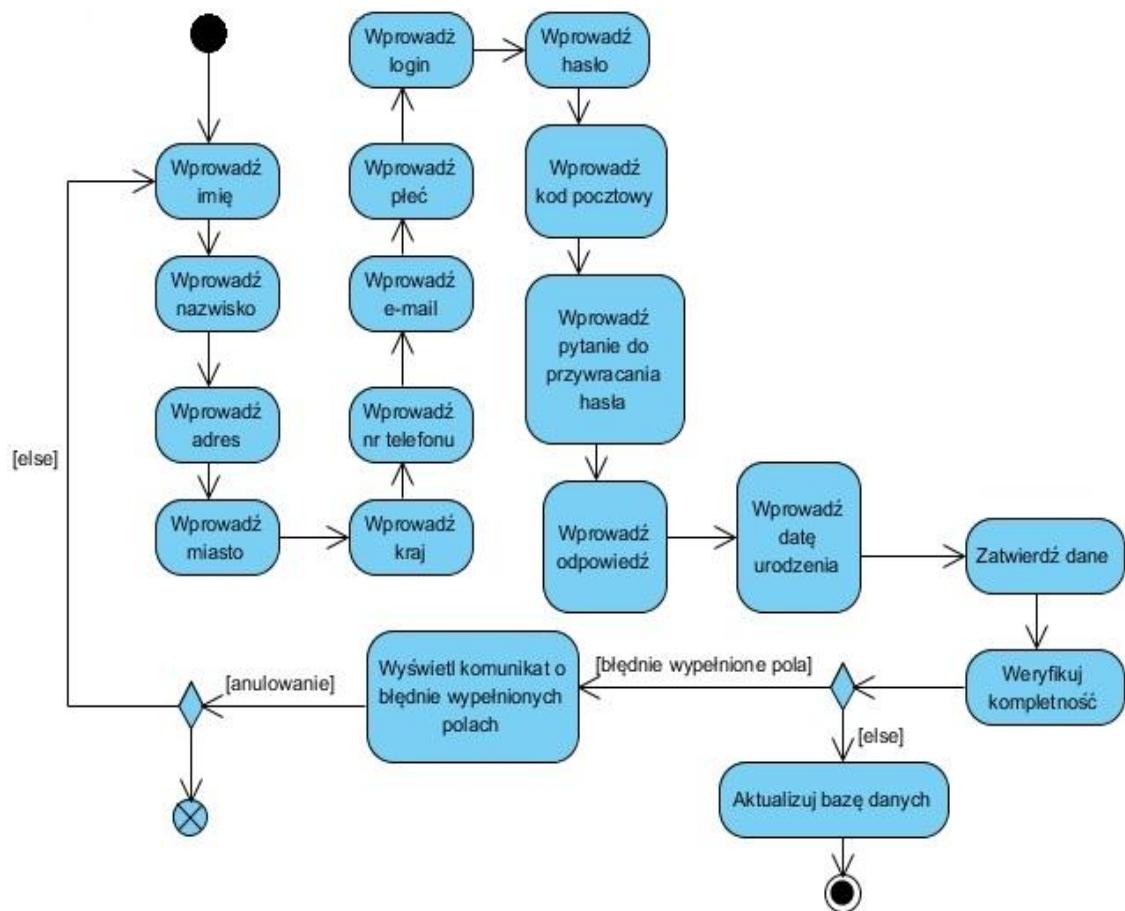
Rys. 3.2.3.1. Diagram dodania nowej restauracji (opracowanie własne).

2. Diagram czynności dla logowania (Rys. 3.2.3.2).



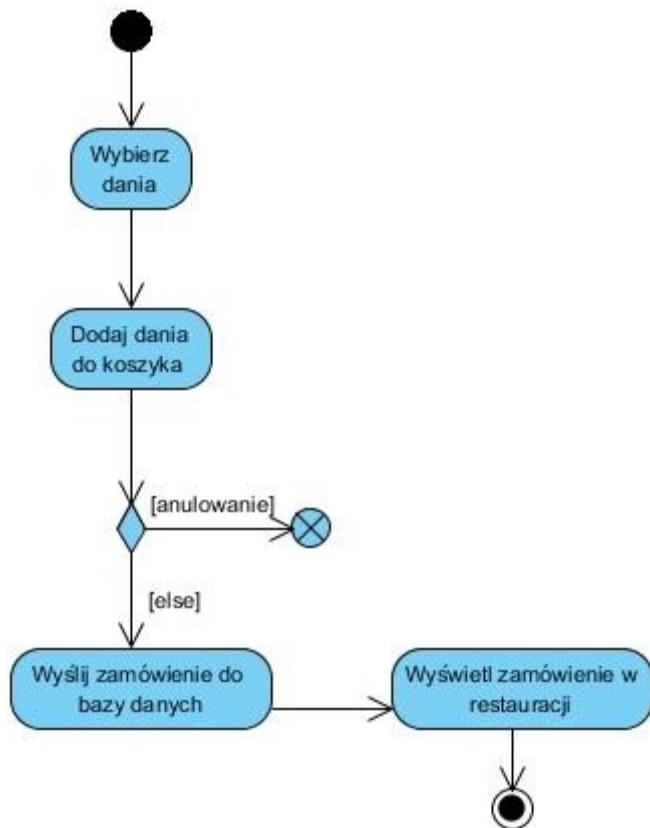
Rys. 3.2.3.2. Diagram czynności dla logowania (opracowanie własne).

3. Diagram czynności dla rejestracji użytkownika (Rys. 3.2.3.3).



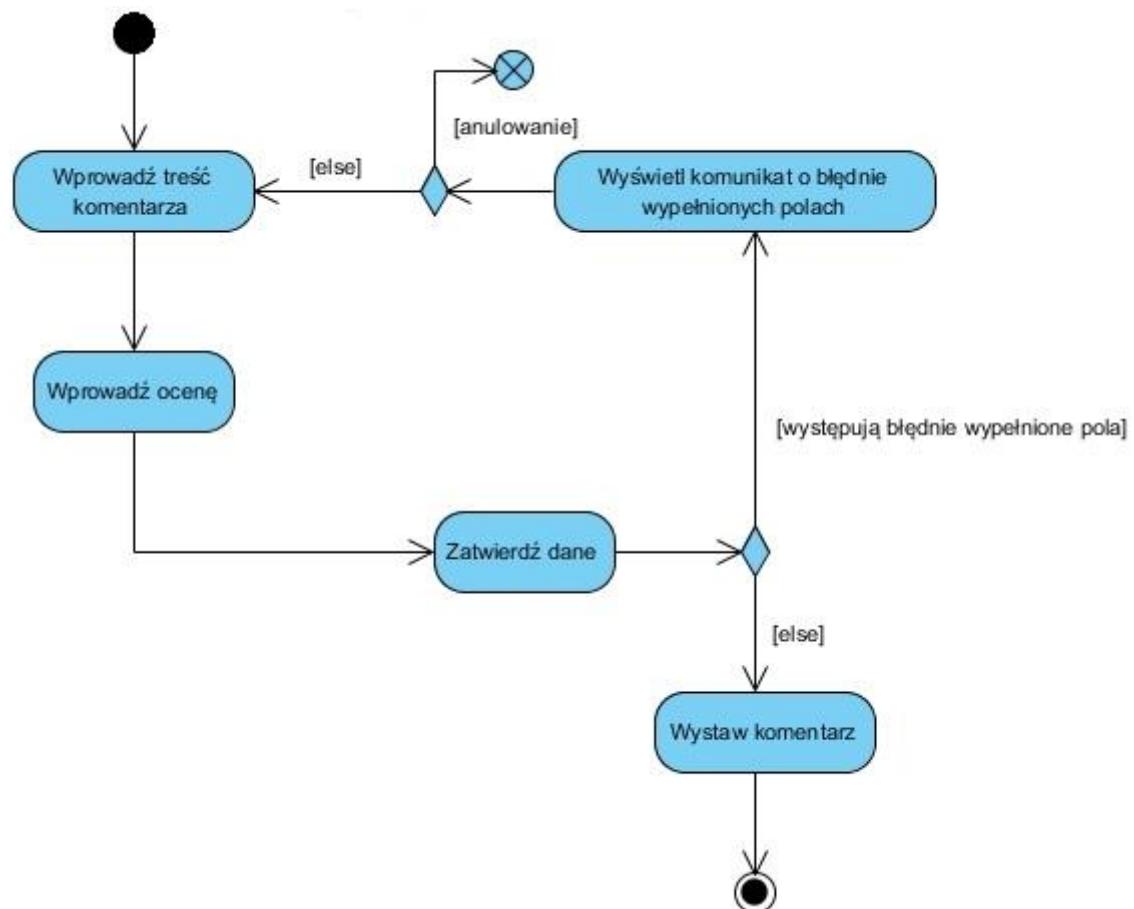
Rys. 3.2.3.3. Diagram czynności dla rejestracji użytkownika (opracowanie własne).

4. Diagram czynności dla wykonania zamówienia (Rys. 3.2.3.4).



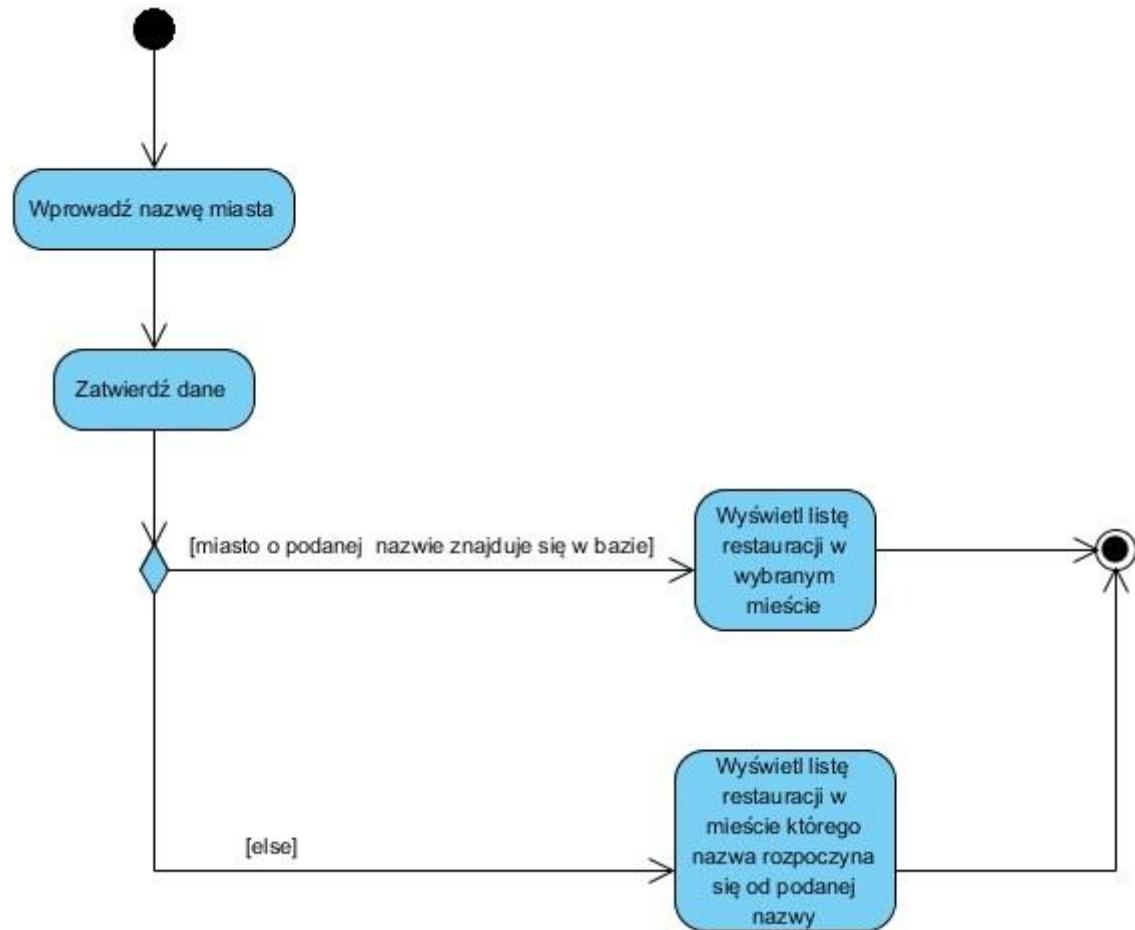
Rys. 3.2.3.4. Diagram czynności dla wykonania zamówienia (opracowanie własne).

5. Diagram czynności dla wystawienia komentarza (Rys. 3.2.3.5).



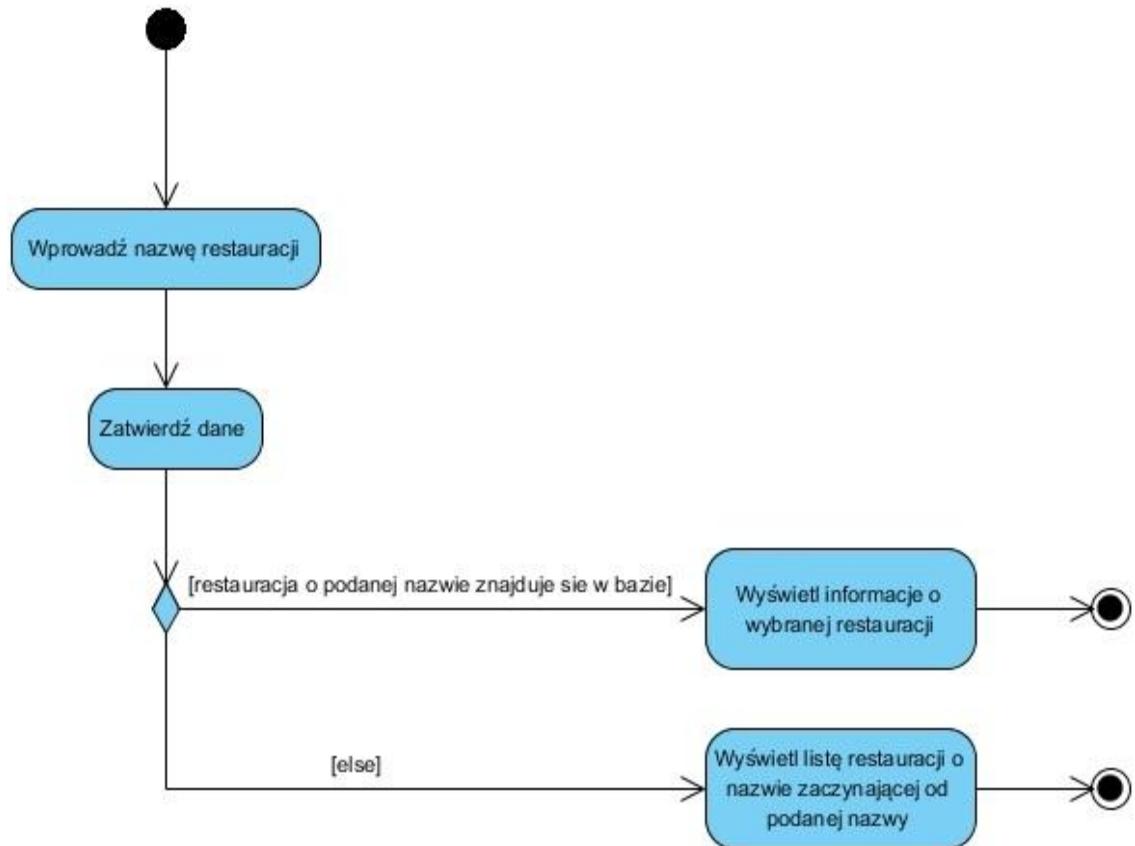
Rys. 3.2.3.5. Diagram czynności dla wystawienia komentarza (opracowanie własne).

6. Diagram czynności dla wyszukania miasta (Rys. 3.2.3.6).



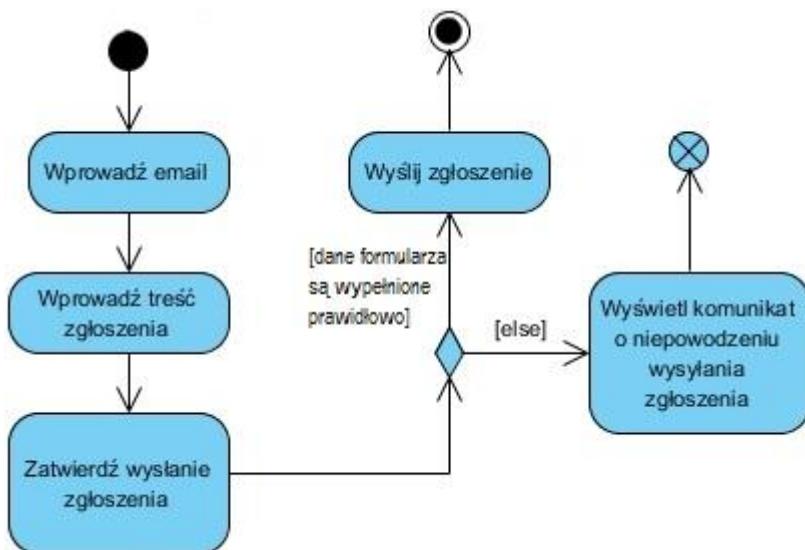
Rys. 3.2.3.6. Diagram czynności dla wyszukania miasta (opracowanie własne).

7. Diagram czynności dla wyszukania restauracji (Rys. 3.2.3.7).



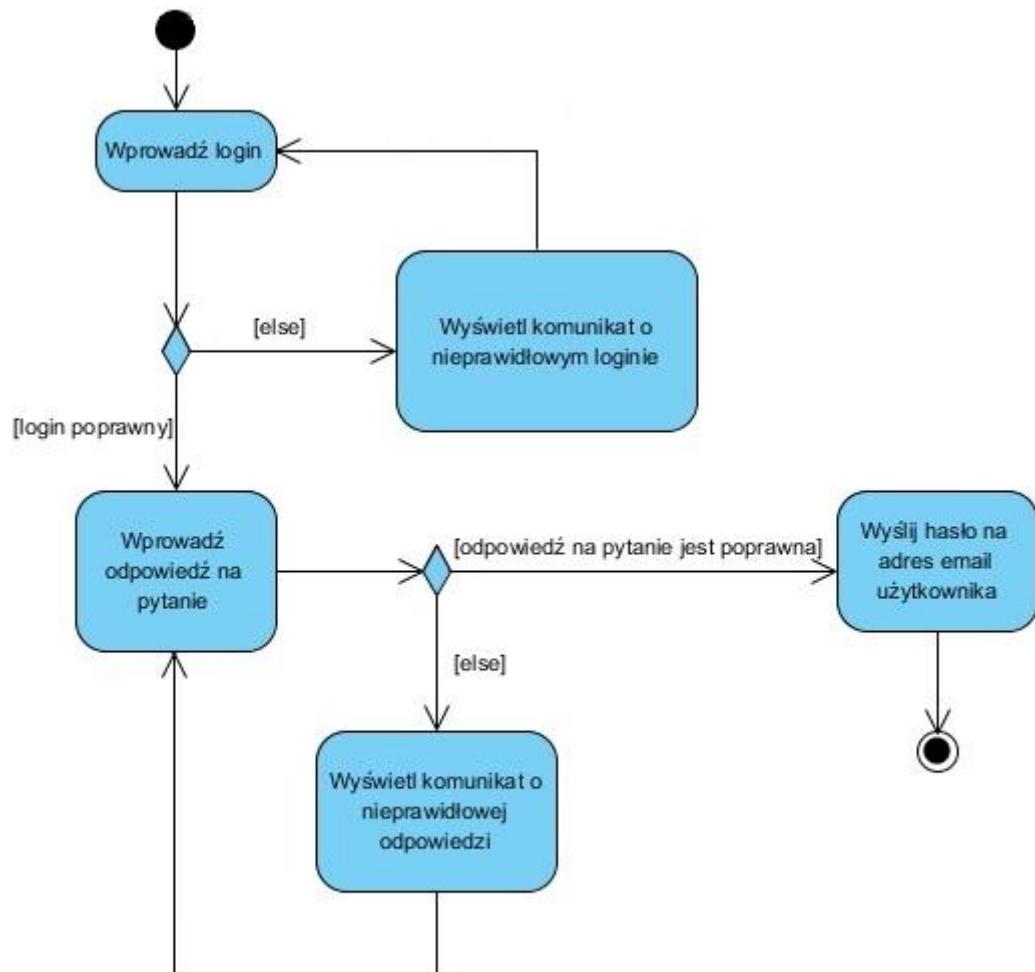
Rys. 3.2.3.7. Diagram czynności dla wyszukania restauracji (opracowanie własne).

8. Diagram czynności dla zgłoszania błędów (Rys. 3.2.3.8).



Rys. 3.2.3.8. Diagram czynności dla zgłoszania błędów (opracowanie własne).

9. Diagram czynności dla przywracania hasła (Rys. 3.2.3.9).



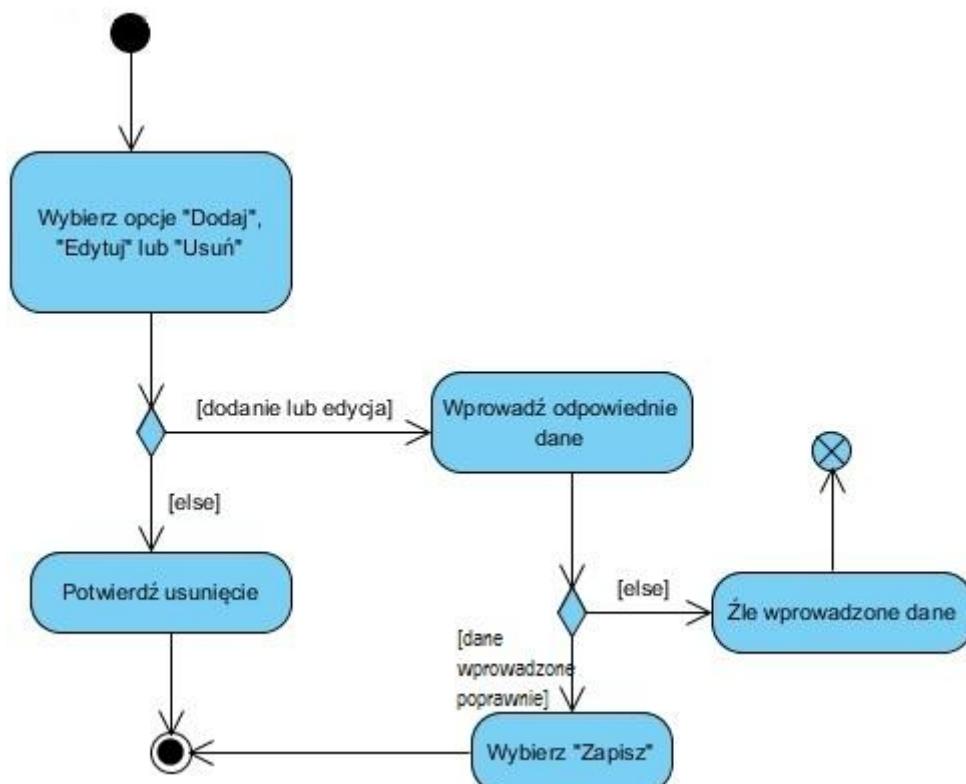
Rys. 3.2.3.9. Diagram czynności dla przywracania hasła (opracowanie własne).

10. Diagram czynności dla drukowania zamówienia (Rys. 3.2.3.10).



Rys. 3.2.3.10. Diagram czynności dla drukowania zamówienia (opracowanie własne).

11. Diagram czynności dla edycji bazy danych (Rys. 3.2.3.11).



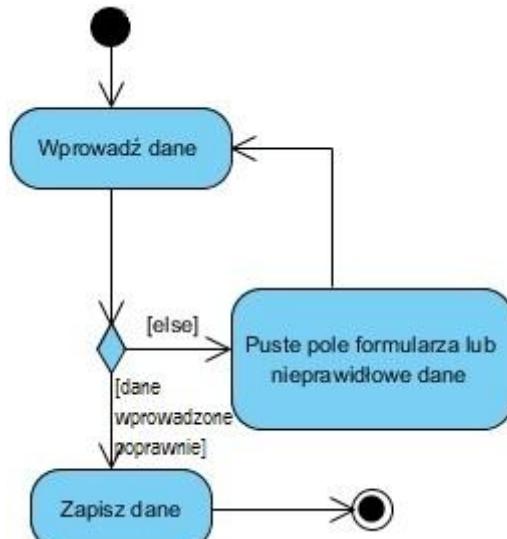
Rys. 3.2.3.11. Diagram czynności dla edycji bazy danych (opracowanie własne).

12. Diagram czynności dla edycji zawartości strony restauracji (Rys. 3.2.3.12).



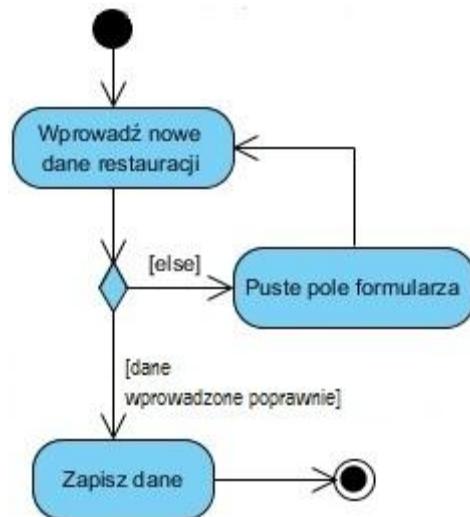
Rys. 3.2.3.12. Diagram czynności dla edycji zawartości strony restauracji (opracowanie własne).

13. Diagram czynności dla edycji danych użytkownika (Rys. 3.2.3.13).



Rys. 3.2.3.13. Diagram czynności dla edycji danych użytkownika (opracowanie własne).

14. Diagram czynności dla edycji danych restauracji (Rys. 3.2.3.14).



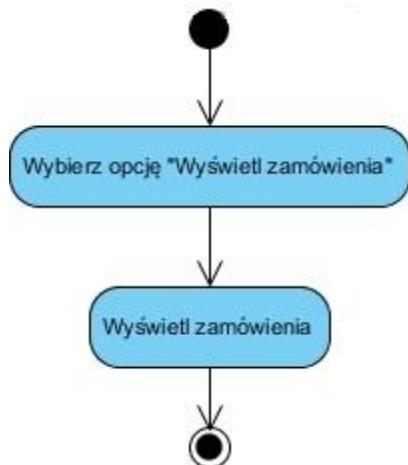
Rys. 3.2.3.14. Diagram czynności dla edycji danych restauracji (opracowanie własne).

15. Diagram czynności dla wyświetlania informacji o zamówieniu (Rys. 3.2.3.15).



Rys. 3.2.3.15. Diagram czynności dla wyświetlania informacji o zamówieniu (opracowanie własne).

16. Diagram czynności dla wyświetlania wszystkich zamówień (Rys. 3.2.3.16).



Rys. 3.2.3.16. Diagram czynności dla wyświetlania wszystkich zamówień (opracowanie własne).

17. Diagram czynności dla zmiany statusu zamówienia (Rys. 3.2.3.17).



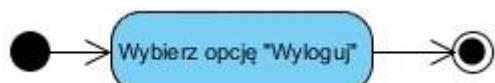
Rys. 3.2.3.17. Diagram czynności dla zmiany statusu zamówienia (opracowanie własne).

18. Diagram czynności dla drukowania rachunku (Rys. 3.2.3.18).



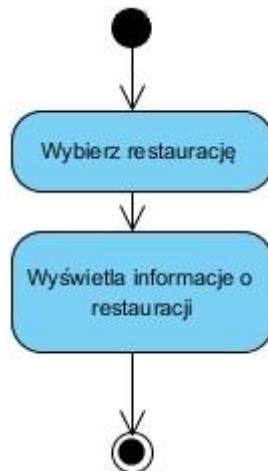
Rys. 3.2.3.18. Diagram czynności dla drukowania rachunku (opracowanie własne).

19. Diagram czynności dla wylogowania (Rys. 3.2.3.19).



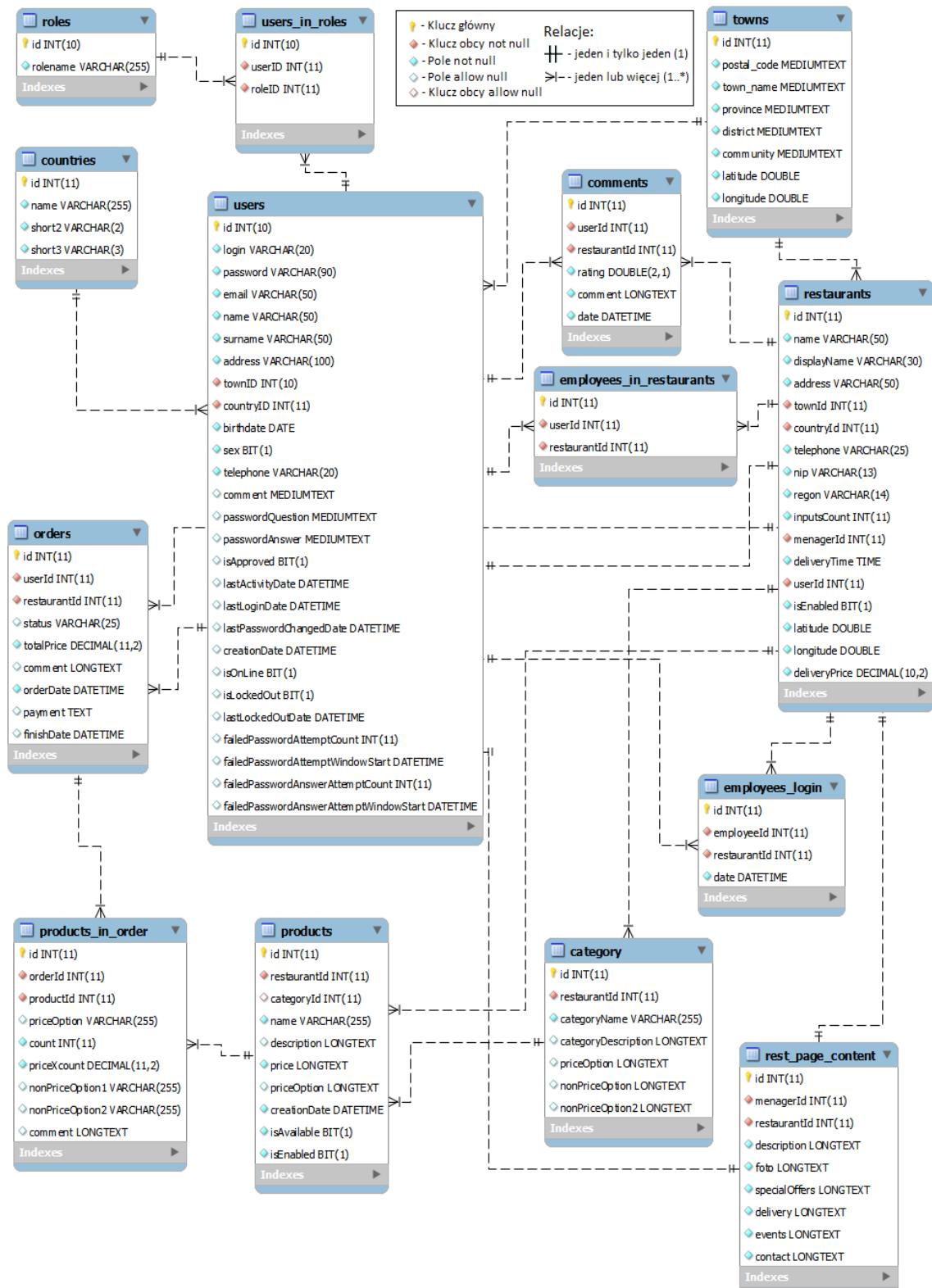
Rys. 3.2.3.19. Diagram czynności dla wylogowania (opracowanie własne).

20. Diagram czynności dla wyświetlania informacji o restauracji (Rys. 3.2.3.20).



Rys. 3.2.3.20. Diagram czynności dla wyświetlania informacji o restauracji (opracowanie własne).

3.2.4. Diagram bazy danych



Rys. 3.2.4.1. Diagram bazy danych (opracowanie własne).

Diagram (Rys. 3.2.4.1) ukazuje tabele w bazie danych połączone relacjami według legendy zamieszczonej w ramce. Każda tabela posiada klucz główny. Klucze obce służą do relacji pomiędzy tabelami. Zastosowaliśmy tutaj relacje 1:n oraz 1:1.

Kolumny w tabeli orders to id, userId, restaurantId, status, totalPrice, comment, orderdate, paymet oraz finishDate. Z diagramu można odczytać typy danych w każdej z tych kolumn. Kluczem głównym jest 'id', natomiast klucze obce to userId, który służy do stworzenia relacji 1:n z tabelą users oraz klucz restaurantId tworzący relacje 1:n z tabelą restaurants. Każde zamówienie musi posiadać adresata, dlatego występuje relacja z tabelą users oraz musi być przynależne do konkretnej restauracji, więc istnieje też relacja z tabelą restaurants. Pole status jest wykorzystywane do powiadamiania klienta o aktualnym statusie zamówienia. Pole totalPrice służy do przechowywania całkowitej ceny danego zamówienia. Comment to miejsce na ewentualny komentarz do zamówienia, który może wpisać klient. orderDate to data złożenia zamówienia, payment wskazuje na typ płatności oraz finishDate jest datą zakończenia zrealizowanego zamówienia.

Tabela roles zawiera nazwy ról, jest połączona z tabelą users_in_roles za pomocą relacji 1:n. Tabela users_in_roles zawiera klucze obce userID oraz roleID oraz jest połączona z tabelą users reprezentującą użytkowników aplikacji za pomocą relacji n:1.

Tabela countries zawiera nazwy państw oraz skróty ich nazw: dwu i trzyliterowe, łączy się z tabelą users za pomocą relacji 1:n.

Tabela towns zawiera nazwy miast, ich kody pocztowe, nazwy województwa, powiatu i gminy oraz współrzędne geograficzne. Jest połączona z tabelami users oraz restaurants za pomocą relacji 1:n.

Tabela comments zawiera oceny i treść komentarzy oraz daty ich wystawienia. Pola userId oraz restaurantId to klucze obce służące do połączenia z tabelami users oraz restaurants w relacjach 1:n.

Tabela users prezentuje zarówno użytkowników (klientów, menadżerów i pracowników), jak i restauracje. Zawiera loginy, hasła, dane osobowe takie jak: adres poczty elektronicznej, imię, nazwisko, adres, datę urodzenia, płeć i telefon oraz klucze obce wskazujące na miasta i kraje, komentarze, pytania i odpowiedzi do odzyskiwania hasła, daty ostatnich: logowań, aktywności oraz zmian hasła, a także daty utworzenia konta.

Tabela restaurants zawiera dane dotyczące restauracji takie jak: nazwa firmy, nazwa restauracji wyświetlana dla użytkowników, adres, klucze obce wskazujące na miasto i kraj, telefon, numer NIP i REGON, ilość wyświetleń strony restauracji, czas i cena dostawy oraz współrzędne geograficzne. Pole isEnabled przechowuje informację o

dostępności restauracji dla klientów. Klucz obcy managerId oraz userId wskazują na tabelę users.

Tabela employees_in_restaurants przypisuje pracowników do restauracji. Zawiera klucze obce userId oraz restaurantId, wskazujące na tabele users oraz restaurants w relacji 1:n.

Tabela employees_login przechowuje dane o ostatnim logowaniu pracownika do panelu restauracji. Zawiera datę logowania oraz klucze obce employeeId oraz restaurantId, wskazujące na tabele users oraz restaurants , w relacji 1:n.

Tabela rest_page_content zawiera treść wyświetlaną na stronie restauracji. Pola managerId oraz restaurantId to klucze obce służące do połączenia z tabelami users oraz restaurants w relacjach 1:n.

Tabela category zawiera dane dotyczące kategorii w menu restauracji. Pole restaurantId to klucz obcy łączący tabelę category z tabelą restaurants w relacji n:1.

W tabeli products przechowywane są dane dotyczące produktów oferowanych przez restauracje. Zawiera nazwy produktów, ich opisy, ceny oraz opcje cenowe (np. rozmiary produktów), a także daty utworzenia tych produktów. Pole isAvailable przechowuje informację o dostępności produktu dla klientów, natomiast pole isEnabled informuje czy produkt nie jest zablokowany, np. gdy dane kategorii do której był przypisany uległy zmianie, produkt może zawierać nieprawidłowe dane więc zostaje zablokowany do czasu edycji produktu przez menadżera restauracji. Pola restaurantId oraz categoryId to klucze obce łączące tabelę products z tabelami restaurants oraz category w relacji n:1.

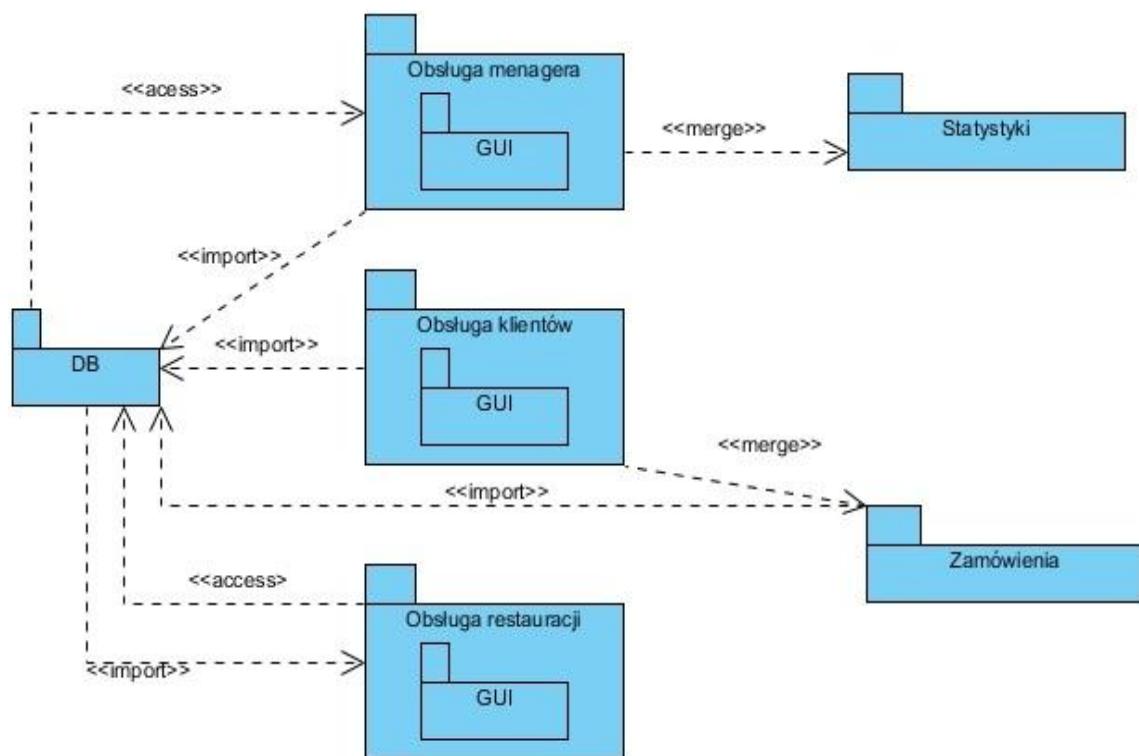
Tabela products_in_orders przypisuje produkty do zamówienia. Zawiera dane dotyczące produktu w danym zamówieniu takie jak: wybrana opcja cenowa, ilość, cena, wybrane inne opcje dotyczące produktu oraz komentarz do produktu. Pola orderId oraz productId to klucze obce łączące tabelę products_in_order z tabelami orders oraz products w relacji n:1.

Nasza baza danych zawiera informacje o zamówieniach, użytkownikach systemu, produktach i kategoriach w restauracjach, samych restauracjach oraz rolach przypisanych do kont użytkowników.

Pola w bazie, które nie zostały opisane powyżej powstały z myślą o funkcjonalnościach, które planujemy zaimplementować w przyszłości.

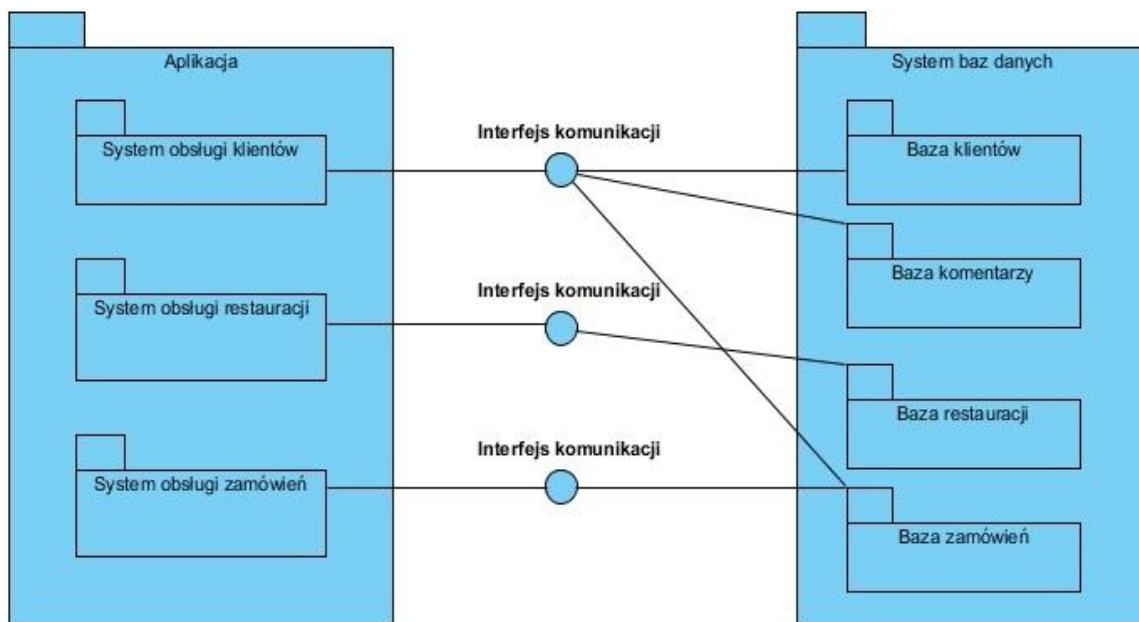
3.2.5. Inne

1. Diagram pakietów prezentujący pakiety oraz relacje między nimi (Rys. 3.2.5.1).



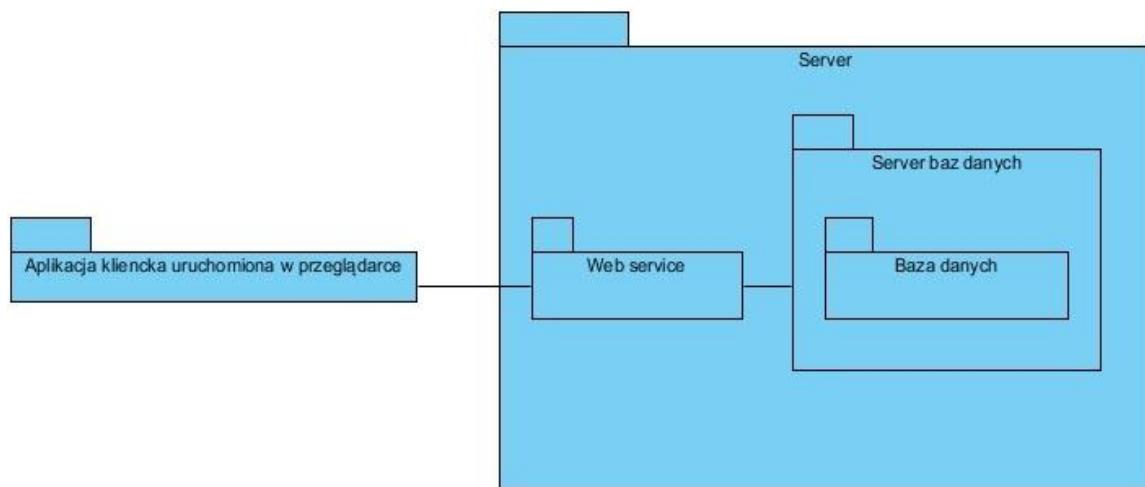
Rys. 3.2.5.1. Diagram pakietów (opracowanie własne).

2. Diagram komponentów przedstawiający fizyczne elementy systemu oraz połączenia między nimi. Komponenty są oznaczone dużymi prostokątami (Rys. 3.2.5.2).



Rys. 3.2.5.2. Diagram komponentów (opracowanie własne).

3. Diagram wdrożenia pokazuje rozmieszczenie fizycznych składników systemu oraz elementów wykonywujących system (tak zwanych węzłów) (Rys. 3.2.5.3).



Rys. 3.2.5.3. Diagram wdrożenia (opracowanie własne).

3.3. Wymagania aplikacji

Funkcje/usługi

KARTA WYMAGANIA			
Identyfikator:	F01	Priorytet:	Kluczowe
Nazwa:	Logowanie do systemu.		
Opis/uzasadnienie:	Użytkownik może się zalogować wpisując w pola tekstowe swój login oraz hasło. Użytkownik może za pomocą zaznaczenia pola jednokrotnego wyboru zapamiętać swój login i hasło oraz wykorzystać funkcję autologowania. Panel logowania powinien zmienić swój wizualny wygląd, gdy użytkownik zostanie zalogowany.		
Dane wejściowe:	Login i hasło użytkownika.		
Warunki początkowe:	Strona logowania jest otwarta i użytkownik nie jest zalogowany.		
Warunki końcowe:	Wyświetlenie odpowiedniej strony w zależności od praw dostępu.		
Sytuacje wyjątkowe:	Niewypełnione pola loginu/hasła: wyświetli informację o niewypełnieniu pola login lub hasło. Niepoprawny login lub hasło: wyświetli informację o niepoprawnym logowaniu. Błąd transferu danych: wyświetlenie komunikatu z informacją o niepowodzeniu operacji.		
Efekty uboczne:	Brak.		
Czynności równoczesne:	Aktualizacja daty ostatniego logowania oraz ostatniej aktywności w bazie danych.		
Stabilność:	Wymaganie nie będzie wymagało zmiany w przyszłości.		
Udziałowiec:	Zespół wdrażający system.		
Źródło:	Rozmowa wewnętrzna w zespole.		

Odpowiedzialny:	Gawroński Paweł / Kohlandt Dariusz
Wymagania powiązane:	F02 (rejestracja w systemie).

KARTA WYMAGANIA			
Identyfikator:	F02	Priorytet:	Kluczowe
Nazwa:	Rejestracja w systemie		
Opis\ uzasadnienie:	Użytkownik ma możliwość zarejestrowania się w systemie (założenia konta) poprzez wypełnienie wymaganych pól formularza rejestracyjnego oraz wybrania opcji „Załóż konto”, po wybraniu której nastąpi dodanie użytkownika do bazy danych. Po założeniu konta użytkownik zostanie zalogowany do systemu (wymaganie F01).		
Dane wejściowe:	Dane użytkownika wpisane w odpowiednich polach formularza rejestracyjnego.		
Warunki początkowe:	Strona rejestracji jest otwarta i użytkownik nie jest zalogowany.		
Warunki końcowe:	Wyświetlenie strony głównej, zalogowanie do systemu.		
Sytuacje wyjątkowe:	Niepoprawne wypełnienie pól formularza: wyświetl informację o niepoprawnym wypełnieniu pól; Błąd transferu danych: wyświetlenie komunikatu z informacją o niepowodzeniu operacji.		
Efekty uboczne:	Brak.		
Czynności równoczesne:	Logowanie do systemu.		
Stabilność:	Wymaganie nie będzie wymagało zmiany w przyszłości.		
Udziałowiec:	Zespół wdrażający system.		
Źródło:	Rozmowa wewnętrzna w zespole.		
Odpowiedzialny:	Gawroński Paweł / Kohlandt Dariusz		
Wymagania powiązane:	Brak.		

KARTA WYMAGANIA			
Identyfikator:	F03	Priorytet:	Pożądane
Nazwa:	Wyszukiwanie miasta lub restauracji.		
Opis/ uzasadnienie:	Użytkownik ma możliwość wyszukiwania interesującego go miasta lub restauracji poprzez wpisanie nazwy miasta lub restauracji w przeznaczone do tego celu pole oraz wybranie opcji „Szukaj”, po czym wyświetlona zostaje lista restauracji w podanym mieście lub restauracji mających w nazwie podany człon.		
Dane wejściowe:	Nazwa miasta lub restauracji wpisana w polu przeznaczonym do wyszukiwania.		
Warunki początkowe:	Strona początkowa jest otwarta.		
Warunki końcowe:	Wyświetlenie listy restauracji.		

Sytuacje wyjątkowe:	Puste pole wyszukiwania: wyświetl informację o niepoprawnym wypełnieniu pola; Brak elementów spełniających warunki wyszukiwania: wyświetl informację o braku wyników. Błąd transferu danych: wyświetlenie komunikatu z informacją o niepowodzeniu operacji.
Efekty uboczne:	Brak.
Czynności równoczesne:	Brak.
Stabilność:	W przyszłości może nastąpić potrzeba szybszego przeszukiwania zasobów.
Udziałowiec:	Klient restauracji.
Źródło:	Ankieta internetowa.
Odpowiedzialny:	Gawroński Paweł / Kohlandt Dariusz
Wymagania powiązane:	F04 (Wyświetlanie informacji o rejestracji).

KARTA WYMAGANIA	
Identyfikator:	F04
Priorytet:	Kluczowe
Nazwa:	Wyświetlanie informacji o restauracji.
Opis/ uzasadnienie:	Użytkownik ma możliwość wybrania interesującej go restauracji i zobaczenia informacji o restauracji. Możliwe będzie wybieranie różnych zakładek np. „menu”, „dowóz” itp. Po wybraniu jednej z powyższych opcji zmieniają się wyświetlane informacje.
Dane wejściowe:	Wybrana restauracja.
Warunki początkowe:	Strona restauracji jest otwarta.
Warunki końcowe:	Wyświetlenie informacji o restauracji.
Sytuacje wyjątkowe:	Brak.
Efekty uboczne:	Brak.
Czynności równoczesne:	Brak.
Stabilność:	Wymaganie nie będzie wymagało zmiany w przyszłości.
Udziałowiec:	Zespół wdrażający system.
Źródło:	Rozmowa wewnętrzna w zespole.
Odpowiedzialny:	Gawroński Paweł / Kohlandt Dariusz
Wymagania powiązane:	F03 (Wyszukiwanie miasta lub restauracji.).

KARTA WYMAGANIA	
Identyfikator:	F05
Priorytet:	Kluczowe
Nazwa:	Wylogowanie z systemu.

Opis/ uzasadnienie:	Użytkownik może się wylogować z systemu poprzez wybranie opcji „Wyloguj”. Po wybraniu opcji „Wyloguj” wyświetlona zostanie strona początkowa, użytkownik traci możliwość wykonywania zamówień aż do ponownego zalogowania się w systemie.
Dane wejściowe:	Brak.
Warunki początkowe:	Brak.
Warunki końcowe:	Użytkownik jest wylogowany
Sytuacje wyjątkowe:	Brak.
Efekty uboczne:	Brak.
Czynności równoczesne:	Brak.
Stabilność:	Wymaganie nie będzie wymagało zmiany w przyszłości.
Udziałowiec:	Zespół wdrażający system.
Źródło:	Rozmowa wewnętrzna w zespole.
Odpowiedzialny:	Gawroński Paweł / Kohlandt Dariusz
Wymagania powiązane:	F01 (Logowanie do systemu.)

KARTA WYMAGANIA		
Identyfikator:	F06	Priorytet: Kluczowe
Nazwa:	Złożenie zamówienia.	
Opis/ uzasadnienie:	Użytkownik ma możliwość zrealizowania zamówienia poprzez wybranie opcji „Realizuj” w panelu koszyka. Po złożeniu zamówienia użytkownik musi potwierdzić chęć realizacji, po czym zamówienie zostaje przesłane do restauracji.	
Dane wejściowe:	Informacje o zamówieniu.	
Warunki początkowe:	Panel koszyka jest otwarty.	
Warunki końcowe:	Zamówienie zostaje przesłane do restauracji.	
Sytuacje wyjątkowe:	Błąd podczas przesyłania zamówienia: ponów próbę wysłania zamówienia. Błąd transferu danych: wyświetlenie komunikatu z informacją o niepowodzeniu operacji.	
Efekty uboczne:	Brak.	
Czynności równoczesne:	Wyświetlenie zamówienia w panelu restauracji.	
Stabilność:	W przyszłości może nastąpić potrzeba szybszego wysyłania zamówień.	
Udziałowiec:	Zespół wdrażający system.	
Źródło:	Rozmowa wewnętrzna w zespole.	
Odpowiedzialny:	Gawroński Paweł / Kohlandt Dariusz	
Wymagania powiązane:	F08 (Wyświetlanie przychodzących zamówień.)	

KARTA WYMAGANIA			
Identyfikator:	F07	Priorytet:	Pożądane
Nazwa:	Wystawianie komentarzy.		
Opis/ uzasadnienie:	Po zrealizowaniu zamówienia użytkownik ma możliwość wystawiania komentarzy poprzez wypełnienie formularza komentarzy oraz wybranie opcji „Wystaw komentarz”. Po wystawieniu komentarza pojawi się on w informacji dotyczącej restauracji.		
Dane wejściowe:	Dane komentarza.		
Warunki początkowe:	Możliwość wystawienia komentarza jest dostępna.		
Warunki końcowe:	Komentarz jest widoczny w informacji o restauracji.		
Sytuacje wyjątkowe:	Niepoprawne wypełnienie pól formularza: wyświetl informację o niepoprawnym wypełnieniu pól; Błąd transferu danych: wyświetlenie komunikatu z informacją o niepowodzeniu operacji.		
Efekty uboczne:	Brak.		
Czynności równoczesne:	Brak.		
Stabilność:	Wymaganie nie będzie wymagało zmiany w przyszłości.		
Udziałowiec:	Klient restauracji.		
Źródło:	Ankieta internetowa.		
Odpowiedzialny:	Gawroński Paweł / Kohlandt Dariusz		
Wymagania powiązane:	Brak.		

KARTA WYMAGANIA			
Identyfikator:	F08	Priorytet:	Kluczowe
Nazwa:	Wyświetlanie przychodzących zamówień.		
Opis/ uzasadnienie:	W panelu restauracji wyświetlają się przychodzące zamówienia. W zamówieniu zostaną wyświetcone zamówione dania, kwota zamówienia oraz dane osobowe klienta. Każde nowe zamówienie ma nadany aktualny status “oczekujące”.		
Dane wejściowe:	Informacje o zamówieniach.		
Warunki początkowe:	Strona restauracji musi być otwarta, użytkownik panelu restauracji musi być zalogowany na swoje konto oraz status restauracji powinien być ustawiony na online.		
Warunki końcowe:	Zamówienie musi zostać wyświetlone na stronie głównej restauracji.		
Sytuacje wyjątkowe:	Brak wyświetlenia się nowych zamówień: odświeżenie strony.		
Efekty uboczne:	Brak.		
Czynności równoczesne:	Zmiana statusu zamówienia przez użytkownika panelu restauracji.		
Stabilność:	Wymaganie nie wymaga zmiany w przyszłości.		
Udziałowiec:	Menadżer restauracji.		

Źródło:	Rozmowa z udziałowcem.
Odpowiedzialny:	Gawroński Paweł / Kohlandt Dariusz
Wymagania powiązane:	F09 (Zmiana statusu zamówienia).

KARTA WYMAGANIA			
Identyfikator:	F09	Priorytet:	Pożądane
Nazwa:	Zmiana statusu przychodzących zamówień.		
Opis/ uzasadnienie:	W panelu restauracji użytkownik ma możliwość nadania odpowiedniego statusu dla każdego zamówienia. Dostępne stany to: oczekujące, przyjęto do realizacji, wysłano. Przy zmianie statusu zmienia się on również u klienta.		
Dane wejściowe:	Nowe zamówienie.		
Warunki początkowe:	Strona restauracji musi być otwarta, użytkownik panelu restauracji musi być zalogowany na swoje konto oraz status restauracji powinien być ustawiony na online.		
Warunki końcowe:	Zmiana statusu zamówienia skutkuje odświeżeniem tego zamówienia w panelu.		
Sytuacje wyjątkowe:	Brak wyświetlenia się nowych zamówień: odświeżenie strony. Zmiana statusu zamówienia nie zostanie wykonana prawidłowo przez system: odświeżenie zamówienia. Błąd transferu danych z nowym statusem zamówienia: dane będą przesyłane w określonych odstępach czasu, aż do momentu, gdy zostaną poprawnie odebrane przez system i odesłane do panelu.		
Efekty uboczne:	Brak.		
Czynności równoczesne:	Wyświetlanie zamówień w panelu użytkownika restauracji.		
Stabilność:	Wymaganie nie wymaga zmiany w przyszłości.		
Udziałowiec:	Menadżer restauracji.		
Źródło:	Rozmowa z udziałowcem.		
Odpowiedzialny:	Gawroński Paweł / Kohlandt Dariusz		
Wymagania powiązane:	F08 (Wyświetlanie przychodzących zamówień).		

KARTA WYMAGANIA			
Identyfikator:	F10	Priorytet:	Kluczowe
Nazwa:	Dodanie nowej restauracji.		
Opis/ uzasadnienie:	Menadżer może dodać nową restaurację do systemu poprzez dostępną opcję w panelu menadżera. Po poprawnym wypełnieniu formularza oraz wybraniu opcji „dodaj”, stworzone zostaje osobne konto z uprawnieniami restauracji w bazie danych.		
Dane wejściowe:	Szczegółowe dane restauracji.		
Warunki początkowe:	Użytkownik po zalogowaniu musitrzymać prawa menadżera oraz przejść do odpowiedniego formularza.		
Warunki końcowe:	Nowe konto dla restauracji musi zostać prawidłowo zapisane w bazie danych.		

Sytuacje wyjątkowe:	Błąd transferu danych: wyświetlenie komunikatu z informacją o niepowodzeniu operacji.
Efekty uboczne:	Brak.
Czynności równoczesne:	Brak.
Stabilność:	Wymaganie nie wymaga zmiany w przyszłości.
Udziałowiec:	Zespół wdrażający system.
Źródło:	Rozmowa wewnętrzna w zespole.
Odpowiedzialny:	Gawroński Paweł / Kohlandt Dariusz
Wymagania powiązane:	Brak.

KARTA WYMAGANIA			
Identyfikator:	F11	Priorytet:	Kluczowe
Nazwa:	Edycja bazy danych.		
Opis/ uzasadnienie:	Administrator systemu może edytować każdą komórkę oraz jej typ w bazie danych za pomocą narzędzia phpMyAdmin zawartego w panelu administratora.		
Dane wejściowe:	Odpowiednie dane z bazy danych.		
Warunki początkowe:	Użytkownik po zalogowaniu musi otrzymać prawa administratora oraz przejść do panelu phpMyAdmin.		
Warunki końcowe:	Nowe wartości oraz typy danych wpisane do komórek powinny zostać przesłane do bazy danych oraz powinno wyświetlić się pozytywne potwierdzenie wykonania operacji.		
Sytuacje wyjątkowe:	Błąd transferu danych: wyświetlenie komunikatu z informacją o niepowodzeniu operacji. Przepelnienie bazy danych: wyświetlenie komunikatu technicznego z dokładnym kodem błędu na dodatkowym panelu. Brak wypełnionych wymaganych komórek: informacja o tym czy komórka może pozostać pusta. Dane niedopasowane do typu danych w komórce: wyświetlenie informacji o typie danych w danej komórce.		
Efekty uboczne:	Brak.		
Czynności równoczesne:	Brak.		
Stabilność:	Możliwe zmiany w przyszłości.		
Udziałowiec:	Zespół wdrażający system.		
Źródło:	Rozmowa wewnętrzna w zespole.		
Odpowiedzialny:	Gawroński Paweł / Kohlandt Dariusz		
Wymagania powiązane:	Brak.		

KARTA WYMAGANIA			
Identyfikator:	F12	Priorytet:	Kluczowe
Nazwa:	Edycja strony restauracji.		

Opis/ uzasadnienie:	Menadżer restauracji może edytować zawartość strony restauracji, czyli stronę główną restauracji, menu, informacje o dowozie oraz imprezach okolicznościowych, kontakt, a także galerię zdjęć.
Dane wejściowe:	Zawartość strony oraz zdjęcia.
Warunki początkowe:	Użytkownik po zalogowaniu musi otrzymać prawa menadżera oraz przejść do panelu zarządzania restauracją.
Warunki końcowe:	Dane o zawartości strony powinny zostać przesłane do bazy danych oraz powinno przekierować do strony z podglądem strony restauracji.
Sytuacje wyjątkowe:	Błąd transferu danych: wyświetlenie komunikatu z informacją o niepowodzeniu operacji.
Efekty uboczne:	Brak.
Czynności równoczesne:	Brak.
Stabilność:	Możliwe zmiany w przyszłości.
Udziałowiec:	Zespół wdrażający system.
Źródło:	Rozmowa wewnętrzna w zespole.
Odpowiedzialny:	Gawroński Paweł / Kohlandt Dariusz
Wymagania powiązane:	Brak.

Wymagania niefunkcjonalne.

KARTA WYMAGANIA			
Identyfikator:	N01	Priorytet:	Pożądane
Nazwa:	Łatwość użycia.		
Opis/ uzasadnienie:	System ma być prosty w obsłudze. Powinien być intuicyjny, a nauka jego użytkowania ma zatrwać się w 30 minutowym kursie.		
Sytuacje wyjątkowe:	Popełnienie błędu przez użytkownika systemu: możliwość późniejszej korekty przez użytkownika. Popełnienie większej ilości błędów niż jeden dziennie: wymaganie autoryzacji administratora, następnie korekta.		
Efekty uboczne:	W przypadku wystąpienia znacznej ilości błędów: zauważalne lekkie spowolnienie odpowiedzi na zapytania.		
Stabilność:	Wymaganie nie będzie wymagało zmiany w przyszłości.		
Udziałowiec:	Zespół wdrażający system.		
Źródło:	Rozmowa wewnętrzna w zespole.		
Odpowiedzialny:	Gawroński Paweł / Kohlandt Dariusz		

Wymagania powiązane:	Brak.		

KARTA WYMAGANIA			
Identyfikator:	N02	Priorytet:	Kluczowe
Nazwa:	Bezawaryjność.		
Opis/ uzasadnienie:	Pojawiające się błędy powinny być natychmiastowo obsłużone/naprawione. Średnia ilość nieobsłużonych błędów systemu nie powinna przekraczać 2 tygodniowo.		
Sytuacje wyjątkowe:	Pojawienie się błędu: natychmiastowa korekta. Pojawienie się nieobsługiwanego błędu: wymagana korekta bazy błędów. Zatrzymanie pracy: autorestart aplikacji.		
Efekty uboczne:	W przypadku wystąpienia znacznej ilości błędów zauważalne lekkie spowolnienie działania aplikacji.		
Stabilność:	Możliwe zmiany w przyszłości.		
Udziałowiec:	Zespół wdrażający system.		
Źródło:	Rozmowa wewnętrzna w zespole.		
Odpowiedzialny:	Gawroński Paweł / Kohlandt Dariusz		
Wymagania powiązane:	N03 (Szybkość działania).		

KARTA WYMAGANIA			
Identyfikator:	N03	Priorytet:	Pożądane
Nazwa:	Niski czas oczekiwania na dane z bazy danych.		
Opis/ uzasadnienie:	Szybkość ładowania danych z bazy danych przez aplikację powinna być względnie wysoka. Średni czas oczekiwania na odpowiedź z bazy nie powinien przekroczyć 3 sekund.		
Sytuacje	Brak dostępu do bazy danych: wyświetlenie		

wyjątkowe:	komunikatu o tymczasowym braku dostępu do bazy danych. Zerwanie połączenia z bazą danych: wyświetlenie komunikatu o błędzie połączenia.		
Efekty uboczne:	W przypadku błędów: spowolnienie działania aplikacji.		
Stabilność:	Możliwe zmiany w przyszłości.		
Udziałowiec:	Zespół wdrażający system.		
Źródło:	Rozmowa wewnętrzna w zespole.		
Odpowiedzialny:	Gawroński Paweł / Kohlandt Dariusz		
Wymagania powiązane:	N02 (Bezawaryjność).		

KARTA WYMAGANIA			
Identyfikator:	N04	Priorytet:	Pożądane
Nazwa:	Szybkość przetwarzania danych.		
Opis/ uzasadnienie:	W zależności od łącza internetowego, czas odświeżenia strony i przetworzenia danych powinien być mniejszy niż 10 sekund.		
Sytuacje wyjątkowe:	W przypadku powolnego łącza internetowego: dłuższy okres oczekiwania na odpowiedź z serwera.		
Efekty uboczne:	Brak.		
Stabilność:	Możliwe zmiany w przyszłości.		
Udziałowiec:	Zespół wdrażający system.		
Źródło:	Rozmowa wewnętrzna w zespole.		
Odpowiedzialny:	Gawroński Paweł / Kohlandt Dariusz		
Wymagania powiązane:	Brak.		

KARTA WYMAGANIA			
Identyfikator:	N05	Priorytet:	Kluczowe
Nazwa:	Automatyczny restart systemu.		
Opis/	Podczas krytycznego błędu w systemie czas		

uzasadnienie:	ponownego uruchomienia po awarii nie powinien przekroczyć czasu uruchamiania maszyny serwerowej (maksymalnie 1-2 min), na której ten system pracuje (w wypadku awarii zasilania), w wypadku błędu krytycznego aplikacji maksymalny czas oczekiwania to 1-2 min.		
Sytuacje wyjątkowe:	Brak połączenia z maszyną serwerową: ponawianie próby połączenia do czasu ponownego uruchomienia systemu.		
Efekty uboczne:	Brak.		
Stabilność:	Możliwe zmiany wymagania w przyszłości		
Udzialowiec:	Zespół wdrażający system.		
Źródło:	Rozmowa wewnętrzna w zespole.		
Odpowiedzialny:	Gawroński Paweł / Kohlandt Dariusz		
Wymagania powiązane:	N02 (Bezawaryjność).		

KARTA WYMAGANIA			
Identyfikator:	N06	Priorytet:	Pożądane
Nazwa:	Wieloplatformowość systemu.		
Opis/uzasadnienie:	System będzie przenośny. Wykorzystana do jego stworzenia technologia pozwalać będzie na uruchomienie go na wielu różnych urządzeniach (również mobilnych) niezależnie od platformy.		
Sytuacje wyjątkowe:	Nieobsługiwane urządzenie/platforma: wyświetlenie komunikatu o braku kompatybilności z daną platformą oraz braku możliwości wyświetlania usługi.		
Efekty uboczne:	Brak.		
Stabilność:	Przewidywana zmiana wymagania w przyszłości w celu dotarcia systemu do jak największej ilości urządzeń.		
Udzialowiec:	Zespół wdrażający system.		
Źródło:	Rozmowa wewnętrzna w zespole.		
Odpowiedzialny:	Gawroński Paweł / Kohlandt Dariusz		
Wymagania	N02 (Bezawaryjność).		

powiązane:	N04 (Szybkość przetwarzania danych).		
-------------------	--------------------------------------	--	--

Wymagania dziedzinowe

D1. Ze względu na ochronę interesu użytkowników w momencie, gdy klient złoży zamówienie wybierając środek płatności inny niż płatność przy odbiorze, aplikacja powinna wyświetlić potwierdzenie dokonania wpłaty na konto restauracji. Jedynie w sytuacji, gdy płatność została potwierdzona, przez obsługiwany system płatności, zamówienie zostaje wysłane do restauracji w celu realizacji. W przypadku gdy obsługiwany system płatności nie potwierdzi dokonania płatności aplikacja powinna wyświetlić informację o nieudanej próbie dokonania płatności wraz z instrukcją postępowania w celu zwrotu poniesionych kosztów.

Wymagania na środowisko docelowe

W środowisku, w którym aplikacja będzie działać nie są niezbędne specyficzne wymagania.

Wymagania dotyczące procesu wytwarzania

Działający system musi zostać stworzony do lutego 2013 roku, przy założeniu kaskadowej metody wytwarzania.

Kryteria akceptacji rozwiązania

Kluczowe wymagania dla warstwy klienta restauracji:

- możliwość rejestracji;
- możliwość logowania;
- możliwość wyboru dań/restauracji;
- możliwość złożenia zamówienia;
- otrzymanie potwierdzenia przyjęcia zamówienia.

Kluczowe wymagania dla warstwy menadżera restauracji:

- możliwość rejestracji;
- możliwość logowania;
- możliwość dodania nowej restauracji;

- możliwość edycji danych restauracji;
- możliwość edycji strony restauracji;
- możliwość dodawania dań oraz kategorii;

Kluczowe wymagania dla warstwy restauracji:

- możliwość logowania;
- możliwość przyjęcia zamówienia;
- możliwość zmiany statusu zamówienia;

Kluczowe wymagania dla warstwy administratora systemu:

- możliwość edycji danych restauracji i klientów.

4. PROCES WYTWARZANIA APLIKACJI

4.1. Opis głównych funkcjonalności

4.1.1. Logowanie (Paweł Gawroński)

Logowanie do systemu odbywa się za pomocą formularza zawierającego pola tekstowe pozwalające na wprowadzenie loginu i hasła użytkownika oraz przycisk wyboru pozwalający na zapamiętanie użytkownika przez przeglądarkę internetową. Pola loginu i hasła muszą być wypełnione. Logowanie odbywa się po kliknięciu przycisku „Zaloguj”. Sprawdzanie poprawności danych logowania jest przeprowadzane za pomocą metody ValidateUser z klasy CustomMembershipProvider, która pobiera hasło z bazy, porównuje je z hasłem podanym przez użytkownika odpowiednio zahashowanym lub zaszyfrowanym zależnie od ustawień aplikacji. Jeżeli dane logowania są poprawne zaktualizowana zostaje data ostatniego logowania użytkownika oraz jeśli użytkownik nie jest przypisany do roli „Pracownik” dane zostają zapisane do plików cookies przeglądarki. Następnie użytkownik zostaje przekierowany do strony głównej warstwy aplikacji, na którą wskazuje przypisana mu rola, co zostało zaprezentowane w fragmencie kodu (Kod 4.1.1.1.).

```
if (ModelState.IsValid)
{
    CustomMembershipProvider cmp = new CustomMembershipProvider();
    if (cmp.ValidateUser(model.Login, model.Password))
    {
        CustomRoleProvider role =
        (CustomRoleProvider)System.Web.Security.Roles.Providers["CustomRoleProvider"];
        if (role.IsUserInRole(model.Login, "Menadżer"))
        {
            FormsAuthentication.SetAuthCookie(model.Login, model.RememberMe);
            return RedirectToAction("Index", "ManagePanel");
        }
        elseif (role.IsUserInRole(model.Login, "Admin"))
        {
            FormsAuthentication.SetAuthCookie(model.Login, model.RememberMe);
            return RedirectToAction("Index", "Admin");
        }
        elseif (role.IsUserInRole(model.Login, "Restauracja"))
        {
            FormsAuthentication.SetAuthCookie(model.Login, model.RememberMe);
            return RedirectToAction("Locked", "POS");
        }
        elseif (role.IsUserInRole(model.Login, "Pracownik"))
```

```

{
ModelState.AddModelError("", "Zaloguj się najpierw na konto
restauracji.");
}
else
{
FormsAuthentication.SetAuthCookie(model.Login, model.RememberMe);
if (Url.IsLocalUrl(returnUrl) && returnUrl.Length> 1
&& returnUrl.StartsWith("/")
&& !returnUrl.StartsWith("//") && !returnUrl.StartsWith("//\\"))
{
return Redirect(returnUrl);
}
else
{
returnRedirectToAction("Index", "Home");
}
}
else
{
ModelState.AddModelError("", "");
}
}
}

```

Kod 4.1.1.1. Logowanie użytkownika oraz przekierowanie do odpowiadającej roli stronie.

4.1.2. Rejestracja (Paweł Gawroński)

Dodawanie konta nowego użytkownika (klienta, menadżera lub pracownika) odbywa się za pomocą formularza zawierającego pola tekstowe pozwalające na wprowadzenie loginu, adresu poczty elektronicznej, hasła, pytania do przywracania hasła oraz odpowiedzi na to pytanie, imienia, nazwiska, adresu, nazwy miasta oraz kodu pocztowego, nazwy kraju, daty urodzenia, płci oraz numeru telefonu. Wszystkie pola muszą być wypełnione. Dodatkowo pola „Email” i „Powtórz email” oraz „Hasło” i „Powtórz hasło” muszą zawierać jednakową treść. Data urodzenia musi wykazywać pełnoletniość użytkownika, a nazwę kraju należy wybrać z listy rozwijanej. Zakładanie nowego konta odbywa się po kliknięciu przycisku „Załóż konto”. Pobrała zostaje lista miast zgodnych z nazwą miasta oraz kodem pocztowym, jeśli nie znaleziono miast spełniających kryteria, użytkownik zostaje poproszony o poprawne wpisanie danych miasta. Jeśli znaleziono więcej niż jedno miasto, użytkownik zostaje poproszony o poprawienie danych lub wybranie miasta z dołączonej do formularza mapki (Rys. 4.1.2.1)



Rys. 4.1.2.1. Mapa dopasowanych miast oraz kodów pocztowych (opracowanie własne).

Jeśli znaleziono tylko jedno miasto, a pozostałe dane są wypełnione poprawnie zostaje wywołana metoda `CreateUser` z klasy `CustomMembershipProvider`, która sprawdza czy login oraz adres poczty elektronicznej nie są przypisane do innego użytkownika oraz czy hasło spełnia kryteria ustalone w ustawieniach aplikacji dotyczące, między innymi, wymaganej ilości znaków i wymaganej ilości cyfr w haśle (Kod 4.1.2.1.).

```
ValidateEventArgs args =
new ValidateEventArgs(login, password, true);

OnValidatingPassword(args);

if (args.Cancel)
{
    status = MembershipCreateStatus.InvalidPassword;
    return null;
}

if (RequiresUniqueEmail &&
! (String.IsNullOrEmpty(GetUserNameByEmail(email))))
{
    status = MembershipCreateStatus.DuplicateEmail;
    return null;
}
```

Kod 4.1.2.1. Walidacja loginu, hasła oraz adresu poczty elektronicznej.

Jeżeli nie zostaną wykryte nieprawidłowości w podanych danych, hasło oraz odpowiedź do przywracania hasła zostają zaszyfrowane lub zahashowane, a nowy użytkownik zostaje dodany do systemu poprzez zapisanie jego danych w bazie, po czym zostaje mu przypisana odpowiednia rola (Kod 4.1.2.2.).

```
CustomRoleProvider role =
```

```
(CustomRoleProvider) System.Web.Security.Roles.Providers["CustomRoleProvider"];
role.AddUsersToRoles(newstring[] { user.Login }, newstring[] { "Klient" });
}
```

Kod 4.1.2.2. Przypisanie użytkownika do roli.

Proces rejestracji kończy się zalogowaniem stworzonego użytkownika do systemu.

4.1.3. Odzyskiwanie hasła (Paweł Gawroński)

Jeśli odzyskiwanie zapomnianego hasła jest możliwe, czyli opcja enablePasswordReset jest ustawiona na `true` w pliku ustawień aplikacji, to odzyskiwanie hasła odbywa się poprzez podanie loginu do swojego konta i kliknięcie przycisku „Dalej”. Następnie pobierane jest pytanie do odzyskiwania hasła oraz informacja, czy konto użytkownika nie jest zablokowane. Jeśli użytkownik o podanym loginie istnieje i nie jest zablokowany, wyświetlane zostaje pytanie podane podczas rejestracji w systemie oraz pole tekstowe przeznaczone na odpowiedź na pytanie. Po wprowadzeniu odpowiedzi na pytanie do odzyskiwania hasła oraz kliknięciu przycisku „Dalej”, wywołana zostaje metoda `ResetPassword` z klasy `CustomMembershipProvider`, która generuje nowe hasło (Kod 4.1.3.1).

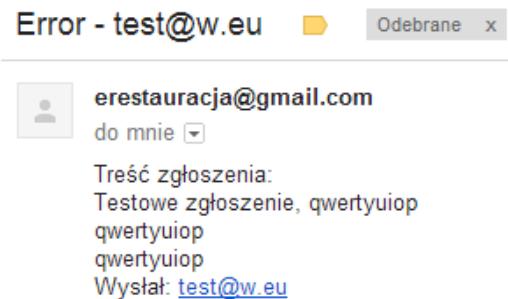
```
string newPassword =
System.Web.Security.Membership.GeneratePassword(newPasswordLength,
MinRequiredNonAlphanumericCharacters);
```

Kod 4.1.3.1. Generowanie nowego hasła.

Następnie sprawdzana jest zgodność udzielonej odpowiedzi z odpowiedzią podaną podczas rejestracji w systemie. Jeśli odpowiedzi są takie same, nowe hasło jest szyfrowane lub hashowane, zapisywane w bazie danych i wysypane na adres poczty elektronicznej użytkownika.

4.1.4. Zgłaszanie błędów (Paweł Gawroński)

Zgłaszanie błędów odbywa się poprzez wypełnienie formularza zawierającego pole tekstowe przeznaczone na treść zgłoszenia oraz pole tekstowe przeznaczone na adres poczty elektronicznej zgłaszającego. W przypadku, gdy zgłaszającym błęd jest osoba zalogowana w systemie, pole przeznaczone na adres poczty elektronicznej nie jest wyświetlane. Jeśli powyższe pola są wypełnione, zgłoszenie zostaje wysłane na adres `erestauracja.bledy@gmail.com` (Rys. 4.1.4.1).



Rys. 4.1.4.1. Wiadomość email z treścią zgłoszenia (opracowanie własne).

4.1.5. Pomoc (Paweł Gawroński)

Po Wybraniu opcji „Pomoc” zostaje wyświetlona strona zawierająca instrukcję obsługi dla klienta lub menadżera, jeśli opcję wybrał użytkownik zalogowany na konto menadżera.

```
<script type="text/javascript">
    $(document).ready(function () {
        window.onload = function () {
            document.getElementById("iframepdf").src =
"../../Content/Manual-klient.pdf";
        }
    });
</script>
```

Kod 4.1.5.1. Ładowanie dokumentu pdf do atrybutu src elementu iframe.

Instrukcję można zapisać na dysk swojego komputera, wydrukować lub przeglądać na stronie aplikacji. Dostępne są również opcje (Rys. 4.1.5.1) takie jak zmiana strony, powiększanie, pomniejszanie oraz przejście do panelu zaawansowanego (Rys. 4.1.5.2), w którym istnieje możliwość konwertowania instrukcji do innych formatów, wysłanie instrukcji, jako załącznik wiadomości elektronicznej oraz dopasowywanie rozmiaru dokumentu do wielkości okna, w którym jest wyświetlany. Opcje różnią się w zależności od zainstalowanego dodatku do przeglądarki odpowiedzialnego za odczyt plików pdf.



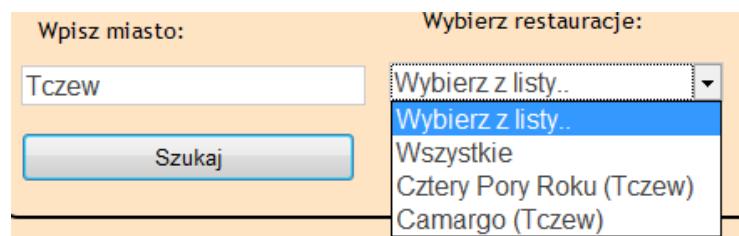
Rys. 4.1.5.1. Podstawowy panel zarządzania dokumentem pdf (opracowanie własne).



Rys. 4.1.5.2. Zaawansowany panel zarządzania dokumentem pdf (opracowanie własne).

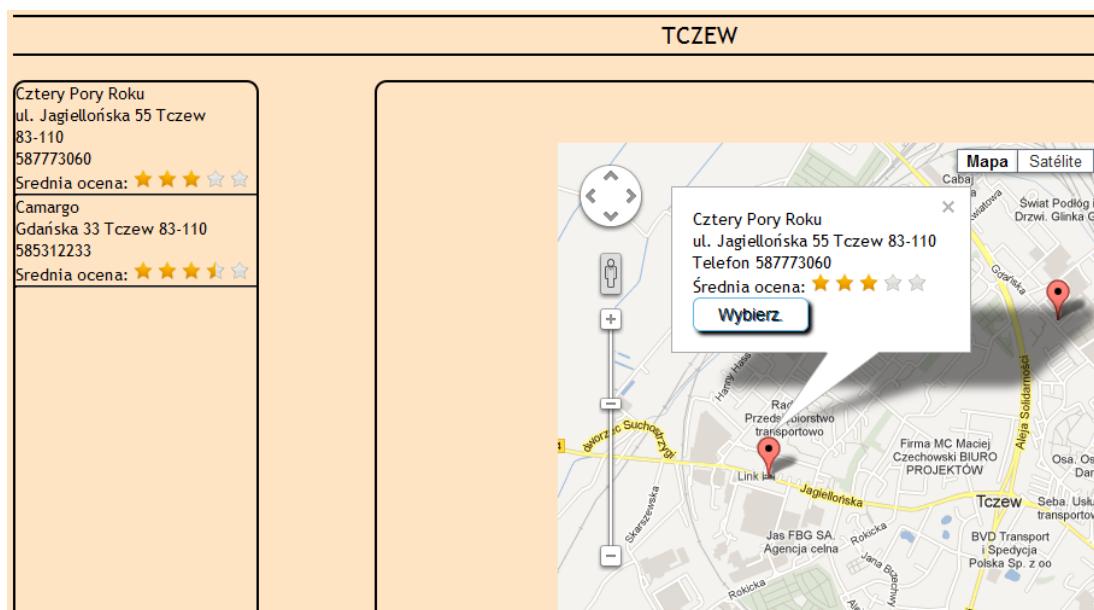
4.1.6. Wyszukiwanie restauracji (Paweł Gawroński)

Wyszukiwanie restauracji odbywa się za pomocą jednego z dwóch przeznaczonych do tego celu paneli. Panel szybkiego wyszukiwania (Rys. 4.1.6.1), umieszczony na stronie głównej aplikacji, pozwala na wpisanie nazwy miasta, w którym znajduje się szukana przez użytkownika restauracja. Po kliknięciu przycisku „Szukaj” wyświetlona jest lista restauracji, które znajdują się w podanym mieście.



Rys. 4.1.6.1. Podstawowy panel wyszukiwania restauracji (opracowanie własne).

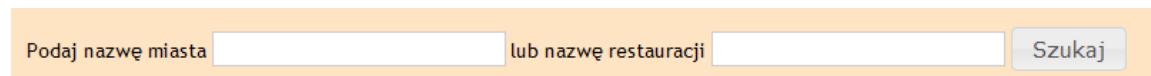
Wybranie z listy restauracji powoduje przekierowanie do strony restauracji, natomiast wybranie opcji „Wszystkie” spowoduje wyświetlenie wszystkich restauracji w danym mieście wraz z danymi o nich oraz mapkę z zaznaczanymi lokalizacjami tych restauracji (Rys. 4.1.6.2).



Rys. 4.1.6.2. Lista restauracji w danym mieście (opracowanie własne).

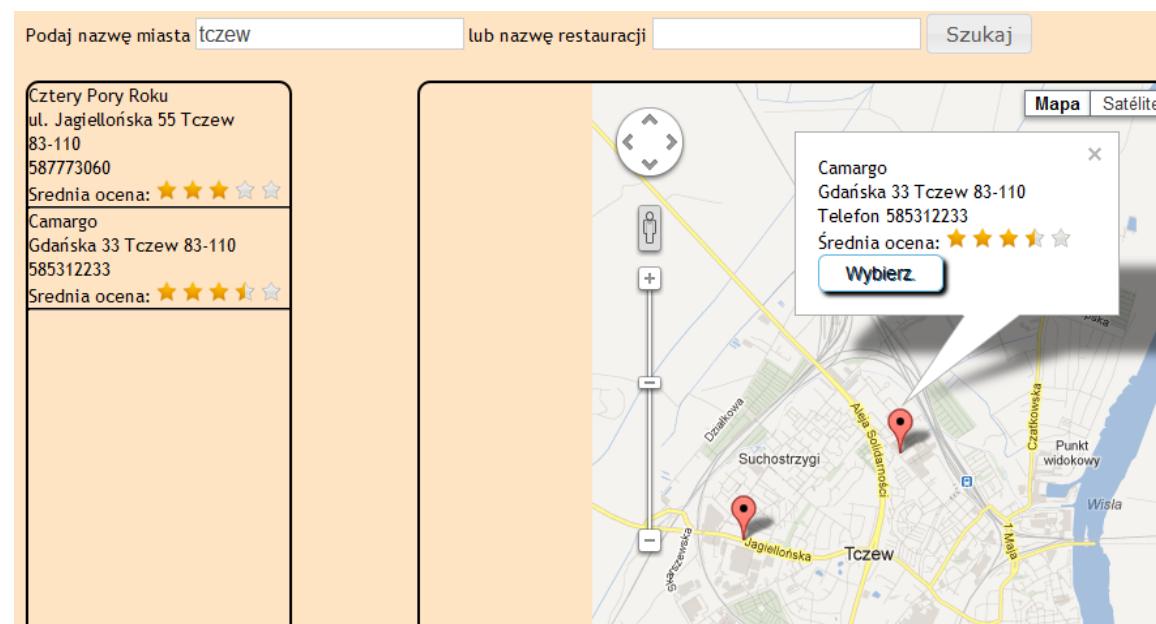
Wybranie restauracji z listy, bądź przy pomocy znacznika z mapki spowoduje przekierowanie na stronę restauracji.

Do szczegółowego wyszukiwania (Rys. 4.1.6.3) przechodzimy wybierając opcję „Szukaj” z menu głównego lub „Szczegółowe wyszukiwanie” z panelu szybkiego wyszukiwania.



Rys. 4.1.6.3. Zaawansowane wyszukiwanie restauracji (opracowanie własne).

Panel zawiera pola tekstowe przeznaczone na nazwę miasta, w którym znajdują się szukane restauracje oraz nazwę szukanej restauracji. Przynajmniej jedno z podanych wyżej pól musi zostać wypełnione. Nie jest wymagane podanie całej nazwy miasta czy restauracji, wystarczy ich fragment, np. „gda” zamiast „Gdańsk”. Po kliknięciu przycisku „Szukaj” wyświetlona zostaje lista restauracji, spełniających podane kryteria, wraz z informacjami o tych restauracjach oraz mapka z zaznaczanymi lokalizacjami tych restauracji (Rys. 4.1.6.4).



Rys. 4.1.6.4. Wyniki zaawansowanego wyszukiwania restauracji po nazwie miasta (opracowanie własne).

Wybranie restauracji z listy bądź przy pomocy znacznika z mapki spowoduje przekierowanie na stronę restauracji.

4.1.7. Strona restauracji (Paweł Gawroński)

Aby przejść do strony restauracji należy wybrać restaurację z panelu „10 najnowszych restauracji” lub za pomocą wyszukiwania opisanego powyżej. Strona

restauracji składa się z siedmiu zakładek „Strona główna”, „Menu”, „Dowóz”, „Imprezy okolicznościowe”, „Galeria”, „Kontakt”, „Komentarze”. „Strona główna”, „Dowóz”, „Imprezy okolicznościowe” oraz „Kontakt” zawierają w sobie informacje ustawione przez menadżera danej restauracji. Zakładka „Menu” zawiera kategorie oraz przypisane do poszczególnych kategorii produkty oferowane przez restauracje oraz informacje o danym produkcie (Rys. 4.1.7.1).

Sosy do wyboru: czosnkowy, jogurtowy, ostry, mieszany.

▼ Kebab

Opis: mięso, surówki, sosy

Cena:
12.50

Do wyboru: Czosnkowy

Ilość 1

Komentarz do produktu:

Do koszyka

▶ XL Kebab

▶ Kebab z frytkami

Rys. 4.1.7.1. Podgląd produktu w danej kategorii wybranej restauracji (opracowanie własne).

Jeśli użytkownik jest zalogowany ma możliwość dodania produktu do koszyka poprzez kliknięcie przycisku „Dodaj do koszyka”, dodatkowo użytkownik ma możliwość wybrania dostępnych opcji dotyczących danego produktu, wybrania ilości zamawianych produktów oraz wpisania komentarza dotyczącego zamawianego produktu. Zakładka „Galeria” zawiera zdjęcia udostępnione przez menadżera danej restauracji. Zdjęcia wyświetlane są w postaci miniaturek, a po kliknięciu na zdjęcie zostaje ono powiększone do oryginalnych rozmiarów. W zakładce „Komentarze” przedstawione są komentarze, jakie zostały wystawione restauracji wraz z oceną, w skali od 0.5 do 5, przyznane przez osoby wystawiające komentarze. Jeśli użytkownik jest zalogowany, to ma możliwość wystawienia komentarza danej restauracji. Dodawanie komentarzy zostało opisane w podrozdziale 4.1.10.

4.1.8. Edycja danych konta (Paweł Gawroński)

Edycja danych użytkownika (klienta, menadżera lub pracownika) odbywa się za pomocą formularza zawierającego pola tekstowe pozwalające na wprowadzenie adresu

poczty elektronicznej, imienia, nazwiska, adresu, nazwy miasta oraz kodu pocztowego, nazwy kraju, daty urodzenia, płci oraz numeru telefonu. Wszystkie pola muszą być wypełnione. Data urodzenia musi wykazywać pełnoletniość użytkownika, a nazwę kraju należy wybrać z listy rozwijanej. Edycja danych konta odbywa się po kliknięciu przycisku „Zapisz”. Pobrała zostaje lista miast zgodnych z nazwą miasta oraz kodem pocztowym, jeśli nie znaleziono miast spełniających kryteria, użytkownik zostaje poproszony o poprawne wpisanie danych miasta. Jeśli znaleziono więcej niż jedno miasto, użytkownik zostaje poproszony o poprawienie danych lub wybranie miasta z dołączonej do formularza mapki (Rys. 4.1.8.1).



Rys. 4.1.8.1. Mapa dopasowanych miast oraz kodów pocztowych (opracowanie własne).

Jeśli znaleziono tylko jedno miasto oraz pozostałe dane są wypełnione poprawnie zostaje wywołana metoda `CustomUpdateUser` z klasy `CustomMembershipProvider`, która sprawdza czy adres poczty elektronicznej nie jest przypisany do innego użytkownika. Jeśli wszystkie pola są wypełnione prawidłowo, zostaje wywołana metoda zapisująca dane do bazy danych (Kod 4.1.8.1).

```
User user =
 GetUserFromCustomMembershipUser((CustomMembershipUser)muser);

        string useremail = GetUserNameByEmail(user.Email);
        if (RequiresUniqueEmail &&
! (String.IsNullOrEmpty(useremail)))
        {
            if (useremail != user.Login)
            {
                status = "Podany adres email jest zajęty";
            }
        }
```

```

    if (String.IsNullOrEmpty(status))
    {
        bool value = false;
        try
        {
            ServiceReference.EresServiceClient client = new
ServiceReference.EresServiceClient();
            using (client)
            {
                value = client.UpdateUser(user);
            }
            client.Close();
        }
        catch (Exception e)
        {
            value = false;
            if (WriteExceptionsToEventLog)
            {
                WriteToEventLog(e, "UpdateUser");
                throw new ProviderException(exceptionMessage);
            }
            else
            {
                throw e;
            }
        }
        if (value == false)
        {
            status = "Nieznany błąd podczas zapisywania danych";
        }
    }

```

Kod 4.1.8.1. Walidacja oraz wywołanie metody edytującej dane

4.1.9. Zmiana hasła (Paweł Gawroński)

Zmiana dotychczasowego hasła do konta użytkownika odbywa się poprzez wypełnienie formularza zawierającego pola tekstowe „Dotychczasowe hasło”, „Nowe hasło” oraz „Powtórz hasło”. Wszystkie pola muszą zostać wypełnione, pola „Nowe hasło” oraz „Powtórz hasło” muszą zawierać jednakową treść. Zmiana hasła do konta odbywa się po kliknięciu przycisku „Zatwierdź”. Jeśli dotychczasowe hasło jest podane prawidłowo, wywołana zostaje metoda `ChangePassword` lub `ChangeEmployeePassword` (w przypadku zmiany hasła pracowników) z klasy `CustomMembershipProvider`, w której sprawdzane jest czy hasło spełnia kryteria ustalone w ustawieniach aplikacji dotyczące między innymi wymaganej ilości znaków i wymaganej ilości cyfr w haśle (Kod 4.1.9.1).

```

if (!ValidateUser(login, oldPwd))
    return false;

```

```

ValidateEventArgs args =
    new ValidateEventArgs(login, newPwd, true);

OnValidatingPassword(args);

if (args.Cancel)
    if (args.FailureInformation != null)
        throw args.FailureInformation;
else
    throw new MembershipPasswordException("Change
password canceled due to new password validation failure.");
Kod 4.1.9.1. Weryfikacja danych

```

Jeżeli nie zostaną wykryte nieprawidłowości w podanym haśle, hasło zostanie zaszyfrowane lub zahashowane, a nowe hasło zostaje dodane do systemu poprzez zapisanie go w bazie danych.

4.1.10. Komentarze (Paweł Gawroński)

Jeśli użytkownik jest zalogowany, to ma możliwość wystawienia komentarza danej restauracji poprzez wypełnienie formularza zawierającego pole ocen w formie gwiazdek oraz pole tekstowe przeznaczone na treść komentarza, znajdującego się w zakładce „Komentarze” na stronie wybranej restauracji (Rys. 4.1.10.1). Wypełnienie pola z treścią komentarza nie jest wymagane, w przypadku nie wypełnienia tego pola wystawiony komentarz zawierać będzie tylko ocenę.

Dodaj komentarz

Ocena:

Treść komentarza:

Wyślij

Numer komentarza: 2. Wstawiony przez: test 03/12/2012 16:30:00

Ocena: ★★★★★

Treść komentarza:

super

Rys. 4.1.10.1. Wystawianie komentarza (opracowanie własne).

Po kliknięciu przycisku „Wyślij” komentarz zostaje zapisany w bazie danych. W zakładce „Konto” - „Komentarze” zalogowany użytkownik może zobaczyć wszystkie wystawione przez siebie komentarze wraz z wystawioną oceną. Jeśli zajdzie taka potrzeba komentarz może zostać usunięty przez użytkownika, który go wystawił (Rys. 4.1.10.2).

Numer komentarza: 2. Wstawiony dla: Cztery Pory Roku (Tczew) 03/12/2012 16:30:00
Ocena: ★★★★☆
Treść komentarza:
super
Usuń

Rys. 4.1.10.2. Podgląd własnego komentarza (opracowanie własne).

Menadżer restauracji ma możliwość zgłoszenia nieprawidłowego lub nieodpowiedniego komentarza poprzez zgłoszenie nadużycia podczas edycji strony swojej restauracji (Rys. 4.1.10.3).

Numer komentarza: 2. Wstawiony przez: test 03/12/2012 16:30:00	
Ocena: ★★★★☆	
Treść komentarza:	
super	
Zgłaszcenie nadużycia w treści komentarza	
Powód	<input type="text"/>
Wyślij	

Rys. 4.1.10.3. Zgłaszcenie komentarza do administracji (opracowanie własne).

Zgłaszcenie nadużycia polega na wypełnieniu pola tekstowego przeznaczonego na powód zgłoszenia oraz kliknięciu przycisku „Wyślij”. Jeśli pole „Powód” zostało wypełnione zgłoszenie zostaje wysłane na adres poczty elektronicznej erestauracja.bledy@gmail.com (Kod 4.1.10.1).

```
SmtpClient klient = smtp;
        MailMessage wiadomosc = new MailMessage();
        try
        {
            wiadomosc.From = new MailAddress(eresEmail);
            wiadomosc.To.Add(eresError);
            wiadomosc.Subject = "ReportComment - " + login + " (" +
resId + ")";
            wiadomosc.Body = "Treść zgłoszenia: " +
System.Environment.NewLine + " " + report + System.Environment.NewLine +
"Komentarz: " + id +
System.Environment.NewLine + " Użytkownik: " +
userLogin + System.Environment.NewLine + " Treść komentarza: " +
comment;

            klient.Port = port;
            klient.Credentials = credential;
            klient.EnableSsl = true;
            klient.Send(wiadomosc);

            return true;
        }
```

```

    catch (Exception ex)
    {
        EventLog log = new EventLog();
        log.Source = eventSource;
        log.Log = eventLog;

        string info = "Błąd podczas wysyłania wiadomości email z
zgłoszeniem nadużycia";
        info += "Action: " + "SendReportComment" + "\n\n";
        info += "Exception: " + ex.ToString();

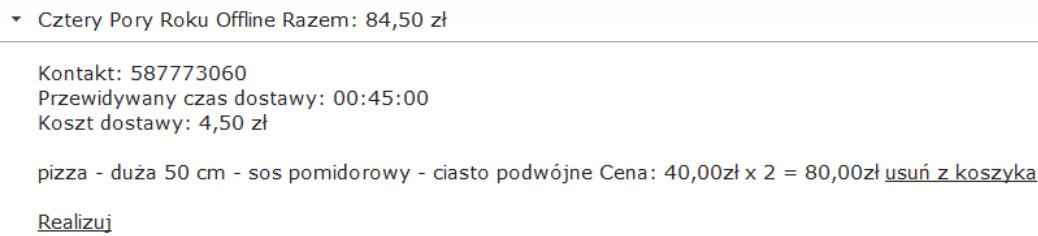
        return false;
    }

```

Kod 4.1.10.1. Wysyłanie wiadomości email.

4.1.11. Zamówienia (Paweł Gawroński)

Składanie zamówień odbywa się poprzez wybranie produktów z koszyka (zakładka „Koszyk”). Jeśli chcemy zrezygnować z któregoś z produktów znajdujących się w koszyku możemy go usunąć klikając znajdujący się przy nim link „usuń z koszyka” (Rys. 4.1.11.1).



Rys. 4.1.11.1. Podgląd koszyka (opracowanie własne).

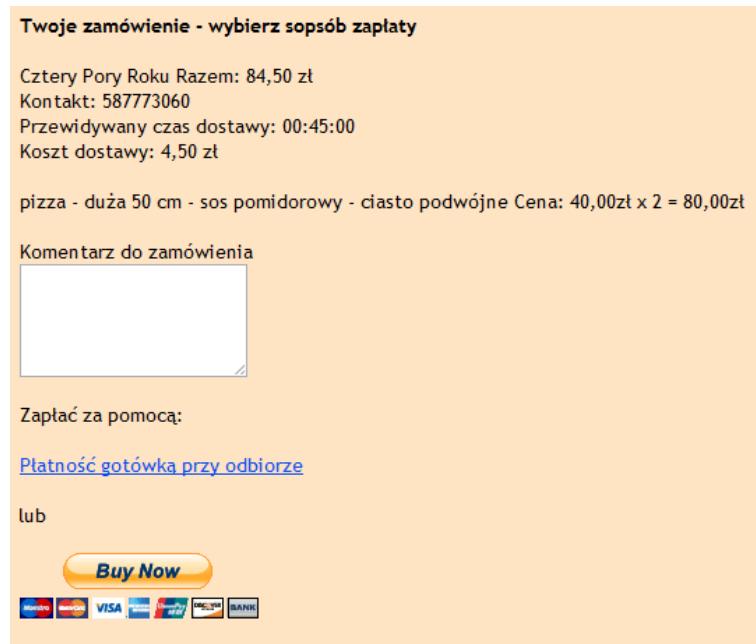
Klikając „Realizuj” użytkownik zostaje przekierowany do strony płatności (Rys. 4.1.11.2), zamówienie zostaje zapisane w bazie oraz następuje sprawdzenie czy liczba zamówionych produktów nie przekracza dwudziestu (Kod 4.1.11.1).

```

int count = -1;
foreach(BasketProduct item in rest.Products)
{
    count += item.Count;
    if (count > 20)
    {
        ViewData["error"] = "Maksymalna ilość zamówionych
produktów w jednym zamówieniu to 20";
        return View();
    }
}

```

Kod 4.1.11.1. Ograniczenie ilości zamawianych produktów.



Rys. 4.1.11.2. Realizacja zamówienia – wybór sposobu zapłaty (opracowanie własne).

Pole tekstowe „Komentarz do zamówienia” pozwala na wprowadzenie swoich uwag dotyczących zamówienia, pole to nie musi być wypełnione. Jeśli klikniemy na link „Płatność gotówką przy odbiorze” lub obrazek przedstawiony na Rys. 4.1.11.3.



Rys. 4.1.11.3. Podgląd własnego komentarza (opracowanie własne).

status restauracji zostanie sprawdzony i jeśli jest „Offline”, to użytkownik zostanie poinformowany o braku możliwości realizacji zamówienia (Kod 4.1.11.2).

```

bool test = false;
try
{
    Erestauracja.ServiceReference.EresServiceClient client =
new Erestauracja.ServiceReference.EresServiceClient();
    using (client)
    {
        test = client.IsRestaurantOnline(resid);
    }
    client.Close();
}
catch (Exception e)
{
    return RedirectToAction("Realize", "Basket", new { data
= "", error = "Błąd podczas sprawdzania czy restauracja jest dostępna"
});
}

```

```

if (test == false)
{
    return RedirectToAction("Realize", "Basket", new { data
= "", error = "Restauracja jest offline" });
}

```

Kod 4.1.11.2. Sprawdzanie dostępności restauracji przed wysłaniem zamówienia.

Wybierając płatność przy odbiorze dane dotyczące zamówienia zostaną zapisane do bazy, a na adres poczty elektronicznej restauracji zostanie wysłana wiadomość zawierająca szczegóły zamówienia. Jeśli użytkownik wybierze opcję PayPal zostanie przekierowany do strony <https://www.sandbox.paypal.com>, gdzie po zalogowaniu się i zaakceptowaniu warunków serwisu PayPal użytkownik ma możliwość dokonania płatności jedną z oferowanych przez serwis PayPal metod (Rys. 4.1.11.4).

E Res's Test Store

Rys. 4.1.11.4. Płatność PayPal (źródło: <https://www.sandbox.paypal.com>).

Po dokonaniu płatności użytkownik zostanie przekierowany do strony restauracji. Jeśli dane zwrócone przez serwis PayPal wskazują na powodzenie transakcji, użytkownik zostanie poinformowany o powodzeniu, w przeciwnym razie użytkownik zostanie poinformowany o sposobie, w jaki ma postępować w celu zwrotu poniesionych kosztów.

```

// Choose whether to use sandbox or live environment
string paypalUrl = useSandbox ?
"https://www.sandbox.paypal.com/cgi-bin/webscr"
: "https://www.paypal.com/cgi-bin/webscr";

```

```

        HttpWebRequest req =
(HttpWebRequest)WebRequest.Create(paypalUrl);

        // Set values for the request back
        req.Method = "POST";
        req.ContentType = "application/x-www-form-urlencoded";

        byte[] param = Request.BinaryRead(Request.ContentLength);
        string strRequest = Encoding.ASCII.GetString(param);

        StringBuilder sb = new StringBuilder();
        sb.Append(strRequest);

        foreach (string key in formVals.Keys)
        {
            sb.AppendFormat("&{0}={1}", key, formVals[key]);
        }
        strRequest = sb.ToString();
        req.ContentLength = strRequest.Length;

        //Send the request to PayPal and get the response
        string response = "";
        using (StreamWriter streamOut = new
StreamWriter(req.GetRequestStream(), System.Text.Encoding.ASCII))
{
    streamOut.Write(strRequest);
    streamOut.Close();
    using (StreamReader streamIn = new
StreamReader(req.GetResponse().GetResponseStream()))
    {
        response = streamIn.ReadToEnd();
    }
}

return response;

if (response == "VERIFIED")
{
    pp.txn_id = Request["txnid"];
    pp.mc_gross = Request["mc_gross"];
    pp.txn_type = Request["txn_type"];
    int id = int.Parse(pp.item_number);

    bool value = false;
    try
    {
        ServiceReference.EresServiceClient client = new
ServiceReference.EresServiceClient();
        using (client)
        {
            value = client.Pay(User.Identity.Name, id, comm,
"PayPal~" + pp.txn_id + "~" + pp.txn_type);
        }
        client.Close();
    }
    catch (Exception e)
    {
        value = false;
    }
}

```

```

        }
        if (value == false)
        {
            ModelState.AddModelError("", "Wysyłanie zamówienia
nie powiodło się.");
            return RedirectToAction("PayError", "Basket", new {
id = id });
        }
        else
        {
            DeleteRest(resid);

            return RedirectToAction("PaySuccess", "Basket", new
{ id = id });
        }
    }

```

Kod 4.1.11.3. Weryfikacja płatności PayPal.

Jeśli płatność zakończyła się powodzeniem(Kod 4.1.11.3.), zamówione produkty zostają usunięte z koszyka (Kod 4.1.11.4).

```

string lista = String.Empty;
lista = Restore();
List<string> usun = new List<string>();

foreach (string product in lista.Split(' | '))
{
    string[] dane = product.Split('~');
    if (id == Int32.Parse(dane[1]))
    {
        usun.Add(product);
    }
}

string newlista = lista;
foreach (String item in usun)
{
    if (!String.IsNullOrWhiteSpace(item))
    {
        int index = newlista.IndexOf(item);
        if (newlista.Length == item.Length)
        {
            newlista = newlista.Remove(index, item.Length);
        }
        else
        {
            newlista = (index == 0) ? newlista.Remove(index,
item.Length + 1) : newlista.Remove(index - 1, item.Length + 1);
        }
    }
}

int x = 0;
foreach(string item in newlista.Split(' | '))
{
    string[] data = item.Split('~');
    data[0] = x.ToString();
    x++;
}

```

Store(newlista);

Kod 4.1.11.4. Usuwanie zamówienia z koszyka.

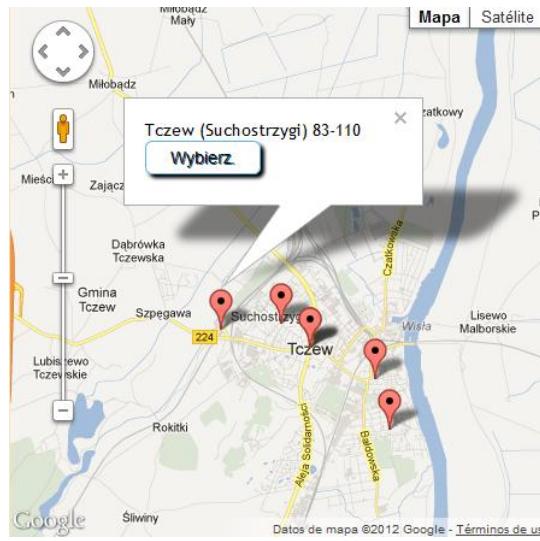
W zakładce „Aktualne zamówienia” zalogowany użytkownik może zobaczyć swoje zamówienia, które są w trakcie realizacji lub niedługo po zrealizowaniu. W zakładce „Konto” – „Historia zamówień” użytkownik ma możliwość zobaczenia historii swoich zamówień z okresu ograniczonego datami „Od” i „Do” (Rys. 4.1.11.5).

Od:	19/12/2012	To:	26/12/2012	Filtruj
▶ Numer:	175 Camargo Razem: 65,00 zł Płatność przy odbiorze Status: Zakończone			
▶ Numer:	185 Cztery Pory Roku Razem: 84,50 zł Płatność PayPal Status: Oczekujące			

Rys. 4.1.11.5. Podgląd historii zamówień (opracowanie własne).

4.1.12. Dodawanie restauracji (Paweł Gawroński)

Dodawanie nowej restauracji odbywa się za pomocą formularza zawierającego pola tekstowe pozwalające na wprowadzenie loginu, adresu poczty elektronicznej, hasła, pytania do przywracania hasła oraz odpowiedzi na to pytanie, nazwy firmy, nazwy wyświetlanej klientom na stronie głównej, adresu lokalu, nazwy miasta oraz kodu pocztowego, nazwy kraju, numeru telefonu, NIP-u, numeru REGON, czasu dostawy oraz ceny dostawy. Wszystkie pola muszą być wypełnione. Dodatkowo pola „Email” i „Powtórz email” oraz „Hasło” i „Powtórz hasło” muszą zawierać jednakową treść. Nazwę kraju należy wybrać z listy rozwijanej. Zakładanie nowego konta restauracji odbywa się po kliknięciu przycisku „Zapisz”. Pobrała zostaje lista miast zgodnych z nazwą miasta oraz kodem pocztowym. Jeśli nie znaleziono miast spełniających kryteria, użytkownik zostaje poproszony o poprawne wpisanie danych miasta. Jeśli znaleziono więcej niż jedno miasto, użytkownik zostaje poproszony o poprawienie danych lub wybranie miasta z dołączonej do formularza mapki (Rys. 4.1.12.1).



Rys. 4.1.12.1. Mapa dopasowanych miast oraz kodów pocztowych (opracowanie własne).

Jeśli znaleziono tylko jedno miasto oraz pozostałe dane są wypełnione poprawnie zostaje wywołana metoda `CreateRestaurant` z klasy `CustomMembershipProvider`, która sprawdza czy login oraz adres poczty elektronicznej nie są przypisane do innego użytkownika oraz czy hasło spełnia kryteria ustalone w ustawieniach aplikacji dotyczące, między innymi, wymaganej ilości znaków i wymaganej ilości cyfr w haśle (Kod 4.1.12.1).

```
ValidateEventArgs args =
    new ValidateEventArgs(login, password, true);

OnValidatingPassword(args);

if (args.Cancel)
{
    status = MembershipCreateStatus.InvalidPassword;
    return false;
}

if (RequiresUniqueEmail &&
! (String.IsNullOrEmpty(GetUserNameByEmail(email))))
{
    status = MembershipCreateStatus.DuplicateEmail;
    return false;
}
```

Kod 4.1.12.1. Walidacja danych.

Jeżeli nie zostaną wykryte nieprawidłowości w podanych danych, hasło oraz odpowiedź do przywracania hasła zostają zaszyfrowane lub zahashowane, ta nowa restauracja zostaje dodana do systemu poprzez stworzenie użytkownika będącego restauracją (Kod 4.1.12.2), któremu nadawana jest rola „Restauracja”. Współrzędne restauracji są pobierane na

podstawie adresu lokalu, dane restauracji zapisują się w bazie danych oraz tworzą się puste dane zawartości strony restauracji.

```
MembershipCreateStatus createStatus;

CustomMembershipProvider customMembership =
(CustomMembershipProvider)System.Web.Security.Membership.Providers["CustomMembershipProvider"];

bool user = customMembership.CreateRestaurant(model.Login, model.Email,
model.Password, model.Question, model.Answer, model.Name,
model.DisplayName, model.Address, value[0].ID, model.Country,
model.Telephone, model.Nip, model.Regon, model.DeliveryTime,
User.Identity.Name, out createStatus, price);
```

Kod 4.1.12.2. Dodawanie restauracji do systemu.

4.1.13. Edycja danych restauracji (Paweł Gawroński)

Edycja danych restauracji odbywa się za pomocą formularza zawierającego pola tekstowe pozwalające na wprowadzenie adresu poczty elektronicznej, nazwy firmy, nazwy wyświetlanej klientom na stronie głównej, adresu lokalu, nazwy miasta oraz kodu pocztowego, nazwy kraju, numeru telefonu, NIP-u, numeru REGON, czasu dostawy, ceny dostawy oraz przycisku wyboru pozwalającego ustawić czy restauracja ma być widoczna dla klientów. Wszystkie pola muszą być wypełnione, podany adres poczty elektronicznej nie może być wykorzystywany przez innego użytkownika. Nazwę kraju należy wybrać z listy rozwijanej. Edycja danych konta odbywa się po kliknięciu przycisku „Zatwierdź zmiany”. Pobrała zostaje lista miast zgodnych z nazwą miasta oraz kodem pocztowym. Jeśli nie znaleziono miast spełniających kryteria, użytkownik zostaje poproszony o poprawne wpisanie danych miasta. Jeśli znaleziono więcej niż jedno miasto, użytkownik zostaje poproszony o poprawienie danych lub wybranie miasta z dołączonej do formularza mapki (Rys. 4.1.13.1).

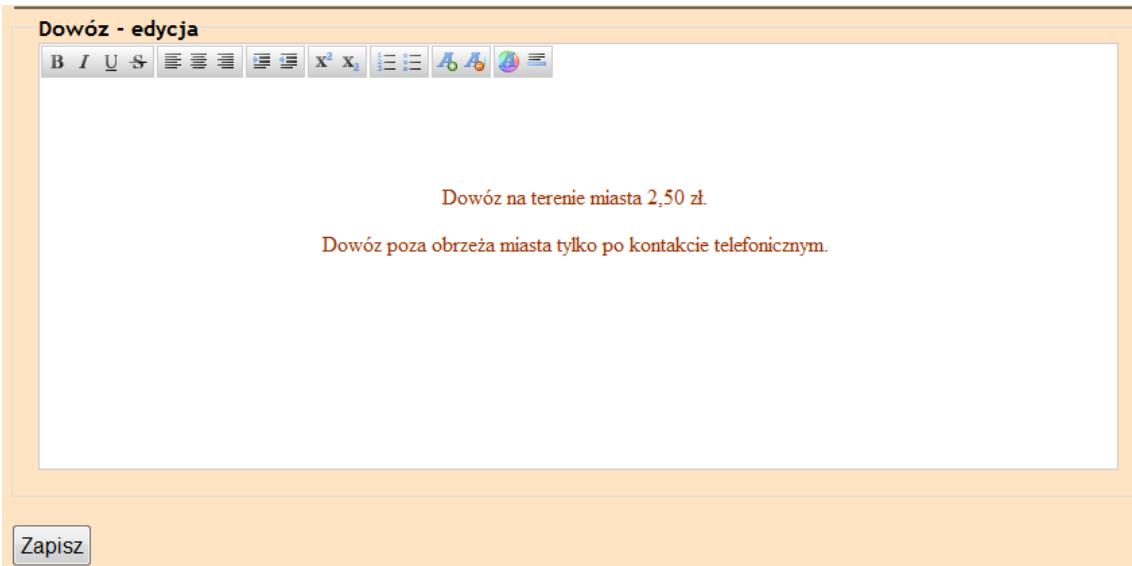


Rys. 4.1.13.1. Mapa dopasowanych miast oraz kodów pocztowych (opracowanie własne).

Jeśli znaleziono tylko jedno miasto oraz pozostałe dane są wypełnione poprawnie zostaje wywołana metoda `EditRestaurant`, która pobiera współrzędne restauracji oraz zapisuje nowe dane do bazy danych.

4.1.14. Zarządzanie restauracją (Paweł Gawroński)

Do panelu zarządzania menadżer przechodzi poprzez kliknięcie linku „Zarządzaj” w zakładce „Twoje restauracje”. W panelu zarządzania przedstawiona jest strona restauracji, którą widzi odwiedzający stronę danej restauracji klient, wraz z zakładkami „Strona główna”, „Menu”, „Dowóz”, „Imprezy okolicznościowe”, „Galeria”, „Kontakt”, „Komentarze”. W zakładce „Komentarze” wyświetlane są komentarze wystawione danej restauracji, wraz z możliwością zgłoszenia nadużycia, co jest opisane w podpunkcie 4.1.10. Zakładki „Strona główna”, „Dowóz”, „Imprezy okolicznościowe” oraz „Kontakt” zawierają dodatkową opcję edycji, po wybraniu której użytkownik ma możliwość wpisać oraz sformatować tekst wyświetlany na stronie restauracji (Rys. 4.1.14.1).



Rys. 4.1.14.1. Edycja strony z informacjami o dowozie (opracowanie własne).

Pole edycji zawiera pole tekstowe przeznaczone na treść strony oraz pasek zawierający opcje takie jak pogrubienie tekstu, kursywa, podkreślenie, skreślenie, wyrównanie do prawej i lewej strony, wyśrodkowanie tekstu, akapity, indeksy dolny i górny, wypunktowanie, zmiana rozmiaru oraz koloru czcionki oraz linie pozioma. Zmiany zostają zapisane po kliknięciu przycisku „Zapisz”. Zakładka „Strona główna” zawiera dodatkowo pole przeznaczone na ustawienie zdjęcia głównego (Rys. 4.1.14.2).



Rys. 4.1.14.2. Przykładowe główne logo restauracji (opracowanie własne).

Link „Załaduj zdjęcie główne” przekierowuje użytkownika do panelu dodawania nowego zdjęcia, gdzie użytkownik ma możliwość wybrania pliku .jpg, .jpeg lub .png oraz zapisania go poprzez kliknięcie przycisku „Wyślij” (Kod 4.1.14.1.).

```
// Create a directory
var newdirFolder = dropBoxStorage.CreateFolder("/Public/images/" +
id.ToString());

var resFolder = dropBoxStorage.GetFolder("/Public/images/" +
id.ToString());
```

```

if (fileUpload != null)
{
    var fileName = Path.GetFileName(fileUpload.FileName);
        var filepath = Path.Combine(Server.MapPath("~/App_Data/uploads"),
fileName);
    fileUpload.SaveAs(filepath);
    String srcFile =
Environment.ExpandEnvironmentVariables(filepath);
    dropBoxStorage.UploadFile(srcFile, resFolder);
//delete file from local storage after we upload him on dropbox
System.IO.File.Delete(filepath);
}

```

Kod 4.1.14.1. Zapis pliku w chmurze Dropbox.

Kliknięcie linku „Usuń zdjęcie główne” powoduje usunięcie zdjęcia głównego.

```

// Creating the cloudstorage object
CloudStorage dropBoxStorage = new CloudStorage();

// get the configuration for dropbox
var dropBoxConfig =
CloudStorage.GetCloudConfigurationEasy(nSupportedCloudConfigurations.Dro
pBox);

// declare an access token
ICloudStorageAccessToken accessToken = null;

// load a valid security token from file
string path = Server.MapPath(Url.Content("~/Content/token.txt"));

        //using (FileStream fs =
System.IO.File.Open("C:\\dropboxtoken.txt", FileMode.Open,
FileAccess.Read, FileShare.None))
using (FileStream fs = System.IO.File.Open(path, FileMode.Open,
FileAccess.Read, FileShare.None))
{
    accessToken = dropBoxStorage.DeserializeSecurityToken(fs);
}

// open the connection
var storageToken = dropBoxStorage.Open(dropBoxConfig, accessToken);

// Delete a file
dropBoxStorage.DeleteFileSystemEntry("/Public/images/" +
resid.ToString() + "/" + plik);
dropBoxStorage.Close();

```

Kod 4.1.14.2. Usuwanie pliku z chmury Dropbox.

Jeśli zdjęcie nie zostało załadowane lub zostało usunięte przez menadżera restauracji, wyświetlane jest zdjęcie przykładowe (Kod 4.1.14.3).

```

//default logo
var defaultLogoFolder =
dropBoxStorage.GetFolder("/Public/images/defaultlogo");

ICloudFileSystemEntry fsel;

```

```

// lista linkow uri
Uri defaultLogo;
String htmlImgLogo = string.Empty;

// enumerate all child (folder and files)
foreach (var fof in defaultLogoFolder)
{
    // check if we have a directory
Boolean bIsDirectory = fof is ICloudDirectoryEntry;

fsel = dropBoxStorage.GetFileSystemObject(fof.Name, defaultLogoFolder);
if (!bIsDirectory)
{
    //pobiera liste linkow do plikow w katalogu rodzica
    defaultLogo =
DropBoxStorageProviderTools.GetPublicObjectUrl(accessToken, fsel);
    htmlImgLogo = "<img style=\"background: url(" +
defaultLogo.AbsoluteUri + "); background-size: auto 100%; \\" \\\>";
    nowy.Foto = "";
}
}

```

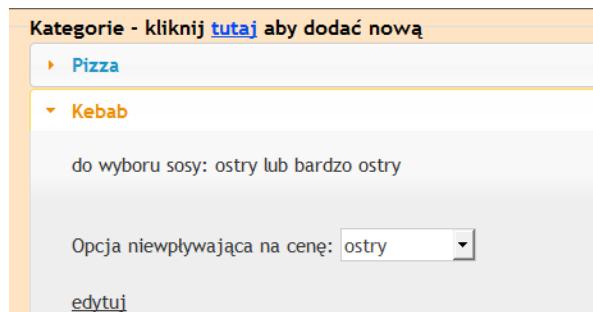
Kod 4.1.14.3. Część odpowiedzialna za odczyt standardowego pliku z logiem restauracji.

Zakładka „Galeria” pozwala na stworzenie galerii zdjęć dla klienta odwiedzającego stronę restauracji. Dodawanie nowych zdjęć dobywa się w sposób analogiczny do opisanego powyżej. Usuwanie zdjęcia dobywa się poprzez kliknięcie przycisku „Usuń zdjęcie”(Kod 4.1.14.2.), miniaturka zdjęcia zostaje zastąpiona informacją o jego usunięciu, a po fizycznym usunięciu zdjęcia z chmury danych znika z galerii (Rys. 4.1.14.3).



Rys. 4.1.14.3. Usunięte zdjęcie w galerii restauracji (opracowanie własne).

Edycja zakładki „Menu” pozwala na edycję oraz dodawanie nowych produktów i kategorii (Rys. 4.1.14.4 oraz Rys. 4.1.14.5).



Rys. 4.1.14.4. Podgląd kategorii (opracowanie własne).

Produkty - kliknij tutaj aby dodać nowy

Pizza

Kebab

test

hbk

testowa

qwerty

aaaaaaaaaaa

Nieprzypisane

Wszystkie

Wszystkie pizze zawierają sos pomidorowy oraz ser mazzarella, do wyboru zwykłe lub podwójne ciasto.

▼ pizza (dostępny)

Opis: sdfghjkld

Cena:

mała 15 cm - 20.00
średnia 30 cm - 30.00
duża 50 cm - 40.00

[Edytuj](#)

▼ picca (dostępny)

▼ dfghjkl (niedostępny)

Rys. 4.1.14.5. Podgląd produktu (opracowanie własne).

Edycja oraz dodawanie nowej kategorii odbywa się poprzez wypełnienie formularza zawierającego pola tekstowe przeznaczone na nazwę kategorii, opis, opcje wpływające na cenę, oraz dwie opcje nie wpływające na cenę (Rys. 4.1.14.6). Pole „Nazwa kategorii” jest wymagane, opcje powinny zostać oddzielone przecinkiem. Dodawanie oraz edycja kategorii następuje po kliknięciu przycisku „Zapisz”.

Nowa kategoria

Nazwa kategorii
Pizza

Opis
Do wyboru średnica pizzy.

Opcja wpływająca na cene (dowolna ilość opcji oddzielonych przecinkiem ',' np. mała,średnia,duża)
mała,średnia,duża

Opcja niewpływająca na cene (dowolna ilość opcji oddzielonych przecinkiem ',' np. ketchup,sos pomidorowy)
ketchup,sos pomidorowy

Opcja niewpływająca na cene (dowolna ilość opcji oddzielonych przecinkiem ',' np. ciasto cieknie,ciasto średnie,podwójne ciasto)

Dodaj

Rys. 4.1.14.6. Dodawanie kategorii (opracowanie własne).

Edycja oraz dodawanie nowego produktu odbywa się poprzez wypełnienie formularza zawierającego pola tekstowe przeznaczone na nazwę produktu, opis oraz pola wyboru kategorii (Rys. 4.1.14.7). Po wybraniu kategorii dla produktu wyświetlane zostaje pole tekstowe przeznaczone na cenę produktu. Pole „Nazwa kategorii” oraz „Cena” są wymagane. Ceny powinny zostać oddzielone pionową kreską ‘|’. Dodawanie kategorii następuje po kliknięciu przycisku „Dodaj”.

Nowy produkt

Nazwa produktu
Familijna

Opis
Składniki: sos, ser, szynka.

Kategoria
Pizza

Cena Wprowadź cenę kolejno dla: mała 15 cm,średnia 30 cm,duża 50 cm (np. 10.00|15.00|20.00)
10.00|15.00|20.00

Dodaj

Rys. 4.1.14.7. Dodawanie produktu (opracowanie własne).

Formularz edycji zawiera dodatkowo przycisk wyboru, za pomocą którego menadżer restauracji ustala czy produkt ma być widoczny dla klientów. Edycję produktu zatwierdzamy klikając przycisk „Zapisz”.

4.1.15. POS (Dariusz Kohlandt)

POS, czyli point of sale został przez nas stworzony w wersji minimalistycznej, tak by spełniał założone wymagania naszej aplikacji. Znajduje się w nim podgląd aktywnych

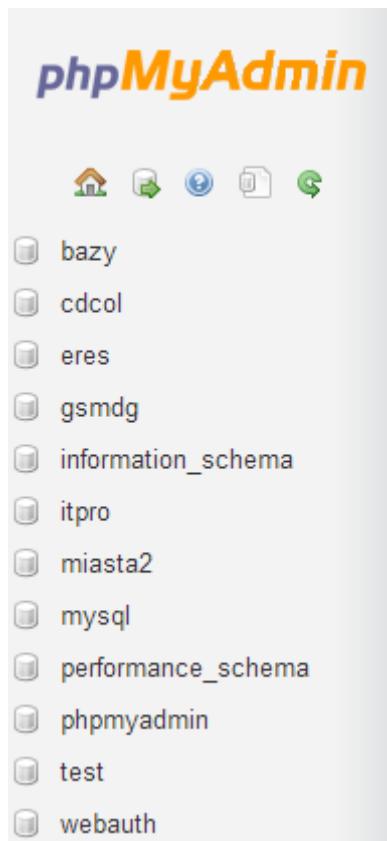
zamówień pobieranych cyklicznie z bazy danych (metoda `GetOrders`), podgląd wszystkich zamówień (metoda `GetAllOrders`) oraz możliwość blokady panelu, by inny pracownik nie mógł wykonywać operacji na zamówieniach z nie swojego konta. Blokowanie panelu wykonuje się poprzez zalogowanie restauracji, co jednocześnie wylogowuje pracownika restauracji (Kod 4.1.15.1.).

```
FormsAuthentication.SetAuthCookie(logs[0], false);  
Kod 4.1.15.1. Wylogowanie pracownika.
```

Przewidzieliśmy również bardzo ważną opcję zmiany statusu restauracji, tak by klient widział po swojej stronie jej dostępność. Służy do tego funkcja `SetRestaurantOnline` przyjmująca, jako parametry `User.Identity.Name` oraz nowy status. Aktywność restauracji jest aktualizowana automatycznie, by zapobiec sytuacjom, w których panel jest odblokowany (zalogowany pracownik), a przy terminalu fizycznym nie znajduje się żadna osoba. W takim wypadku po kilku minutach brak aktywności w panelu powoduje zmianę statusu restauracji na offline. Do pobierania zamówień POS korzysta z metod zaimplementowanych przez nas w serwisie, które zostały opisane wyżej. Zamówienia są wyświetlanie w komponentach jQuery – accordion oraz tabs. Zmiany statusów obsługiwane są przez skrypty pisane przez nas w JavaScript.

4.1.16. Panel Administratora (Dariusz Kohlandt)

W aplikacji stworzyliśmy panel do zarządzania treścią strony i danymi w bazie danych. Do tego celu został zainstalowany na serwerze z bazą danych phpMyAdmin (Rys. 4.1.16.1.).



Rys. 4.1.16.1. phpMyAdmin (źródło phpMyAdmin).

Podstawowe funkcje takie jak tworzenie (Kod 4.1.16.1.)

```
CustomMembershipUser user = customMemebership.CreateUser(model.Login,
model.Password, model.Email, model.Name, model.Surname, model.Address,
value[0].ID, model.Country, model.Birthdate, model.Sex, model.Telephone,
model.Question, model.Answer, true, out createStatus);  
Kod 4.1.16.1. Tworzenie nowego użytkownika.
```

i usuwanie (Kod 4.1.16.2.)

```
customMemebership.DeleteUser(user, true);  
Kod 4.1.16.2. Usuwanie użytkownika.
```

użytkowników oraz tworzenie (Kod 4.1.16.3.)

```
CustomRoleProvider r =
(CustomRoleProvider)System.Web.Security.Roles.Providers["CustomRoleProvider"];
r.CreateRole(model.RoleName);
Kod 4.1.16.3. Tworzenie nowej roli.
```

i usuwanie (Kod 4.1.16.4.)

```
CustomRoleProvider r =
(CustomRoleProvider)System.Web.Security.Roles.Providers["CustomRoleProvider"];
r.DeleteRole(role, true);
```

Kod 4.1.16.4. Usuwanie roli.

ról zostały zaimplementowane dodatkowo w panelu administratora w celu wsparcia jego pracy.

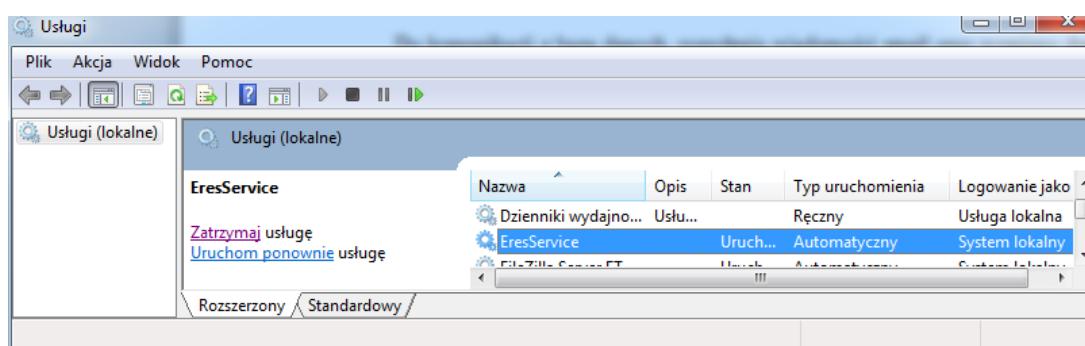
4.2. Budowa aplikacji

4.2.1. Środowisko pracy (Dariusz Kohlandt)

Nasz zespół składa się z dwóch osób, więc zaistniała potrzeba współdzielenia zasobów projektowych na czas wytwarzania aplikacji. Bazę danych postanowiliśmy umieścić na maszynie wirtualnej na Platformie Obsługi Nauki PLATON, która wydała nam się najlepiej konfigurowalnym miejscem w porównaniu do darmowych ofert na rynku. Połączenie było realizowane poprzez wirtualną sieć (VPN) przy użyciu programu Hamachi. Jako system kontroli wersji wykorzystaliśmy SVN z darmową przestrzenią oferowaną przez jednego z dostawców.

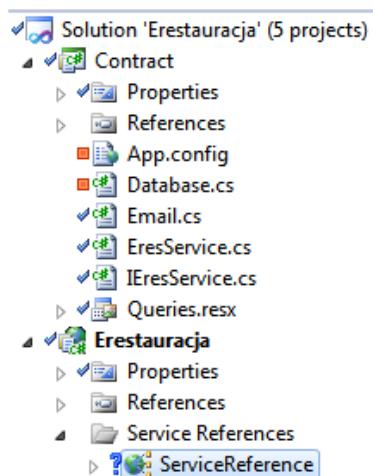
4.2.2. Serwis WCF oparty na usłudze Windows (Paweł Gawroński)

Do komunikacji z bazą danych, rozsyłania wiadomości elektronicznych oraz wymiany danych stworzyliśmy serwis WCF udostępniany, jako usługa systemu Windows (Rys. 4.2.2.1). Dzięki pracy w tle, automatycznemu wznawianiu pracy w przypadku wykrycia błędów lub wyjątków oraz braku wymaganej interakcji z użytkownikiem serwis może pracować nawet, jeśli użytkownik nie jest zalogowany na maszynie, na której serwis jest zainstalowany.



Rys. 4.2.2.1. Usługa EresService (opracowanie własne).

Usługę podłączliśmy do projektu aplikacji za pomocą referencji serwisowych (Rys. 4.2.2.2).



Rys. 4.2.2.2. Contract i podłączona referencja do usługi w projekcie (opracowanie własne).

Plik IEResService.cs zawiera kontrakt oraz definicje obiektów wykorzystywanych do wymiany informacji (Kod 4.2.2.1).

```
[OperationContract]
ValidateUser ValidateUser(string login);

[DataContract]
public class ValidateUser
{
    private string password = null;
    private bool isApproved = false;

    [DataMember]
    public string Password
    {
        get { return password; }
        set { password = value; }
    }

    [DataMember]
    public bool IsApproved
    {
        get { return isApproved; }
        set { isApproved = value; }
    }
}
```

Kod 4.2.2.1. Fragment pliku IEResService.cs.

W pliku EresService.cs znajduje się implementacja kontraktu usługi (Kod 4.2.2.2).

```
public class EresService : IEResService
{
```

```

    public ValidateUser ValidateUser(string login)
    {
        if (!(String.IsNullOrEmpty(login)))
        {
            Database db = new Database();
            ValidateUser value = db.ValidateUser(login);
            return value;
        }
        else
        {
            return null;
        }
    }
}

```

Kod 4.2.2.2. Fragment pliku EresService.cs.

Kolejne ważne pliki tworzące serwis to Database.cs oraz Email.cs. W pliku Email.cs zawarte są metody odpowiedzialne za rozsyłanie wiadomości elektronicznych (Kod 4.2.2.3).

```

class Email
{
    private SmtpClient smtp = new SmtpClient("smtp.gmail.com");
    private string eresEmail = "eresztauracja@gmail.com";
    private string eresError = "eresztauracja.bledy@gmail.com";
    int port = 587;
    System.Net.NetworkCredential credential = new
    System.Net.NetworkCredential("eresztauracja", "Erestauracja123");

    public bool SendPassword(string email, string password)
    {
        SmtpClient klient = smtp;
        MailMessage wiadomosc = new MailMessage();
        try
        {
            wiadomosc.From = new MailAddress(eresEmail);
            wiadomosc.To.Add(email);
            wiadomosc.Subject = "Erestauracja - restet hasła.";
            wiadomosc.Body = "Nowe hasło: " + password;

            klient.Port = port;
            klient.Credentials = credential;
            klient.EnableSsl = true;
            klient.Send(wiadomosc);

            return true;
        }
        catch (Exception ex)
        {//fragment wycięty ze względu na czytelność kodu}
    }
}

```

Kod 4.2.2.3. Fragment pliku Email.cs.

Plik Database.cs zawiera metody odpowiedzialne za komunikację z bazą danych, zarówno pobieranie informacji jak i zapis danych do bazy (Kod 4.2.2.4)

```
private DataSet ExecuteQuery(MySqlCommand command, string action)
{
    MySqlConnection conn = new MySqlConnection();
    DataSet myDS = new DataSet();

    try
    {
        conn = new MySqlConnection(ConnectionString);
        conn.Open();
        command.Connection = conn;

        MySqlDataAdapter da = new MySqlDataAdapter(command);
        da.Fill(myDS);
    }
    catch (MySqlException e)
    {
        //fragment wycięty ze względu na czytelność kodu
    }
    catch (Exception ex)
    {
        //fragment wycięty ze względu na czytelność kodu
    }
    finally
    {
        conn.Close();
    }

    return myDS;
}

public ValidateUser ValidateUser(string login)
{
    MySqlCommand command = new
    MySqlCommand(Queries.ValidateUser);
    command.Parameters.AddWithValue("@login", login);
    command.Parameters.AddWithValue("@isLockedOut", false);

    ValidateUser value = null;

    DataSet ds = new DataSet();
    ds = ExecuteQuery(command, "ValidateUser");

    if (ds.Tables.Count > 0)
    {
        value = new ValidateUser();
        value.Password = null;
        value.IsApproved = false;

        foreach (DataRow row in ds.Tables[0].Rows)
        {
            if (row["password"] != DBNull.Value) value.Password
= row["password"].ToString();
            if (row["isApproved"] != DBNull.Value)
value.IsApproved = Convert.ToBoolean(row["isApproved"]);
        }
    }
}
```

```

        }
    }
    return value;
}

```

Kod 4.2.2.4. Fragment pliku Database.cs.

Wszystkie zapytania SQL zapisane są w oddzielnym pliku Queries.resx w celu poprawy czytelności kodu (Rys. 4.2.2.3).

Name	Value	Comment
AddCategory	INSERT INTO `category`(`restaurantId`, `categoryName`, `categoryDescription`, `priceOption`, `nonPriceOption`, `nonPriceOption2`) VALUES (@restaurantID,@categoryName,@categoryDescription,@priceOption,@nonPriceOption,@nonPriceOption2)	
AddComment	INSERT INTO `comments`(`userId`, `restaurantId`, `rating`, `comment`, `date`) VALUES (SELECT `id` FROM `users` WHERE `login` = @login, @userId, @rating, @comment, GETDATE())	
AddEmptyContent	INSERT INTO `rest_page_content`(`menagerId`, `restaurantId`) VALUES ((SELECT `id` FROM `users` WHERE `login` = @login, @menagerId))	
AddProduct	INSERT INTO `products`(`restaurantId`, `categoryId`, `name`, `description`, `price`, `priceOption`, `creationDate`, `lastUpdate`) VALUES (@restaurantID,@categoryID,@name,@description,@price,@option,@creationDate,@lastUpdate)	
AddProductToOrder	INSERT INTO `products_in_order`(`orderId`, `productId`, `priceOption`, `count`, `priceXcount`, `nonPriceOption1`, `nonPriceOption2`) VALUES (@orderId,@productId,@option,@count,@priceXcount,@option1,@option2)	
AddRestaurant	INSERT INTO `restaurants`(`name`, `displayName`, `address`, `townId`, `countryId`, `telephone`, `nip`, `regon`, `creationDate`, `lastUpdate`) VALUES (@name,@displayName,@address,@townID,@countryID,@telephone,@nip,@regon,@creationDate,@lastUpdate)	
AddUsersToRoles	INSERT INTO `users_in_roles`(`userId`, `roleID`) VALUES ((SELECT `id` FROM `users` WHERE `login` = @login), (SELECT `id` FROM `roles` WHERE `rolename` = @rolename))	
AddUserToRestaurant	INSERT INTO `employees_in_restaurants`(`userId`, `restaurantId`) VALUES (@userId, @restaurantId)	
AllUsersCount	SELECT Count(*) FROM `users`	
ChangePassword	UPDATE `users` SET `password` = @password, `lastPasswordChangedDate` = @lastPasswordChangedDate WHERE `login` = @login	
ChangePasswordQuestionAndAnswer	UPDATE `users` SET `passwordQuestion` = @question, `passwordAnswer` = @answer WHERE `login` = @login	
CreateRole	INSERT INTO `roles`(`rolename`) VALUES (@rolename)	
CreateUser	INSERT INTO `users`(`login`, `password`, `email`, `name`, `surname`, `address`, `townID`, `countryID`, `birthdate`, `sex`, `creationDate`, `lastUpdate`) VALUES (@login, @password, @email, @name, @surname, @address, @townID, @countryID, @birthdate, @sex, GETDATE(), GETDATE())	
DeleteCategory	DELETE FROM `category` WHERE `id` = @id AND `restaurantId` = @restaurantId	
DeleteComment	DELETE FROM `comments` WHERE `id` = @id AND `userId` = (SELECT `id` FROM `users` WHERE `login` = @login)	
DeleteRole	DELETE FROM `roles` WHERE `rolename` = @rolename	

Rys. 4.2.2.3. Fragment pliku Queries.resx (opracowanie własne).

Wszelkie błędy powstające w czasie pracy usługi zapisywane są w dzienniku zdarzeń systemu Windows (Kod 4.2.2.5 i Rys. 4.2.2.4).

```

catch (MySqlException e)
{
    EventLog log = new EventLog();
    log.Source = eventSource;
    log.Log = eventLog;

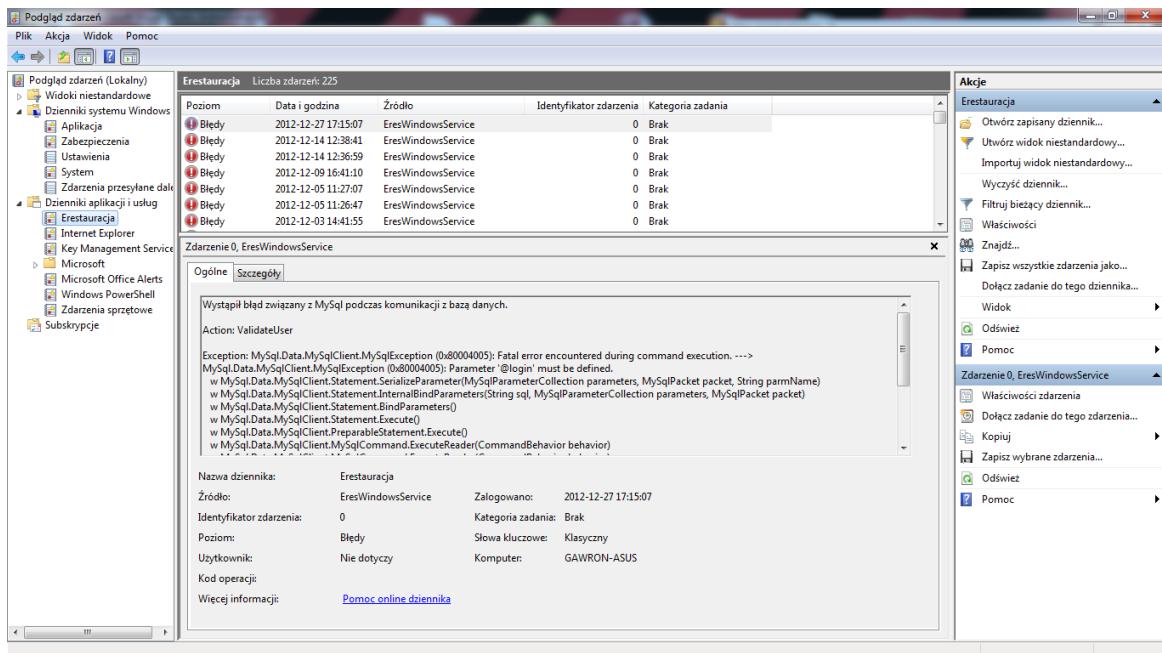
    string wiadomosc = message;
    wiadomosc += "Action: " + "GetTowns" + "\n\n";
    wiadomosc += "Exception: " + e.ToString();

    log.WriteEntry(wiadomosc, EventLogEntryType.Error);

    if (reader != null) { reader.Close(); }
    conn.Close();
    status = "Błąd sql: "+e.Message;
    return null;
}

```

Kod 4.2.2.5. Obsługa wyjątków, zapisywanie informacji do dziennika zdarzeń.



Rys. 4.2.2.4. Dziennik zdarzeń (opracowanie własne).

Aplikacja kliencka, czyli w naszym przypadku strona internetowa łączy się z usługą Windows za pomocą wspominanych wcześniej referencji serwisowych (Kod 4.2.2.6) i pobiera dane potrzebne aplikacji.

```

try
{
    ServiceReference.EresServiceClient client = new
    ServiceReference.EresServiceClient();
    using (client)
    {
        value = new List<Town>(client.GetTowns(out status, model.Town,
model.PostalCode));
    }
    client.Close();
}
catch (Exception e)
{
    ModelState.AddModelError("", "Pobieranie miast nie powiodło się.");
}

```

Kod 4.2.2.6. Połączenie z serwisem.

Aby aplikacja kliencka mogła połączyć się z serwisem oraz aby wymiana danych przebiegała w zaplanowany przez nas sposób wymagana jest prawidłowa konfiguracja klienta przedstawiona poniżej (Kod 4.2.2.7).

```

<binding name="CustomBinding" closeTimeout="00:03:00"
openTimeout="00:03:00"
sendTimeout="00:10:00" maxBufferSize="2147483647"
maxBufferPoolSize="2147483647"
maxReceivedMessageSize="2147483647">

```

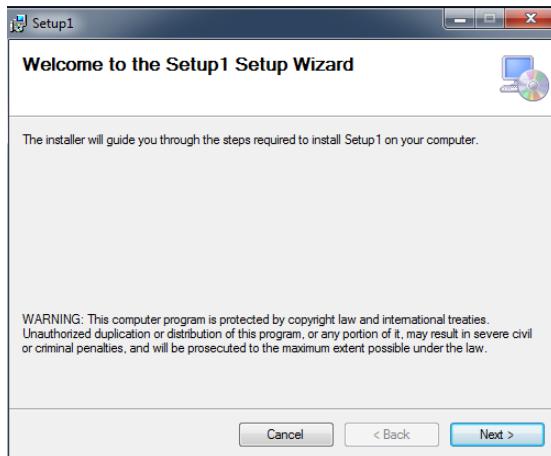
```

        <readerQuotas maxDepth="32" maxStringContentLength="16384"
maxArrayLength="32768"
            maxBytesPerRead="4096" maxNameTableCharCount="16384" />
    </binding>

<client>
    <endpoint
address="http://localhost:8733/Erestauracja/Contract/EresService/"
        binding="basicHttpBinding" bindingConfiguration="CustomBinding"
        contract="ServiceReference.IEresService"
name="BasicHttpBinding_IEresService" />
</client>
```

Kod 4.2.2.7. Konfiguracja punktu końcowego serwisu WCF wraz z konfiguracją.

Instalacja usługi Windows przeprowadzana jest za pomocą podstawowego instalatora (Rys. 4.2.2.5).

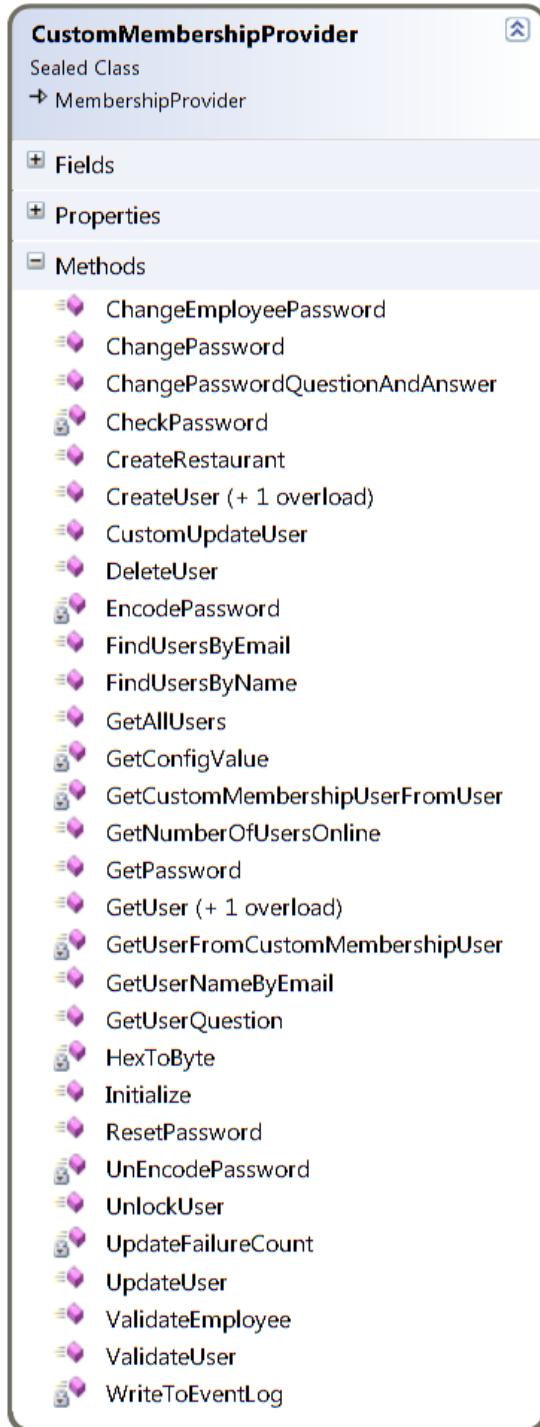


Rys. 4.2.2.5. Instalator usługi (opracowanie własne).

4.2.3. Model dostawców (Paweł Gawroński)

Do zarządzania członkostwem oraz rolami użytkowników wykorzystaliśmy model dostawców ASP.NET 4. Z oferowanych obiektów dostawców wykorzystaliśmy MembershipProvider oraz RoleProvider z przestrzeni nazw System.Web.Security. Klasy te zostały przez nas przerobione, aby dopasować je do wymogów związanych z naszą aplikacją.

Klasa CustomMembershipProvider.cs to dostosowany do naszej aplikacji dostawca członkostwa, bazuje na klasie System.Web.Security.MembershipProvider, której metody zostały nadpisane (Rys. 4.2.3.1).



Rys. 4.2.3.1. Klasa CustomMembershipProvider (opracowanie własne).

Przykład nadpisanej metody w klasie CustomMembershipProvider (Kod 4.2.3.1) jest pokazany poniżej:

```
public override bool ChangePassword(string login, string oldPwd, string newPwd)
{
    if (!ValidateUser(login, oldPwd))
        return false;
```

```

ValidatePasswordEventArgs args =
    new ValidatePasswordEventArgs(login, newPwd, true);

OnValidatingPassword(args);

if (args.Cancel)
    if (args.FailureInformation != null)
        throw args.FailureInformation;
    else
        throw new MembershipPasswordException("Change
password canceled due to new password validation failure.");

bool value = false;
try
{
    ServiceReference.EresServiceClient client = new
ServiceReference.EresServiceClient();
    using (client)
    {
        value = client.ChangePassword(login,
EncodePassword(newPwd));
    }
    client.Close();
}
catch (Exception e)
{
    if (WriteExceptionsToEventLog)
    {
        WriteToEventLog(e, "ChangePassword");
        throw new ProviderException(exceptionMessage);
    }
    else
    {
        throw e;
    }
}

return value;
}

```

Kod 4.2.3.1. Fragment pliku CustomMembershipProvider.cs.

Ustawienia dostawcy członkostwa definiowane są w pliku konfiguracyjnym w znaczniku <membership> (Kod 4.2.3.2).

```

<membership defaultProvider="CustomMembershipProvider">
    <providers>
        <clear />
        <add name="CustomMembershipProvider"
            type="Erestauracja.Providers.CustomMembershipProvider"
            applicationName="Eres"
            maxInvalidPasswordAttempts="5"
            passwordAttemptWindow="10"
            minRequiredNonalphanumericCharacters="0"
            minRequiredPasswordLength="6"
            passwordStrengthRegularExpression=""
            enablePasswordReset="true"
            enablePasswordRetrieval="false"
            requiresQuestionAndAnswer="true"
            requiresUniqueEmail="true"
        </add>
    </providers>
</membership>

```

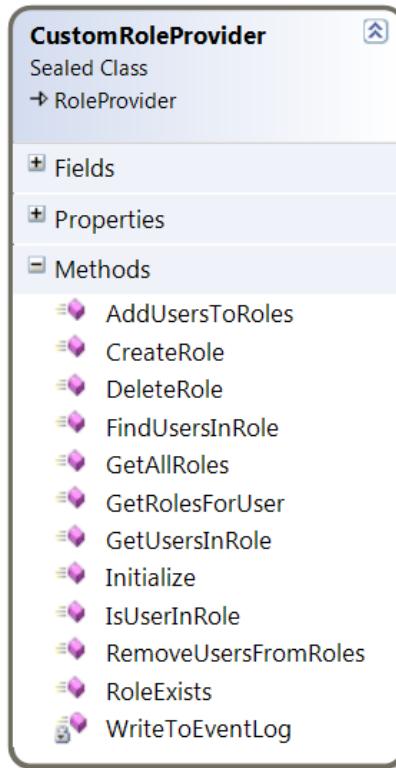
```
        passwordFormat="Clear"
        writeExceptionsToEventLog="true"
        userIsOnlineTimeWindow="15"
        description="" />
    </providers>
</membership>
```

Kod 4.2.3.2. Fragment pliku Web.config.

Istotne fragmenty konfiguracji to:

- minRequiredNonalphanumericCharacters - liczba wymaganych znaków innych niż litery w haśle;
- minRequiredPasswordLength - liczba wymaganych znaków w haśle;
- enablePasswordReset - możliwość resetowania hasła (ustawiania nowego);
- enablePasswordRetrieval - możliwość odzyskiwania hasła (np. wysyłanie użytkownikowi starego hasła);
- requiresQuestionAndAnswer - wymaganie pytania i odpowiedzi do odzyskiwania hasła;
- requiresUniqueEmail - wymaganie unikalności adresu poczty elektronicznej;
- passwordFormat - format hasła zapisywanego do bazy danych (Clear – bez zmian, Encrypted – szyfrowanie kluczem `<machineKey>` z pliku konfiguracji, Hashed - hashowanie);
- writeExceptionsToEventLog - zapisywanie błędów do dziennika zdarzeń;

Klasa `CustomRoleProvider.cs` to dostosowany do naszej aplikacji dostawca ról, bazuje na klasie `System.Web.Security.RoleProvider`, której metody zostały nadpisane (Rys. 4.2.3.2).



Rys. 4.2.3.2. Klasa CustomRoleProvider (opracowanie własne).

Przykład nadpisanej metody w klasie CustomRoleProvider (Kod 4.2.3.3).

```

public override bool IsUserRole(string login, string rolename)
{
    bool userIsInRole = false;

    try
    {
        ServiceReference.EresServiceClient client = new
ServiceReference.EresServiceClient();
        using (client)
        {
            userIsInRole = client.IsUserRole(login, rolename);
        }
        client.Close();
    }
    catch (Exception e)
    {
        if (WriteExceptionsToEventLog)
        {
            WriteToEventLog(e, "IsUserRole");
            throw new ProviderException(exceptionMessage);
        }
        else
        {
            throw e;
        }
    }
    return userIsInRole;
}

```

Kod 4.2.3.3. Fragment pliku CustomRoleProvider.cs.

Ustawienia dostawcy ról definiowane są w pliku konfiguracyjnym w znaczniku `<roleManager>` (Kod 4.2.3.4).

```
<roleManager
    enabled="true"
    cacheRolesInCookie="true"
    cookieName=".ASPROLES"
    defaultProvider="CustomRoleProvider">
    <providers>
        <clear />
        <add
            applicationName="Eres"
            writeExceptionsToEventLog="false"
            name="CustomRoleProvider"
            type="Erestauracja.Providers.CustomRoleProvider" />
    </providers>
</roleManager>
```

Kod 4.2.3.4. Fragment pliku Web.config.

Istotne fragmenty konfiguracji to:

- `writeExceptionsToEventLog` – zapisywanie błędów do dziennika zdarzeń;
- `cacheRolesInCookie` – czy zapisywać dane dotyczące ról do plików cookie;
- `cookieName` – nazwa pliku cookie w którym znajdują się dane dotyczące ról;

Klasa `CustomMembershipUser.cs` bazuje na klasie `System.Web.Security.MembershipUser`, rozszerza klasę bazową o dodatkowe pola, które chcieliśmy obsłużyć za pomocą providera (Kod 4.2.3.5).

```
public class CustomMembershipUser : MembershipUser
{
    public int Id { get; set; }
    public string Login { get; set; }
    public string Name { get; set; }
    public string Surname { get; set; }
    public string Address { get; set; }
    public string Town { get; set; }
    public string PostalCode { get; set; }
    public string Country { get; set; }
    public DateTime Birthdate { get; set; }
    public string Sex { get; set; }
    public string Telephone { get; set; }

    public CustomMembershipUser(
        string providername,
        string email,
        string passwordQuestion,
        string comment,
        bool isApproved,
        bool isLockedOut,
        DateTime creationDate,
```

```

        DateTime lastLoginDate,
        DateTime lastActivityDate,
        DateTime lastPasswordChangedDate,
        DateTime lastLockedOutDate,
        int id,
        string login,
        string name,
        string surname,
        string address,
        string town,
        string postalCode,
        string country,
        DateTime birthdate,
        string sex,
        string telephone
    ) :
    base(providername,
        login,
        Guid.NewGuid(),
        email,
        passwordQuestion,
        comment,
        isApproved,
        isLockedOut,
        creationDate,
        lastLoginDate,
        lastPasswordChangedDate,
        lastActivityDate,
        lastLockedOutDate)
{
    Id = id;
    Login = login;
    Name = name;
    Surname = surname;
    Address = address;
    Town = town;
    PostalCode = postalCode;
    Country = country;
    Birthdate = birthdate;
    Sex = sex;
    Telephone = telephone;
}
}

```

Kod 4.2.3.5. Fragment pliku CustomMembershipUser.cs.

4.2.4. Przekazywanie danych (Dariusz Kohlandt)

Widoki są wypełniane danymi pobieranymi z serwisu i przekazywane w metodach kontrolerów za pomocą:

-modeli (Kod 4.2.4.1., Kod 4.2.4.2., Kod 4.2.4.3. oraz Rys. 4.2.4.1.):

```

public class ProductModel
{
    [HiddenInput]
    public int ProductId { get; set; }

```

```

[HiddenInput]
public int RestaurantID { get; set; }

[Required]
[DataType(DataType.Text)]
[Display(Name = "Nazwa produktu")]
public string ProductName { get; set; }

[DataType(DataType.Text)]
[Display(Name = "Opis")]
public string ProductDescription { get; set; }

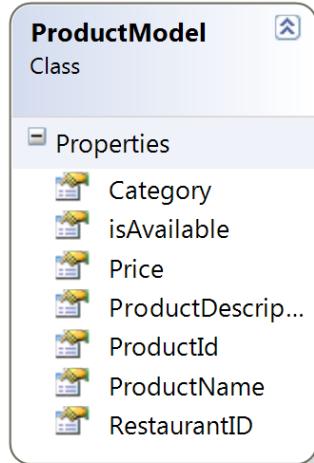
[Required]
[DataType(DataType.Text)]
[Display(Name = "Kategoria")]
public int Category { get; set; }

[Required]
[DataType(DataType.Text)]
[Display(Name = "Cena")]
public string Price { get; set; }

[Display(Name = "Dostępny")]
public bool IsAvailable { get; set; }
}

```

Kod 4.2.4.1. Model ProductModel.



Rys. 4.2.4.1. Model ProduktModel (opracowanie własne).

```

ProductModel model = new ProductModel();
model.IsAvailable = value2.IsAvailable;
model.Category = value2.CategoryId;
model.Price = value2.Price;
model.ProductDescription = value2.ProductDescription;
model.ProductName = value2.ProductName;
model.RestaurantID = value2.RestaurantId;
model.ProductId = value2.ProductId;
return View(model);

```

Kod 4.2.4.2. Przekazanie modelu z danymi do widoku.

```

<%@ Page Language="C#"
Inherits="System.Web.Mvc.ViewPage<Erestauracja.Models.ProductModel>" %>

```

```

<div class="editor-label">
    <%: Html.LabelFor(m => m.ProductName)%>
</div>
<div class="editor-field">
    <%: Html.TextBoxFor(m => m.ProductName)%>
    <%: Html.ValidationMessageFor(m => m.ProductName)%>
</div>

```

Kod 4.2.4.3. Odniesienie do danych modelu w widoku .

-JSON'ów (JavaScript Object Notation) (Kod 4.2.4.4. i Kod 4.2.4.5.):

```

prices = "Wprowadz cenę kolejno dla: "+value.PriceOption+ " (np. ";
if (value.PriceOption != null)
{
    string[] tab = value.PriceOption.Split(',');
    int j = 10;
    for (int i = 1; i <= tab.Length; i++)
    {
        if (i == tab.Length)
            prices += j.ToString() + ".00";
        else
            prices += j.ToString() + ".00|";
        j += 5;
    }
}
else
{
    prices += "10.00";
}
prices += ")";
return Json(prices);

```

Kod 4.2.4.4. Wypełnienie JSON'a danymi .

```

if (!$(id).val()) {
    $.post(url, data, function (data) {
        $('#subBut').show();
        $('#cenatb').show();
        $('#cenalabele').show();
        $('#pricelbl').text(data);
    });
}

```

Kod 4.2.4.5. Przypisanie danych z JSON'a do elementu HTML .

-ViewData (Kod 4.2.4.6. i Kod 4.2.4.7.):

```

CustomMembershipProvider customMemebership =
(CustomMembershipProvider)System.Web.Security.Membership.Providers["CustomMembershipProvider"];
MembershipUserCollection users = customMemebership.GetAllUsers(0, 20,
out count);
ViewData["users"] = users;
return View();

```

Kod 4.2.4.6. Przekazanie danych poprzez ViewData w kontrolerze .

```

<%foreach (CustomMembershipUser id in
(MembershipUserCollection)ViewData["users"] )
{
    <%>
    <li class="mainRole">
        <div class="roleName">
            <%: Html.Label(id.ToString())%></div>
        <div class="deleteAction">
            <%: Html.ActionLink("usuń", "deleteUser",
"Admin", new { user = id }, null)%></div>
        </li>
    <% }%>

```

Kod 4.2.4.7. Wypisanie danych z ViewData w widoku.

4.2.5. Widoki (Dariusz Kohlandt)

Widoki tworzyliśmy przy użyciu HTML5 (Kod. 4.2.5.1.), CSS3 (Kod. 4.2.5.2.) i jQuery (Kod. 4.2.5.3.).

```

<div class="editor-label">
    <%: Html.LabelFor(m => m.Email)%>
</div>

```

Kod. 4.2.5.1. Html helper w kodzie html wyświetlający adres poczty elektronicznej z modelu.

```

.mapTowns
{
    position: relative;
    z-index: 2;
    padding: 10px 0 0 0px;
    float: right;
    height: 100%;
    display: none;
    top: 40px;
    min-width: 450px;
}

```

Kod. 4.2.5.2. Fragment klasy CSS nadającej styl elementowi w widoku.

```

<script type="text/javascript">
    function callMethod() {
        //IsOnline
        $('.ResID').each(function () {
            var Resid = $(this).val();
            var url = '<%: Url.Action("IsOnline", "Home") %>';
            var data = { id: Resid };
            $.post(url, data, function (data) {
                $(".resIsOnline" + Resid).text("(" + data + ")");
                if (data === "Online") {
                    $(".resIsOnline" + Resid).css("color",
"#66CC00");
                }
                else {
                    $(".resIsOnline" + Resid).css("color",
"#FF0000");
                }
            });
        });
    }

```

```

        }
    </script>

    <script type="text/javascript">
        $(document).ready(function () {
            callMethod();
            setInterval("callMethod()", 30000);
        });
    </script>

```

Kod. 4.2.5.3. Cykliczny skrypt wywołujący metodę IsOnline z kontrolera Home, która zwraca status restauracji na stronie głównej.

4.2.6. Autoryzacja użytkowników (Dariusz Kohlandt)

Autoryzacja użytkowników została skonfigurowana w pliku web.config dla kontrolerów, które jej wymagają. Została ona wpisana do tego pliku, ponieważ wymagany był jednoczesny dostęp anonimowy oraz dostęp użytkowników o konkretnej roli do poszczególnych metod w kontrolerach (Kod. 4.2.6.1.).

```

<location path="Find">
    <system.web>
        <authorization>
            <allow roles="Klient" />
            <allow users="?" />
            <deny users="*" />
        </authorization>
    </system.web>
</location>

```

Kod 4.2.6.1. Autoryzacja użytkowników w pliku web.config dla kontrolera Find.

Gdy niewymagane było jednoczesne dopuszczanie użytkowników anonimowych oraz zalogowanych, którym zostały nadane role, wtedy stosowaliśmy zmodyfikowany atrybut autoryzacji podając, jako parametry nazwy ról, np.

[CustomAuthorizeAttribute(Roles = "Restauracja")] (Kod. 4.2.6.2.).

```

public class CustomAuthorizeAttribute : AuthorizeAttribute
{
    public override void OnAuthorization(AuthorizationContext filterContext)
    {
        base.OnAuthorization(filterContext);
        if (!filterContext.HttpContext.User.Identity.IsAuthenticated)
        {
            filterContext.Result = new RedirectToResult("~/Account/Logon");
            return;
        }

        if (filterContext.Result is HttpUnauthorizedResult)
        {

```

```

        filterContext.Result = new
RedirectResult("~/Home/Unauthorized");
        return;
    }
}
}

```

Kod 4.2.6.2. Nadpisana metoda atrybutu autoryzacji.

4.2.7. Połaczanie z chmurą danych (Dariusz Kohlandt)

W celu nawiązania połączenia (Kod 4.2.7.1.) z serwerami Dropbox'a użyliśmy biblioteki SharpBox [27]. Posłużyła ona nam do wysyłania plików graficznych na serwer (Kod 4.1.14.1.). Pliki te były używane do prezentacji galerii na stronie restauracji. Przy jej pomocy również tworzyliśmy katalogi (Kod 4.2.7.2.) oraz usuwaliśmy wybrane pliki(Kod 4.1.14.2.).

```

// Creating the cloudstorage object
CloudStorage dropBoxStorage = new CloudStorage();

// get the configuration for dropbox
var dropBoxConfig =
CloudStorage.GetCloudConfigurationEasy(nSupportedCloudConfigurations.DropBox);

// declare an access token
ICloudStorageAccessToken accessToken = null;

// load a valid security token from file
string path = Server.MapPath(Url.Content("~/Content/token.txt"));

//System.IO.File.Open("C:\\dropboxtoken.txt", FileMode.Open, FileAccess.Read,
FileShare.None)
using (FileStream fs = System.IO.File.Open(path, FileMode.Open, FileAccess.Read,
FileShare.None))
{
    accessToken = dropBoxStorage.DeserializeSecurityToken(fs);
}

// open the connection
var storageToken = dropBoxStorage.Open(dropBoxConfig, accessToken);

```

Kod 4.2.7.1. Otwieranie połącznie za pomocą biblioteki SharpBox.

```

// Create a directory
var newdirFolder = dropBoxStorage.CreateFolder("/Public/images/" + id.ToString());

```

Kod 4.2.7.2. Tworzenie nowego katalogu.

4.2.8. Użyte komponenty (Dariusz Kohlandt)

Accordion, Datepicker, Spinner oraz Tabs są darmowymi komponentami jQuery [28]. Accordion tworzy rozwijaną listę. Został użyty przez nas między innymi do utworzenia menu restauracji oraz wyświetlenia zamówień. Datepicker łączy kalendarz z polem tekstowym, który jest otwierany w momencie kliknięcia w to pole. Zastosowaliśmy go na stronach z edycją danych oraz formularzach rejestracji. Spinner nakłada na pole tekstowe ograniczenia, dzięki którym można wprowadzić tam tylko wartości numeryczne. Dodatkowo dokłada do pola strzałki służące zwiększaniu i zmniejszaniu wartości. Wykorzystaliśmy go by ułatwić wybór ilości zamawianego dania. Tabs jest komponentem pozwalającym tworzyć zakładki. Dzieli on menu restauracji na kategorie.

Kolejnym komponentem użyтыm przez nas jest komponent „raty” [29]. Posłużył on nam do stworzenia systemu oceniania restauracji.

jHtmlArea [30] tworzy prosty edytor kodu html na stronie internetowej. Wykorzystaliśmy go do edycji treści na stronach restauracji.

jQuery Price Format [31] jest dodatkiem tworzącym maskę z ceną w polu tekstowym. Zastosowaliśmy go przy dodawaniu i edycji danych restauracji w polu z ceną dowozu. jQuery on screen keyboard [32] jest dodatkową klawiaturą, która pojawia się na ekranie po ustawnieniu kurSORA na odpowiednie pole tekstowe. Użyliśmy ją w panelu POS do wprowadzania danych w pola tekstowe.

Maska na pola z datą w naszej aplikacji to komponent Masked Input [33].

Mapa ze znacznikami restauracji oraz miast i kodów pocztowych to Google Map bazująca na rozszerzeniach Telerik [34].

YoxView jest dodatkiem tworzącym galerię ze zdjęć [35]. Został użyty przez nas do stworzenia galerii na stronach restauracji.

4.3. Problemy w czasie wytwarzania

4.3.1. Rozsyłanie zamówień (Dariusz Kohlandt)

Podczas projektowania komunikacji zastanawialiśmy się nad sposobem przekazywania informacji dotyczących poszczególnych zamówień składanych przez klientów. Jednym z pomysłów było zmodyfikowanie naszego serwisu na serwis routujący, który rozsyłałby zamówienia bezpośrednio do restauracji. Każda restauracja byłaby w takim wypadku dodatkowym punktem końcowym dla serwisu. Zrezygnowaliśmy z tego rozwiązania z powodu każdorazowej konfiguracji serwisu przy rejestraniu nowej restauracji. Ostatecznie zdecydowaliśmy się na przesył zamówień przez bazę danych (Kod 4.3.1.1. i Kod 4.3.1.2), która spełniała też wymaganą przez nas funkcję archiwum zamówień.

```
try
{
    ServiceReference.EresServiceClient client = new
    ServiceReference.EresServiceClient();
    using (client)
    {
        value = client.SaveOrder(User.Identity.Name, rest);
    }
    client.Close();
}
catch (Exception e)
{
    ModelState.AddModelError("", "Zapisywanie zamówienia nie powiodło
    się.");
}
```

Kod 4.3.1.1. Zapis zamówienia do bazy danych.

```
try
{
    Erestauracja.ServiceReference.EresServiceClient client = new
    Erestauracja.ServiceReference.EresServiceClient();
    using (client)
    {
        value = client.GetOrders(User.Identity.Name);
    }
    client.Close();
}
catch (Exception e)
{
    value = null;
}
```

Kod 4.3.1.2. Odczyt zamówień dla konkretnej restauracji.

4.3.2. Lista miast (Dariusz Kohlandt)

Kolejnym problemem było dla nas pozyskanie kompletnej listy miast wraz z kodami pocztowymi. Nie udało nam się uzyskać dostępu do aktualnego spisu miast i kodów pocztowych w postaci bazy danych, ale uzyskaliśmy dostęp do bazy aktualnej na rok 2009. Znaleziona baza zawierała niepożądane przez nas dane, które zostały usunięte przez napisany w tym celu program kopiujący pożądane informacje z udostępnionej bazy do naszej. Baza miast z 2009 roku nie zawierała współrzędnych geograficznych potrzebnych w naszym projekcie, jednak na podstawie danych miast stworzyliśmy przyporządkowaną listę współrzędnych, które posłużyły do umieszczenia znaczników na zaimplementowanej w naszej aplikacji mapie Google. Współrzędne były pobierane na zasadzie żądań do odpowiedniego serwera Google z podaniem nazwy miasta, gminy, powiatu, województwa, kraju oraz ulicy, jeśli znajdowała się ona w bazie. Serwer zwracał wysokość i szerokość geograficzną, które były wstawiane do przyporządkowanej komórki w bazie danych. Do wykonania tej operacji napisaliśmy zewnętrzna aplikację pobierającą te dane (Kod 4.3.2.1.).

```
public static Coordinate GetCoordinates(string region)
{
    using (var client = new WebClient())
    {
        string uri = "http://maps.google.com/maps/geo?q=" +
region +
"'&output=csv&key=ABQIAAAAzr2EBOXUKnm_jVnk0OJI7xSosDVG8KKPE1" +
"-m51RBrvYughuyMxQ-i1QfUnH94QxWIa6N4U6MouMmBA";

        string[] geocodeInfo =
client.DownloadString(uri).Split(',');

        NumberStyles style;
        CultureInfo culture;

        style = NumberStyles.AllowDecimalPoint |
NumberStyles.AllowThousands;
        culture = CultureInfo.CreateSpecificCulture("en-CA");

        double lat;
        double lng;

        double.TryParse(geocodeInfo[2].ToString(), style,
culture, out lat);
        double.TryParse(geocodeInfo[3].ToString(), style,
culture, out lng);
        return new Coordinate(lat, lng);
    }
}
```

Kod 4.3.2.1. Pobieranie współrzędnych z serwera Google.

4.3.3. Tworzenie menu (Paweł Gawroński)

Projektując menu restauracji staraliśmy się, aby dać menadżerowi, tworzącemu stronę restauracji, możliwość pełnego odwzorowania swojej oferty. Zadanie to okazało się dość kłopotliwe, ponieważ większość pomysłów związana była z pewnymi ograniczeniami, które nie spełniały określonych przez nas wymogów. Przeanalizowaliśmy menu wielu restauracji, co pozwoliło na ustalenie formy obecnego systemu. Pierwszym etapem tworzenia menu jest dodanie nowej kategorii (Rys. 4.3.3.1), poza nazwą kategorii ustawiany jest także opis kategorii pozwalający na podanie informacji wspólnej dla wszystkich produktów znajdujących się w danej kategorii. Opcja wpływająca na cenę pozwala na zdefiniowanie dowolnej ilości cen dla produktu w danej kategorii. Oznacza to, że cena produktu należącego do kategorii np. ‘Pizza’ może być różna dla różnych średnic ciasta. Dzięki temu menadżer nie musi tworzyć klika produktów różniących się tylko rozmiarem i ceną, znacząco poprawia to także czytelność menu restauracji. Opcje niewpływające na cenę pozwalają na zdefiniowanie dwóch kategorii opcji zawierających dowolną ilość elementów, których wybór nie wpływa na cenę produktu, oznacza to, że klient ma możliwość zmiany poszczególnych, ustalonych przez menadżera, składowych produktu, np. wybór zwykłego lub podwójnego ciasta w przypadku kategorii ‘Pizza’. Dzięki tym opcjom menu restauracji odpowiada ofertom istniejących restauracji. Podając opcje wystarczy oddzielić je przecinkiem, a dzięki tej prostej konstrukcji nawet niedoświadczeni użytkownicy będą w stanie odwzorować menu swojej restauracji.

Nowa kategoria

Nazwa kategorii
Pizza

Opis
Do wyboru średnica pizzy.

Opcja wpływająca na cene (dowolna ilość opcji oddzielonych przecinkiem ',' np. mała,średnia,duża)
mała,średnia,duża

Opcja niewpływająca na cene (dowolna ilość opcji oddzielonych przecinkiem ',' np. ketchup,sos pomidorowy)
ketchup,sos pomidorowy

Opcja niewpływająca na cene (dowolna ilość opcji oddzielonych przecinkiem ',' np. ciasto cieknie,ciasto średnie,podwójne ciasto)

Dodaj

Rys. 4.3.3.1. Dodawanie nowej kategorii (opracowanie własne).

Dodawanie produktów jest równie intuicyjne jak dodawanie kategorii (Rys. 4.3.3.2). Poza nazwą produktu menadżer ma możliwość dodania opisu zawierającego szczegółowe informacje o danym produkcie. Z listy rozwijanej menadżer wybiera kategorię, do której przypisany ma zostać dany produkt. Po wybraniu kategorii wyświetlona zostaje odpowiedź dotycząca formy, w jakiej ma zostać podana cena produktu.

Nowy produkt

Nazwa produktu
Familijna

Opis
Składniki: sos, ser, szynka.

Kategoria
Pizza ▾

Cena Wprowadz cenę kolejno dla: mała 15 cm,średnia 30 cm,duża 50 cm (np. 10.00|15.00|20.00)
10.00|15.00|20.00

Dodaj

Rys. 4.3.3.2. Dodawanie nowego produktu (opracowanie własne).

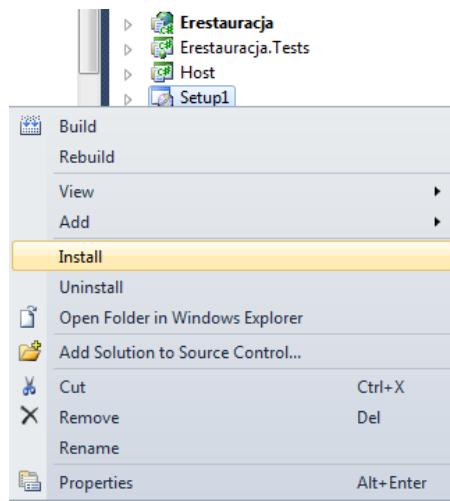
5. INSTRUKCJE URUCHOMIENIA ORAZ OBSŁUGI APLIKACJI

5.1. Instrukcja uruchomienia

5.1.1. Instrukcja uruchomienia usługi EresService (Paweł Gawroński)

5.1.1.1. Instalacja usługi

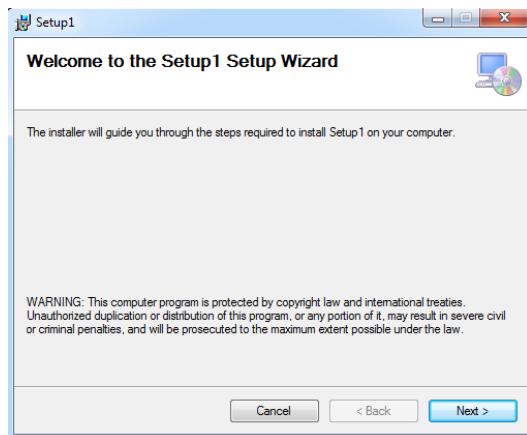
Aby zainstalować usługę EresService należy otworzyć plik setup.exe znajdujący się w lokalizacji: katalog projektu\Setup1\Debug lub w Visual Studio klikając prawym przyciskiem myszy na projekt Setup1, a następnie wybierając opcję „Install” (Rys. 5.1.1.1.1).



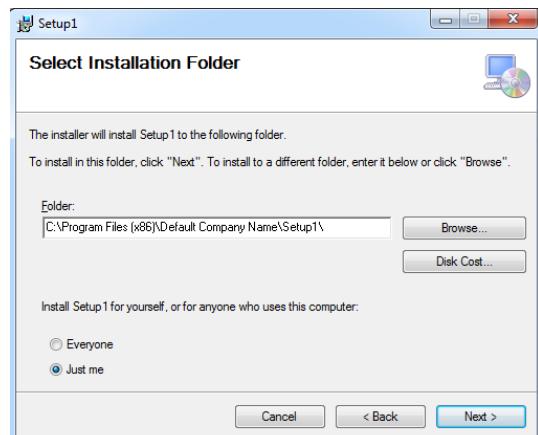
Rys. 5.1.1.1.1. Instalacja w Visual Studio.

Instalacja usługi przebiega w pięciu krokach:

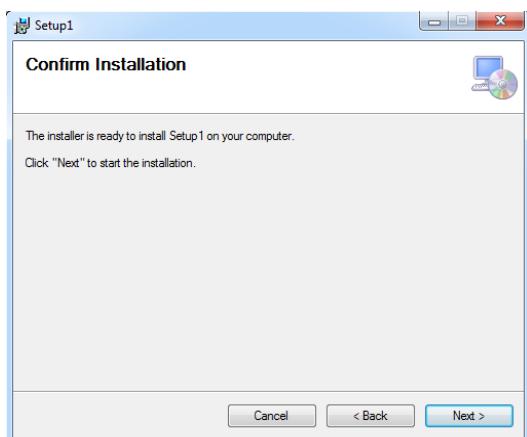
1. Wstęp do instalacji (Rys. 5.1.1.1.2);
2. Wybór katalogu do instalacji (Rys. 5.1.1.1.3);
3. Potwierdzenie instalacji (Rys. 5.1.1.1.4);
4. Instalowanie (Rys. 5.1.1.1.5);
5. Zakończenie instalacji (Rys. 5.1.1.1.6);



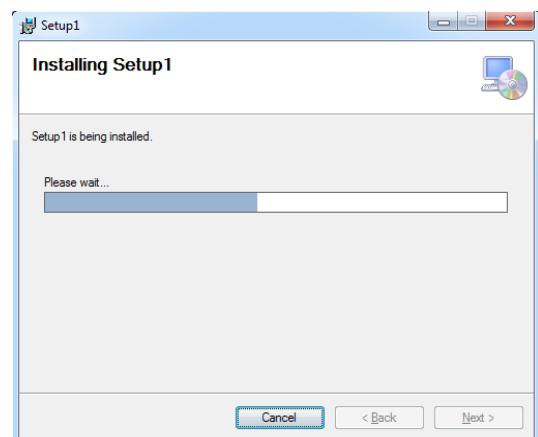
Rys. 5.1.1.1.2. Wstęp do instalacji.



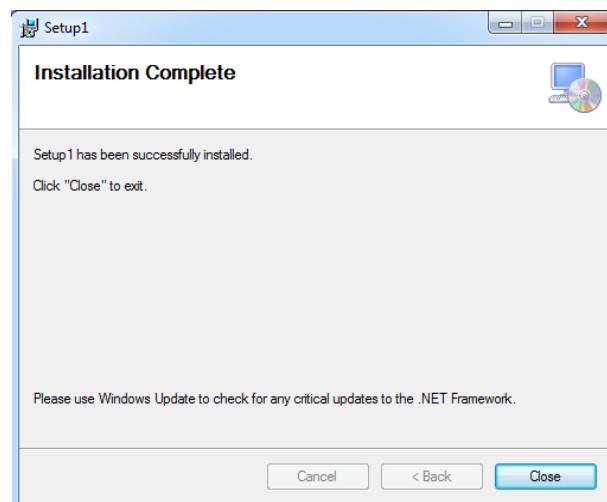
Rys. 5.1.1.1.3. Wybór katalogu do instalacji.



Rys. 5.1.1.1.4. Potwierdzenie instalacji.



Rys. 5.1.1.1.5. Instalowanie.

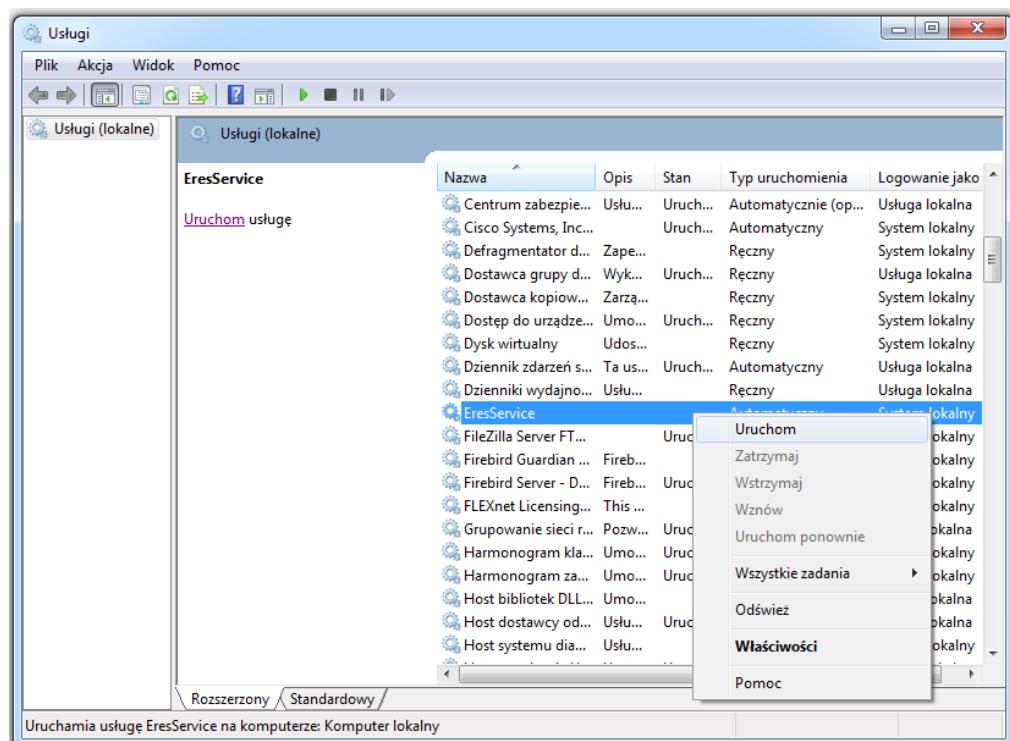


Rys. 5.1.1.1.6. Zakończenie instalacji.

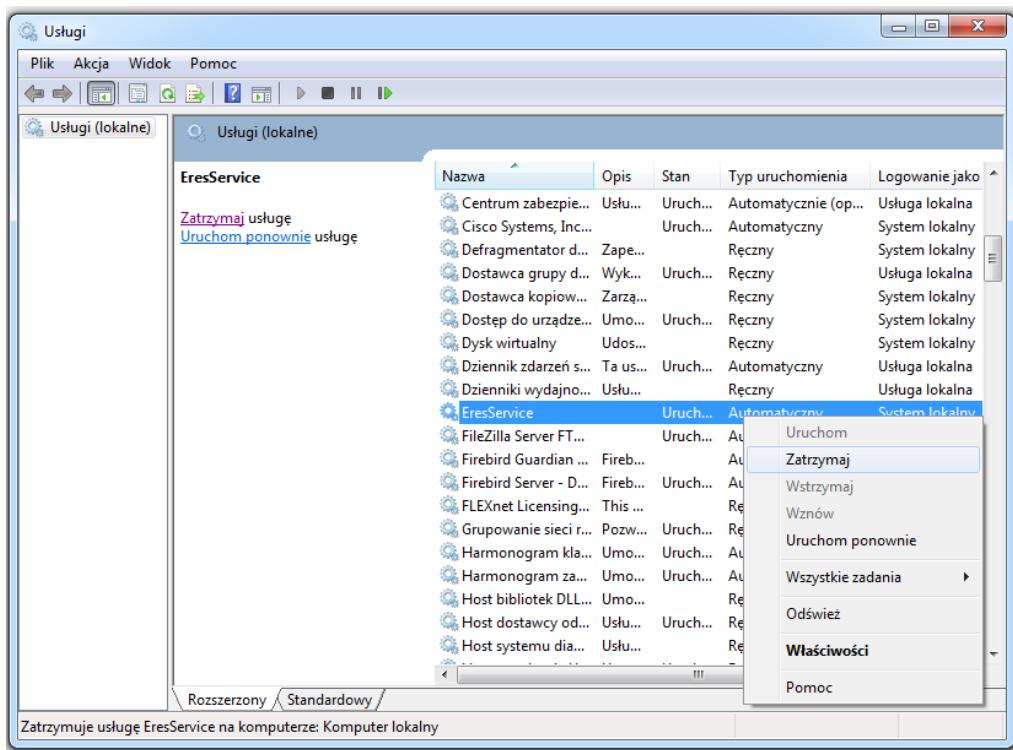
5.1.1.2. Uruchamianie usługi

Aby uruchomić usługę EresService należy przejść do panelu usług systemu Windows: Panel Sterowania -> System i zabezpieczenia -> Narzędzia administracyjne -> Usługi lub wpisując: ‘services.msc’ w Uruchom na pasku startu, a następnie kliknąć opcję ‘Uruchom’ (Rys. 5.1.1.2.1).

Po pierwszym uruchomieniu usługi, usługa będzie uruchamiana automatycznie przy starcie systemu Windows, oraz uruchamiana ponownie w przypadku wystąpienia błędu powodującego jej zamknięcie. Dostępne są również opcje zatrzymania oraz ponownego uruchomienia usługi (Rys. 5.1.1.2.2).



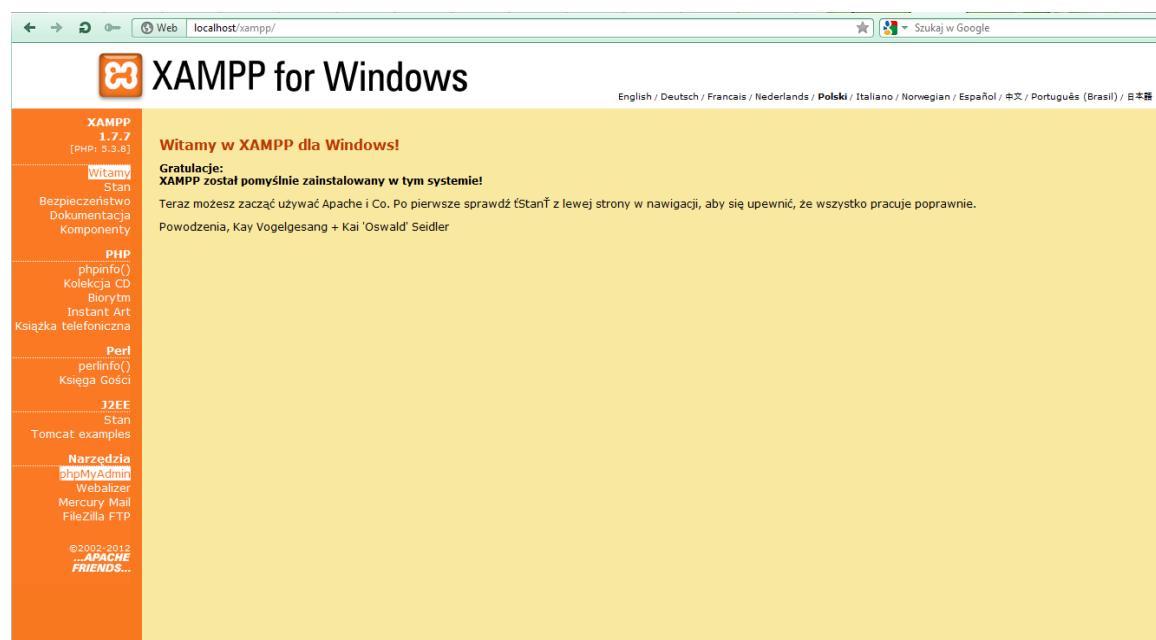
Rys. 5.1.1.2.1. Uruchamianie usługi.



Rys. 5.1.1.2.2. Zatrzymanie lub ponowne uruchomienie usługi.

5.1.2. Import bazy danych (Paweł Gawroński)

Do zainportowania bazy danych wymagany jest zainstalowany program XAMPP, którego można pobrać z oficjalnej strony <http://www.apachefriends.org/en/xampp.html>, na której znajduje się również instrukcja instalacji oraz konfiguracji. Mając zainstalowany program XAMPP przechodzimy do narzędzia phpMyAdmin (Rys. 5.1.2.1).



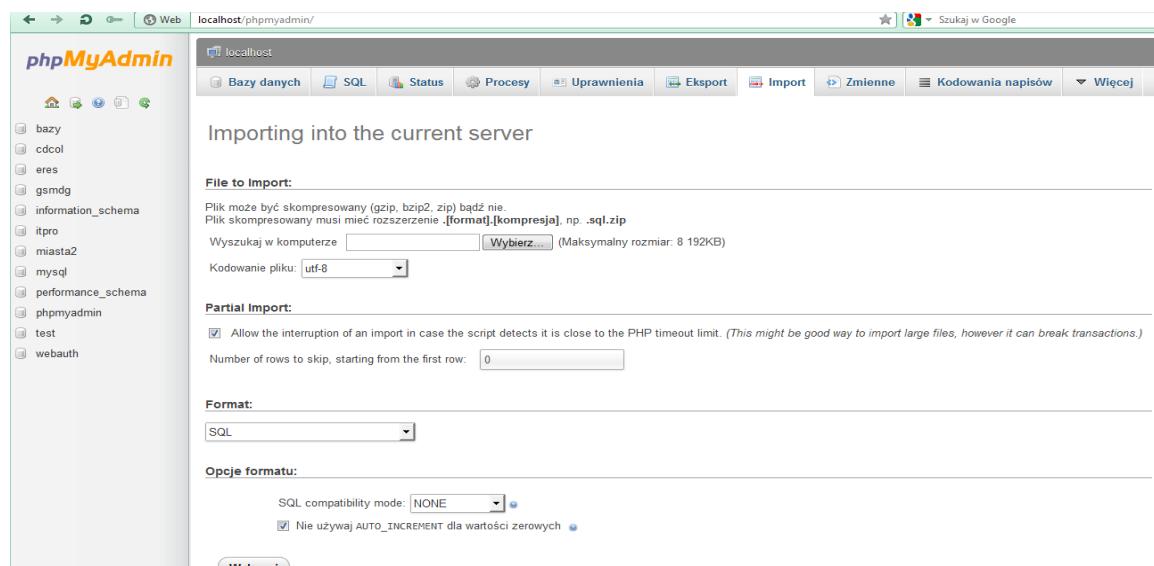
Rys. 5.1.2.1. Okno programu XAMPP.

Usługa EresService do poprawnej komunikacji z bazą danych wymaga stworzenia użytkownika w narzędziu phpMyAdmin o nazwie ‘erestauracja’ i haśle ‘Erestauracja123’ z wszystkimi uprawnieniami. Można to wykonać za pomocą zapytania (Kod 5.1.2.1):

```
CREATE USER 'erestauracja'@'localhost' IDENTIFIED BY 'Erestauracja123';
GRANT ALL PRIVILEGES ON * . * TO 'erestauracja'@'localhost' IDENTIFIED BY 'Erestauracja123'
WITH GRANT OPTION MAX_QUERIES_PER_HOUR 0 MAX_CONNECTIONS_PER_HOUR 0
MAX_UPDATES_PER_HOUR 0 MAX_USER_CONNECTIONS 0;
```

Kod 5.1.2.1. Tworzenie nowego użytkownika w narzędziu phpMyAdmin.

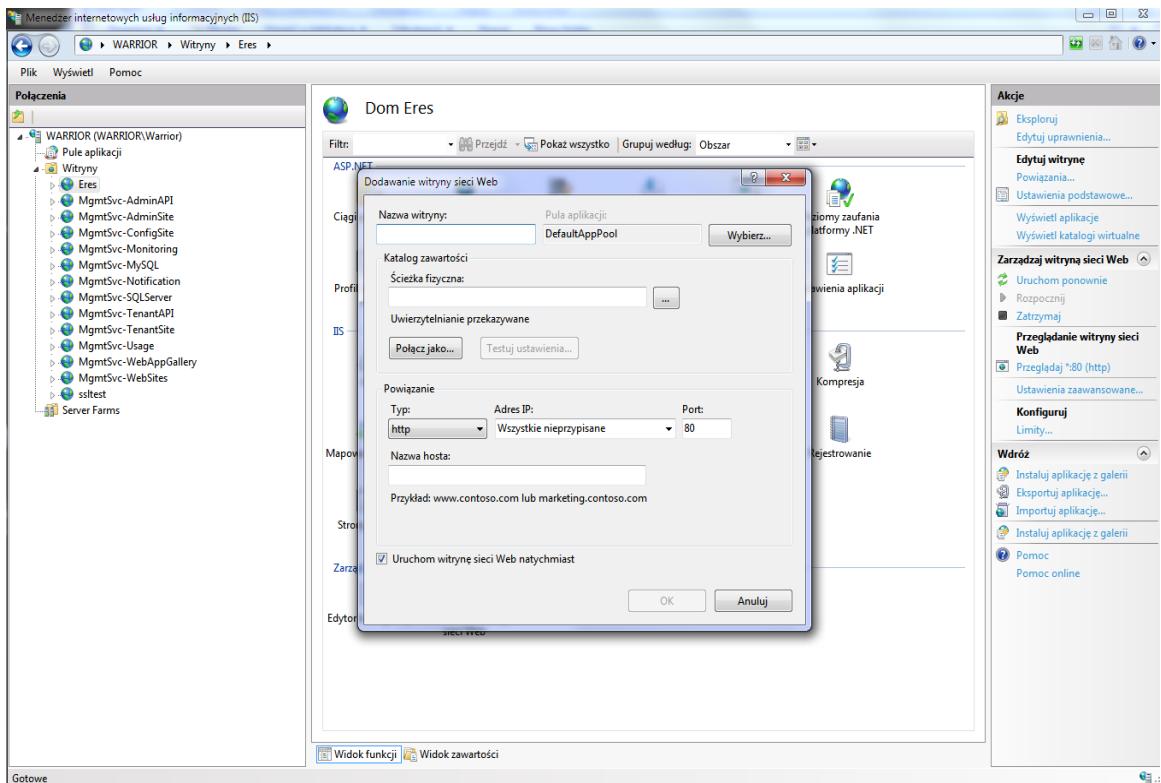
Import bazy danych odbywa się w zakładce „Import” (Rys. 5.1.2.2), należy kliknąć przycisk „Wybierz”, a następnie wybrać plik zawierający bazę danych. Po kliknięciu „Wykonaj” baza o nazwie ‘eres’ zostanie utworzona.



Rys. 5.1.2.2. Okno importu bazy danych.

5.1.3. IIS (Dariusz Kohlandt)

Aplikację można uruchomić poprzez Visual Studio 2010 (wymagany .NET 4.0) lub poprzez samodzielna konfigurację serwera IIS7, który został wcześniej zainstalowany i włączony.



Rys. 5.1.3.1. Menedżer internetowych usług informacyjnych.

W tym celu należy przejść do Panel sterowania -> Narzędzia administracyjne -> Menedżer internetowych usług informacyjnych (IIS) (Rys. 5.1.3.1.). Można to również zrobić wpisując polecenie ‘inetmgr’ w polu ‘Uruchom’ wybranego z Menu start. Z rozwijanego menu w lewym panelu wybieramy ‘Witryny’. Klikamy PPM i wybieramy ‘Dodaj witrynę sieci Web’. W oknie, które się otworzy wprowadzamy ścieżkę fizyczną do witryny ‘Erestauracja/Erestauracja’. Dodajemy witrynę przyciskiem OK. W pulach witryn wybieramy pulę, która albo się utworzyła sama albo wybraliśmy podczas dodawania nowej witryny i sprawdzamy, czy wersja architektury .NET jest ustawiona na 4.0. Jeżeli tak jest to przechodzimy do następnego kroku. W tym momencie częstym problemem jest brak uprawnień do katalogu z witryną. Można je nadać na przykład otwierając menu ‘uwierzytelnianie’ po wybraniu naszej witryny. Prawym przyciskiem myszy edytujemy wpis ‘Uwierzytelnianie anonimowe’ i zmieniamy na ‘Tożsamość puli aplikacji’. Czasem przy pierwszej konfiguracji i braku zainstalowanych wymaganych bibliotek pomaga komenda:

```
%windir%\Microsoft.NET\Framework64\v4.0.30319\aspnet_regiis.exe -ir  
wpisana w polu ‘uruchom’.
```

5.1.4. Konta dostępowe (Dariusz Kohlandt)

Konta dostępowe służą do przetestowania pełnych możliwości aplikacji.

Dostęp do panelu administratora:

login: admin,

hasło: admin1;

Dostęp do panelu klienta:

login: test,

hasło: 123456;

Dostęp do panelu menadżera:

login: test2,

hasło: 123456;

Dostęp do panelu restauracji:

login: 4pory,

hasło: 123456.

Pracownik tej restauracji:

login: test3,

hasło: 123456;

Dane do kont deweloperskich PayPal (<https://developer.paypal.com/>):

Login główny: erestauracja@gmail.com,

hasło: Erestauracja123;

Login do testowego konta kupującego:

eres_1354279048_per@gmail.com,

hasło: 12345678;

Login do testowego konta restauracji 4 pory roku:

4pory_1354887153_biz@gmail.com,

hasło: 12345678;

Aby dokonać płatności w naszej aplikacji za pośrednictwem tych kont, trzeba najpierw zalogować się na konto główne, a przy wykonywaniu płatności logować się na konto testowe będąc zalogowanym na koncie głównym deweloperskim do serwisu PayPal.

5.2. Instrukcja obsługi (Dariusz Kohlandt)

Instrukcję obsługi aplikacji zostały stworzone w ramach projektu realizowanego przez grupę (Gawroński Paweł, Kohlandt Dariusz, Langowski Piotr) na przedmiocie Oprogramowanie Aplikacyjne. Obecna wersja instrukcji obsługi została stworzona na podstawie dokumentów opracowanych przez jednego z członków grupy – Piotra Langowskiego.

5.2.1. Instrukcja dla klienta (Dariusz Kohlandt)

Instrukcja dla klienta jest dołączona w postaci załącznika do pracy (Załącznik 1 – Manual – Klient.pdf).

5.2.2. Instrukcja dla menadżera (Paweł Gawroński)

Instrukcja dla menadżera jest dołączona w postaci załącznika do pracy (Załącznik 2 – Manual – Menadżer.pdf).

5.2.3. Instrukcja dla restauracji (Paweł Gawroński)

Instrukcja dla restauracji jest dołączona w postaci załącznika do pracy (Załącznik 3 – Manual – POS.pdf).

5.2.4. Instrukcja dla administratora (Dariusz Kohlandt)

Instrukcja dla administratora jest dołączona w postaci załącznika do pracy (Załącznik 4 – Manual – Admin.pdf).

6.Testy (Dariusz Kohlandt)

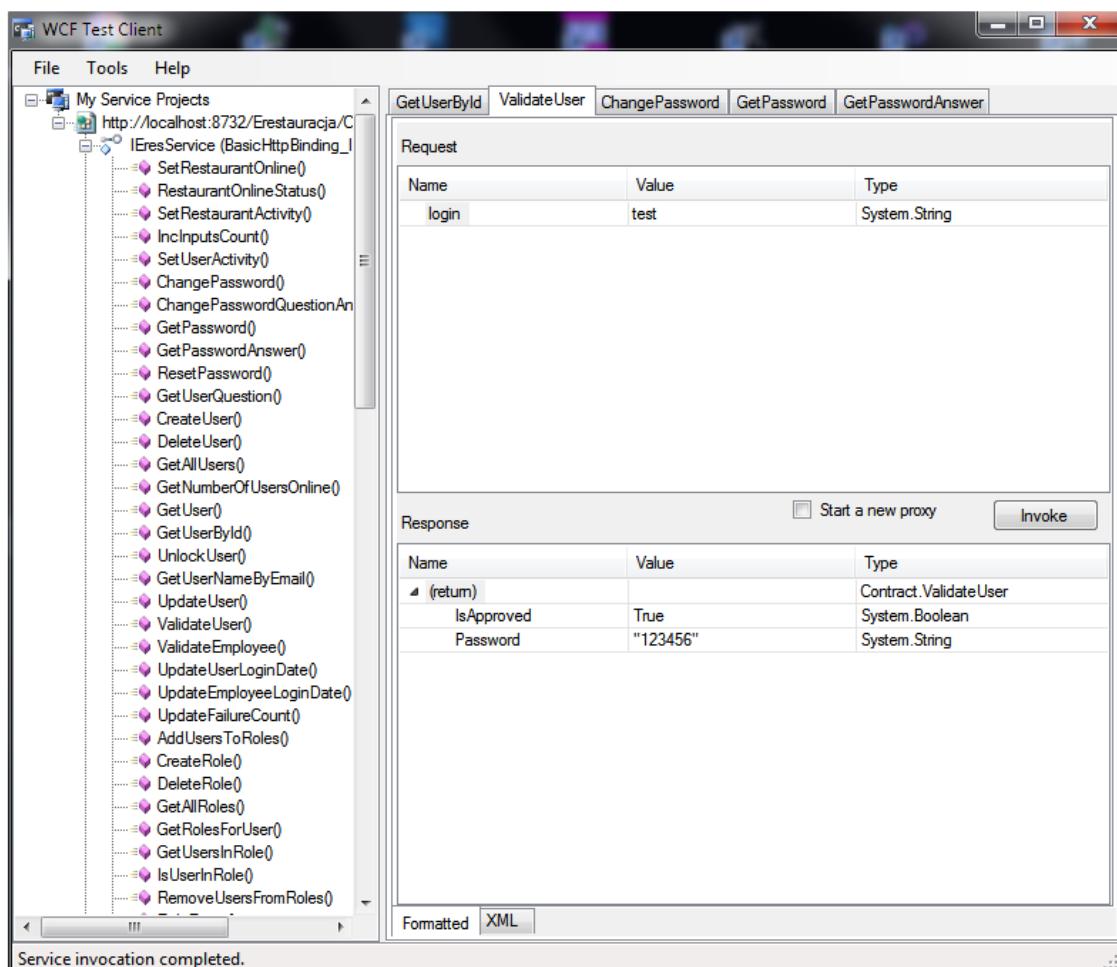
Ograniczenia czasowe nie pozwoliły nam na zastosowanie testów jednostkowych. Jednak do przetestowania modułów naszego systemu każda metoda pisana w języku C# była sprawdzana debuggerem wbudowanym w środowisko Microsoft Visual Studio 2010 (Rys. 6.1).

```
HomeController.cs  X Register.aspx  POSController.cs  AccountController.cs  Web.config
Erestauracja.Controllers.HomeController  Index()
```

```
9  using System.Web.Security;
10 using Erestauracja.Authorization;
11 using Erestauracja.Models;
12 using Erestauracja.Helpers;
13 using Erestauracja.ServiceReference;
14 using Erestauracja.Providers;
15
16 namespace Erestauracja.Controllers
17 {
18
19     public class HomeController : Controller
20     {
21         public ActionResult Index()
22         {
23             CustomRoleProvider rp = new CustomRoleProvider();
24             if (User.Identity.IsAuthenticated)  User.Identity.IsAuthenticated false
25             {
26                 if (!rp.IsUserInRole(User.Identity.Name, "Klient"))
27                 {
28                     return RedirectToAction("Unauthorized");
29                 }
30             }
31
32             List<RestaurantTop> value = null;
33             Statistics stats = null;
34             try
35             {
36                 ServiceReference.EresServiceClient client = new ServiceReference.EresServiceClient();
37                 using (client)
38                 {
39                     value = new List<RestaurantTop>(client.GetTopRestaurant());
40                     stats = client.GetStatistics();
41                 }
42                 client.Close();
43             }
44             catch (Exception e)
```

Rys. 6.1. Debugger w kontrolerze Home (opracowanie własne).

Metody, które zostały napisane w serwisie WCF testowane były za pomocą WCF Test Client w środowisku Microsoft Visual Studio 2010 (Rys. 6.2).



Rys. 6.2. WCF Test Client (Visual Studio 2010).

Gotowe moduły i funkcjonalności testowaliśmy na uruchomionej aplikacji imituając zachowanie użytkowników końcowych.

Wartości w różnych komórkach bazy danych były przez nas modyfikowane, by poznać reakcje aplikacji na niedozwolone wartości pobierane z bazy danych.

PODSUMOWANIE (Dariusz Kohlandt)

Celem pracy było stworzenie portalu internetowego przy pomocy technologii ASP.NET wraz ze wzorcem MVC3 oraz serwisem WCF. Ów cel został osiągnięty, a napisana aplikacja internetowa jest w stanie sprawnie wykonywać założone przez nas funkcjonalności.

W ramach niniejszego projektu inżynierskiego zaimplementowaliśmy system składania zamówień i zarządzania nimi. System ten pod każdym względem został przez nas dokładnie przemyślany i zaprojektowany. Dla menadżerów został stworzony panel do administracji restauracjami i treściami dla klientów widocznymi w portalu. Dodatkowo, menadżerowie mogą dodawać do obsługi restauracji nowych pracowników oraz edytować ich dane. Mogą również tworzyć i edytować menu swoich lokali. Dla restauracji został utworzony komponent przeznaczony do obsługi zamówień przychodzących za pomocą panelu dotykowego. Klienci mają do dyspozycji możliwość wyszukiwania restauracji oraz przeglądania jej menu, galerii oraz innych udostępnionych informacji na stronie lokalu. Klient może również utworzyć sobie konto, z którego będzie składać zamówienia. Bardzo przydatną funkcjonalnością jest możliwość podejrzenia rzeczywistej lokalizacji restauracji na mapie ze znacznikami. Mapa pomaga również przy rejestracji lub edycji danych, w wypadku podania nieprawidłowego kodu pocztowego lub nazwy miasta przez użytkownika.

W pracy stworzony został również system komentarzy i oceniania restauracji. Dla ułatwienia stworzyliśmy koszyk dla klienta służący do składania zamówienia. Panel szybkiego wyszukiwania pomaga użytkownikom w znalezieniu restauracji w danym mieście. Dodatkowo został on rozszerzony o stronę z wyszukiwaniem zaawansowanym. W celu rozwijania aplikacji i pozbywania się niedoskonałości umożliwiliśmy użytkownikom zgłaszanie błędów przez specjalny formularz.

Na potrzeby aplikacji napisaliśmy własnego dostawcę członkostwa oraz ról zarządzającego użytkownikami i rolami. Dzięki temu uzyskaliśmy możliwość rejestracji, logowania, zmiany hasła i innych mniej lub bardziej ważnych funkcji.

Sam interfejs użytkownika został przemyślany i zaprojektowany w taki sposób, by użytkownik nie miał problemów w odnalezieniu w nim potrzebnych funkcji systemu, które są mu udostępnione.

W aplikacji zastosowaliśmy system płatności PayPal, który wydawał się nam najbardziej popularny z dostępnych, włączając w to karty kredytowe i debetowe. System PayPal jest też odpowiednio zabezpieczony, co czyni go godnym zaufania.

Na potrzeby aplikacji stworzyliśmy własnoręcznie bazę danych miast i kodów pocztowych, która była nam potrzebna do pobrania współrzędnych i ustawiania znaczników na mapie. Baza ta obejmuje wszystkie miejscowości w Polsce, włączając w to dzielnice większych miejscowości.

Wszystkie połączenia zabezpieczyliśmy tworząc własną usługę systemu Windows, która przekazywała dane pomiędzy bazą danych a aplikacją.

Poznaliśmy wzorzec MVC zastosowany w ASP.NET, który mimo iż na początku stwarzał nam problemy, to z czasem okazał się bardzo przyjaznym narzędziem. Dzięki licznym zaletom, technologia ASP.NET jest bardzo konkurencyjna na rynku tworzenia stron internetowych, lecz poważnym jej minusem jest komercyjne środowisko Visual Studio, bez którego aplikacja nie mogłaby powstać.

Stworzona aplikacja jest bardzo konkurencyjna na rynku, ponieważ pod względem funkcjonalności nie ma sobie równych, dlatego też w przyszłości planujemy zawiązać współpracę z producentami oprogramowania dla branży gastronomicznej. W ten sposób chcemy połączyć nasz produkt z ich systemami, dostarczając klientom końcowym gotowe rozwiązańe do obsługi całego łańcucha konsumenckiego.

Podział pracy:

	Kohlandt Dariusz	Gawroński Paweł
Dokumentacja:	Wstęp; Rozdział 1: 1.2, 1.3, 1.4; Rozdział 2: 2.1 (drugi akapit), 2.3 (ostatni akapit), 2.5 (drugi akapit), 2.6, 2.7, 2.8; Rozdział 3: Opracowanie wspólne; Rozdział 4: 4.1.15-16, 4.2.1, 4.2.4-8, 4.3.1-2; Rozdział 5: 5.1.3-4, 5.2(Instrukcja dla klienta oraz administratora) Rozdział 6; Podsumowanie;	Rozdział 1: 1.1; Rozdział 2: 2.1(pierwszy akapit), 2.2, 2.3(bez ostatniego akapitu), 2.4, 2.5(pierwszy akapit); Rozdział 3: Opracowanie wspólne; Rozdział 4: 4.1.1-14, 4.2.2-3, 4.3.3; Rozdział 5: 5.1.1-2, 5.2(Instrukcja dla menadżera oraz POSu);
Aplikacja:	Własna autoryzacja użytkowników; Implementacja Google Maps; Implementacja PayPal; Ustawianie zawartości strony za pomocą CSS, Html, JS oraz JQuery; Implementacja komunikacji z chmurą danych DropBox; Konfiguracja środowiska uruchomieniowego ;	Kierowanie projektem; Serwis WCF wraz z instalatorem (Projekty: Contract, Host oraz Setup1); Własny model dostawców; Projektowanie panelu menadżera oraz klienta; Funkcjonalności: zgłaszanie błędów, wyświetlanie informacji o stronie, zakładanie konta menadżera,

	<p>Projektowanie panelu administratora oraz POSu;</p> <p>Wszystkie skrypty napisane w JavaScript;</p> <p>Funkcjonalności: wyświetlanie instrukcji obsługi na stronie, szybkie wyszukiwanie na stronie głównej, galeria oraz zdjęcie główne na stronie restauracji, panel administratora(całość), panel POS(prócz logowania);</p>	<p>wyświetlanie statystyk na stronie głównej, lista '10 najnowszych restauracji' na stronie głównej, wyszukiwanie restauracji, logowanie, rejestracja użytkowników, odzyskiwanie hasła, strona restauracji(prócz galerii oraz zdjęcia głównego), koszyk, realizacja zamówień (oprócz PayPal), wyświetlanie aktualnych zamówień, edycja danych użytkownika, zmiana hasła, wyświetlanie historii zamówień, system komentarzy, panel menadżera(całość), logowanie w panelu POS;</p>
--	--	--

LITERATURA

- [1] Bill Evjen, Scott Hanselman, Devin Rader, *ASP.NET 4 z wykorzystaniem C# i VB*, Gliwice : Wydawnictwo HELION, 2011.
- [2] http://en.wikipedia.org/wiki/Microsoft_Visual_Studio [dostęp: 4 stycznia 2013].
- [3] http://pl.wikipedia.org/wiki/Microsoft_Visual_Studio [dostęp: 4 stycznia 2013].
- [4] What's New in Visual Studio 2010 [http://msdn.microsoft.com/en-us/library/bb386063\(v=vs.100\).aspx](http://msdn.microsoft.com/en-us/library/bb386063(v=vs.100).aspx) [dostęp: 4 stycznia 2013].
- [5] <http://monodevelop.com/> [dostęp: 4 stycznia 2013].
- [6] <http://en.wikipedia.org/wiki/PhpMyAdmin> [dostęp: 4 stycznia 2013].
- [7] <http://en.wikipedia.org/wiki/XAMPP> [dostęp: 4 stycznia 2013].
- [8] http://en.wikipedia.org/wiki/Virtual_machine [dostęp: 4 stycznia 2013].
- [9] <https://task.cloud.pionier.net.pl/> [dostęp: 4 stycznia 2013].
- [10] [http://en.wikipedia.org/wiki/C_Sharp_\(programming_language\)](http://en.wikipedia.org/wiki/C_Sharp_(programming_language)) [dostęp: 4 stycznia 2013].
- [11] What's New in Visual C# 2010 [http://msdn.microsoft.com/en-us/library/bb383815\(v=vs.100\).aspx](http://msdn.microsoft.com/en-us/library/bb383815(v=vs.100).aspx) [dostęp: 4 stycznia 2013].
- [12] Wiktor Zychla, *Programowanie pod Windows*, Instytut Informatyki Uniwersytetu Wrocławskiego
- [13] http://en.wikipedia.org/wiki/.NET_Framework [dostęp: 4 stycznia 2013].
- [14] What's New in the .NET Framework 4 [http://msdn.microsoft.com/en-us/library/vstudio/ms171868\(v=vs.100\).aspx](http://msdn.microsoft.com/en-us/library/vstudio/ms171868(v=vs.100).aspx) [dostęp: 4 stycznia 2013].
- [15] http://www.mono-project.com/Main_Page [dostęp: 4 stycznia 2013].
- [16] Wprowadzenie do wzorca projektowego Model-View-ViewModel na przykładzie aplikacji WPF <http://msdn.microsoft.com/pl-pl/library/wprowadzenie-do-wzorca-projektowego-model-view-viewmodel-na-przykładzie-aplikacji-wpf.aspx> [dostęp: 4 stycznia 2013].
- [17] <http://en.wikipedia.org/wiki/Model–view–presenter> [dostęp: 4 stycznia 2013].
- [18] MATULEWSKI, Jacek i in. *Visual Studio 2010 dla programistów C#*. Gliwice : Wydawnictwo HELION, 2011.
- [19] What's New in WCF 4? http://msdn.microsoft.com/en-us/vs2010trainingcourse_whatsnewinwcf4.aspx [dostęp: 4 stycznia 2013].
- [20] <http://en.wikipedia.org/wiki/MySQL> [dostęp: 4 stycznia 2013].

- [21] <http://en.wikipedia.org/wiki/JavaScript> [dostęp: 4 stycznia 2013].
- [22] <http://en.wikipedia.org/wiki/Jquery> [dostęp: 4 stycznia 2013].
- [23] http://en.wikipedia.org/wiki/Cloud_storage [dostęp: 4 stycznia 2013].
- [24] <http://en.wikipedia.org/wiki/Ftp> [dostęp: 4 stycznia 2013].
- [25] <http://en.wikipedia.org/wiki/Html> [dostęp: 4 stycznia 2013].
- [26] <http://en.wikipedia.org/wiki/Css> [dostęp: 4 stycznia 2013].
- [27] <http://api.jqueryui.com/category/widgets/> [dostęp: 4 stycznia 2013].
- [28] <http://sharpbox.codeplex.com/> [dostęp: 4 stycznia 2013].
- [29] <http://wbotelhos.com/raty/> [dostęp: 4 stycznia 2013].
- [30] <http://jhtmlarea.codeplex.com/> [dostęp: 4 stycznia 2013].
- [31] <http://jquerypriceformat.com> [dostęp: 4 stycznia 2013].
- [32] <http://www.jquery4u.com/plugins/jquery-screen-keyboard-plugin/> [dostęp: 4 stycznia 2013].
- [34] <http://googlemapmvc.codeplex.com/> [dostęp: 4 stycznia 2013].
- [35] <http://www.yoxigen.com/yoxview/> [dostęp: 4 stycznia 2013].

ZAŁĄCZNIKI

Załącznik 1 – Manual – Klient

Załącznik 2 – Manual – Menadżer

Załącznik 3 – Manual – POS

Załącznik 4 – Manual – Admin

Załącznik 5 – Płyta CD

E.Postrumagia

Instrukcja Obsługi



Dla Klientów

Szanowny Kliencie

Opracowanie to zostało stworzone, aby ułatwić państwu korzystanie z usług systemu E-Restauracja.

Prosimy o korzystanie z oprogramowania w sposób opisany w poniższej instrukcji obsługi. Wszelkie odstępstwa od niej mogą prowadzić do usterek.

Oprogramowanie zostało dokładnie sprawdzone oraz przetestowane pod względem bezpieczeństwa oraz funkcjonalności.

Przed rozpoczęciem korzystania z serwisu należy uważnie przeczytać dostarczoną do niego dokumentację i zachować ją w celu użycia w przyszłości.

Wstęp

Portal E-Restauracja to doskonałe narzędzie do zamawiania swoich posiłków w sieci internetowej. W bardzo wygony sposób można odnaleźć naszą ulubioną restaurację, zamówić danie, zapłacić za nie oraz śledzić status złożonego zamówienia. Wiedząc, jaki status realizacji posiada nasze zamówienie, pozostaje nam czekać, aż zostanie ono do nas dostarczone, a wszystko to z przed własnego komputera.

Portal E-Restauracja to uniwersalne narzędzie zarówno dla klientów jak i menadżerów oraz pracowników restauracji.

Mamy nadzieję, że będą Państwo zadowoleni z korzystania z naszego oprogramowania.

1. Strona główna

The screenshot shows the homepage of the E-Restauracja website. At the top, there is a navigation bar with a logo, the text "E-Restauracja", and a "Zaloguj" button. Below the navigation bar, there is a horizontal menu with links: "Strona główna", "Szukaj", "Konto", "Koszyk", and "Aktualne zamówienia".

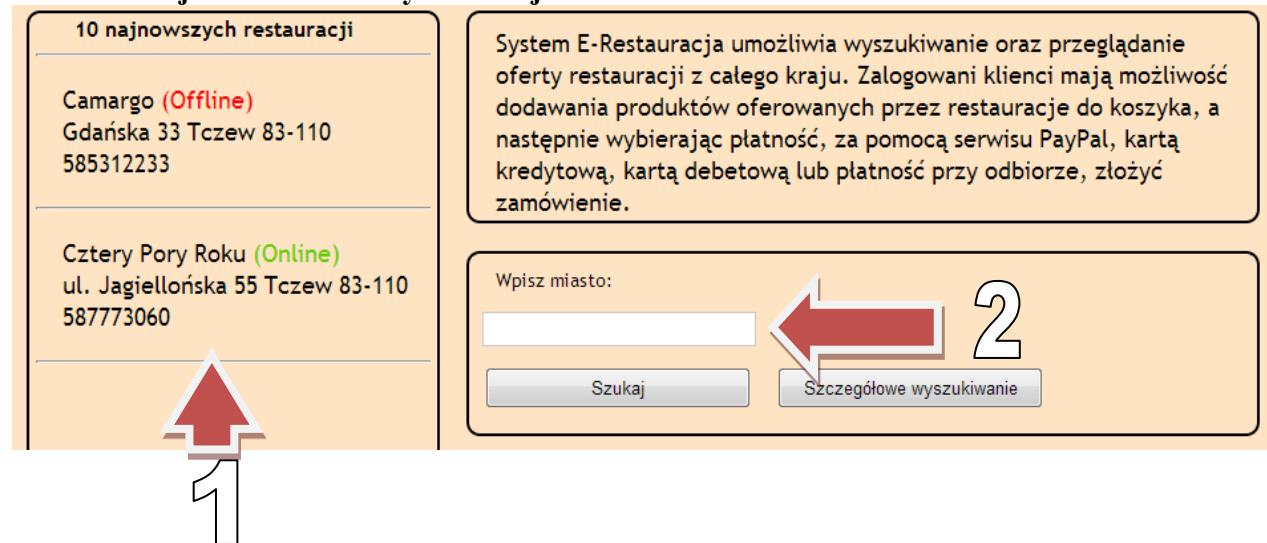
The main content area features several sections:

- 10 najnowszych restauracji**: A list of two restaurants:
 - Camargo (Offline)**
Gdańska 33 Tczew 83-110
585312233
 - Cztery Pory Roku (Online)**
ul. Jagiellońska 55 Tczew 83-110
587773060
- A descriptive text block explaining the system's features, mentioning offline and online restaurants, and payment methods.
- A search form with a text input field labeled "Wpisz miasto:", a "Szukaj" button, and a "Szczegółowe wyszukiwanie" link.
- A summary section with statistics: 19 products, 2 restaurants, 19 users, and 1 active user.

At the bottom, there is a footer with a logo, links to "Dodaj swoją restaurację", "Pomoc", "Informacje", and "Zgłaszczenie błędów", and a copyright notice: "© 2012 Erestauracja".

1.1. Niezalogowany użytkownik

1.1.1. Funkcjonalności Strony Głównej

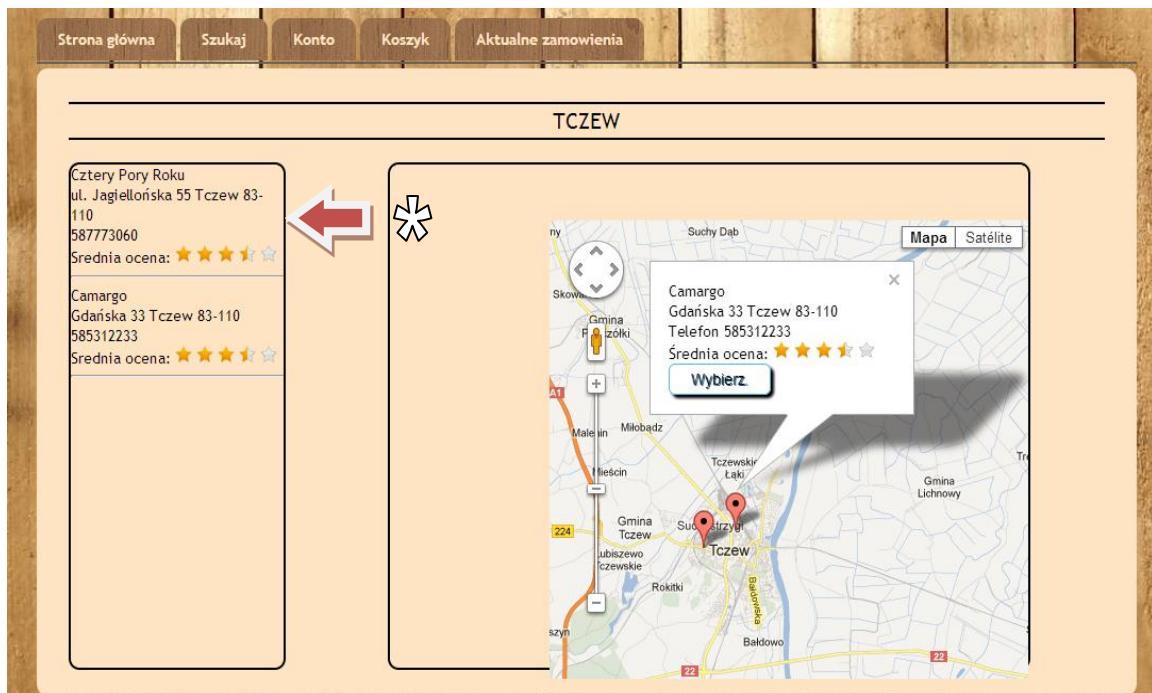


1. Po lewej stronie strony głównej znajduje się lista „10 najnowszych restauracji” (1). Możemy przejść do wybranej restauracji klikając na jej nazwę.

2. Po prawej stronie (2), znajduje się wyszukiwanie restauracji po nazwie miasta.



- a. Wpisujemy w pole (a) nazwę miasta.
- b. Pojawi się rozwijana lista (b), z której wybieramy restauracje – zostaniemy przekierowani do strony restauracji.
- c. Wybierając „Wszystkie” otwiera się strona z mapą, na której widnieją znaczniki restauracji:

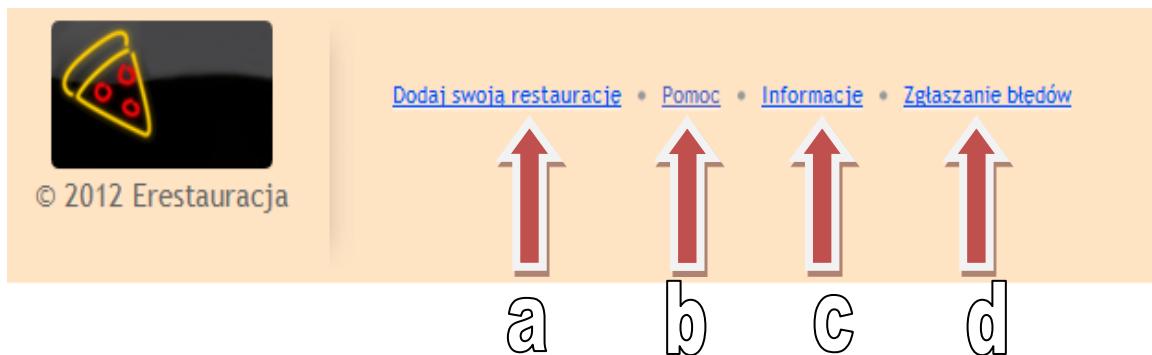


- Po lewej (*) mamy listę restauracji**

3. Na górze strony mamy pasek dostępu do podstron (1). Z tego miejsca również możemy się zalogować (2).



4. Stopka



1. Wybierając opcję dodaj swoją restaurację (a) zostaniemy przekierowani do strony zakładania konta menadżerskiego.

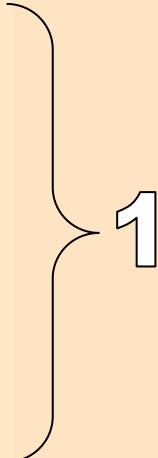
Uwaga!
Aby dodać nową restaurację należy stworzyć konto menedżera, które umożliwia nie tylko dodawanie nowych lokali ale również zarządzanie nimi oraz personelem.
Wykorzystując w tym celu istniejące konto, stracisz możliwość wykonywania zamówień.

Utwórz nowe konto menedżera wypełniając poniższy formularz lub wykorzystaj [istniejące konto](#)

Dane rejestracji

Wprowadź swoje dane, a następnie kliknij 'Załóż konto'.

Login	<input type="text"/>
Email	<input type="text"/>
Powtórz email	<input type="text"/>
Hasło (Minimum 6 znaków.)	<input type="password"/>
Powtórz hasło	<input type="password"/>
Pytanie do przywracania hasła	<input type="text"/>
Odpowiedz	<input type="text"/>
Imię	<input type="text"/>
Nazwisko	<input type="text"/>
Adres	<input type="text"/>
Miasto	<input type="text"/>
Kod pocztowy	<input type="text"/>
Kraj	Afganistan
Data urodzenia	<input type="text"/>
Płeć	Mężczyzna
Numer telefonu	<input type="text"/>
<input type="button" value="Załóż konto"/>	



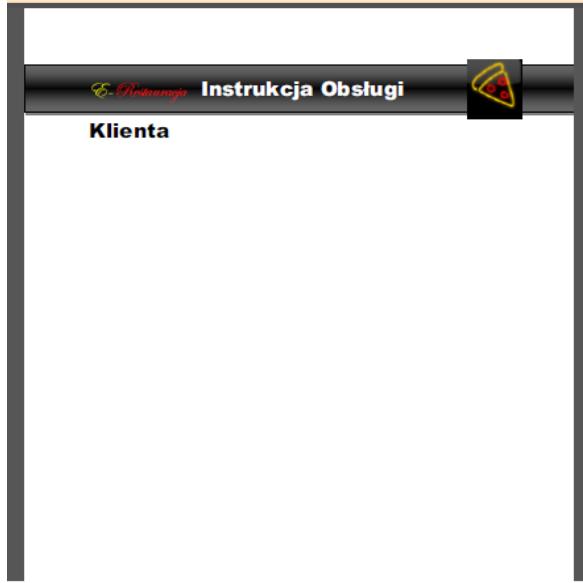
Uwaga!

Aby dodać nową restaurację należy stworzyć konto menedżera, które umożliwia nie tylko dodawanie nowych lokali, ale również zarządzanie nimi oraz personelem.

Wykorzystując w tym celu istniejące konto, stracisz możliwość wykonywania zamówień.

- W tym miejscu wypełniamy dane (1) i zatwierdzamy przyciskiem załóż konto (2).**

2. Wybierając opcję pomoc (b) zostaniemy przekierowani do strony pomocy.



3. Wybierając opcję informacje (c) zostaniemy przekierowani do strony z informacjami.

Informacje o E-Restauracji

System E-Restauracja umożliwia wyszukiwanie oraz przeglądanie oferty restauracji z całego kraju. Zalogowani klienci mają możliwość dodawania produktów oferowanych przez restauracje do koszyka, a następnie wybierając płatność, za pomocą serwisu PayPal, kartą kredytową, kartą debetową lub płatność przy odbiorze, złożyć zamówienie.

Właściciele oraz menadżerowie restauracji mają możliwość dodawania nowych restauracji oraz ich personelu do serwisu. Mają także możliwość edytowania zawartości witryny swojej restauracji, dzięki prostemu w użyciu edytorowi. Możliwość dodawania kategorii oraz produktów pozwala odwzorować niemal każde menu.

Personel restauracji ma do dyspozycji intuicyjny POS (point-of-sale), przystosowany do obsługi na ekranie dotykowym. Daje on możliwość podglądu spływających zamówień oraz zmianę ich statusu, co daje klientowi wiedzę o stanie zamówienia.

Wszystko to można obsługiwać za pomocą popularnych przeglądarek internetowych, takich jak: Chrome, Firefox, IE czy Opera.

Kontakt

Email: erestauracja@gmail.com

4. Wybierając opcję zgłaszanie błędów (d) zostaniemy przekierowani do formularza zgłaszania błędów.

Podaj swój adres email oraz treść zgłoszenia, a następnie kliknij "Wyślij", aby powiadomić nas o znalezionym błędzie.

Formularz zgłoszenia

Email 1

Treść zgłoszenia 2

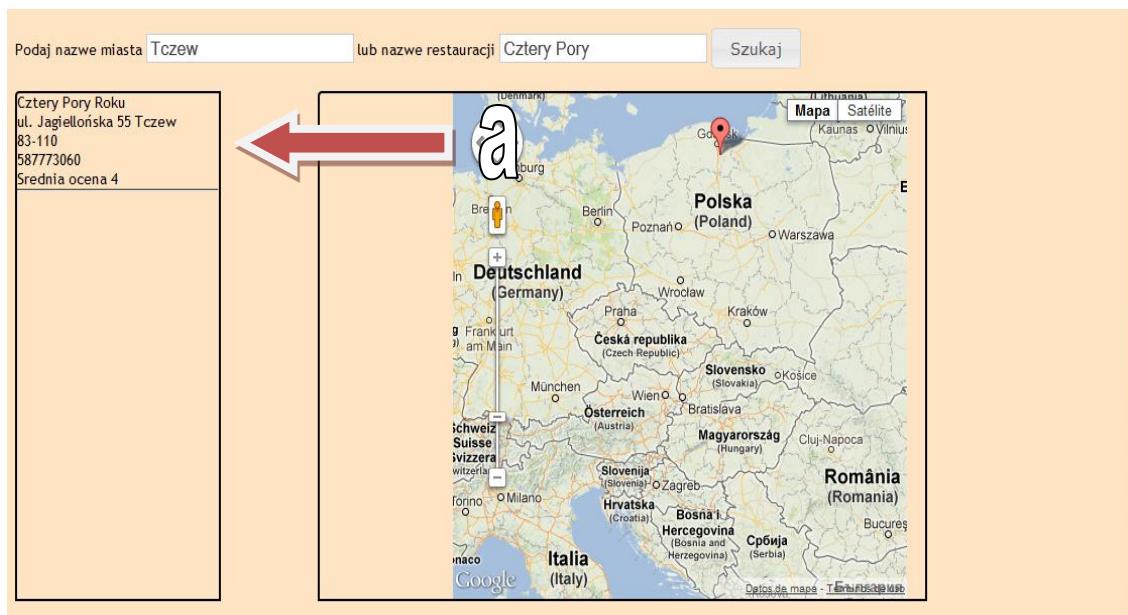
Wyślij 3

- **Wpisujemy swój adres poczty elektronicznej (1),**
- **Wpisujemy treść zgłoszenia (2) tj. szczegóły błędu,**
- **Zatwierdzamy przyciskiem wyslij (3).**

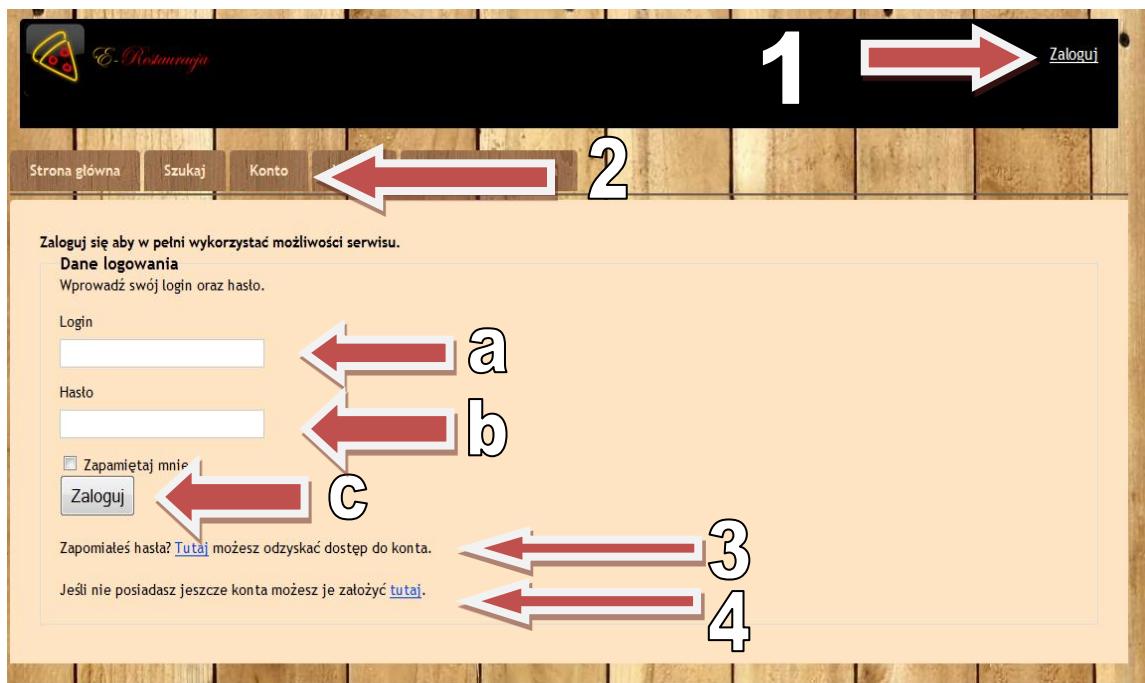
1.1.2. Szukaj (Szukanie zaawansowane)

1. **Wybieramy opcję „Szukaj” (1),**
2. **Wpisujemy nazwę miasta (2),**
3. **Wpisujemy nazwę restauracji (3), (opcjonalnie 2, 3 lub 2 i 3),**
4. **Zatwierdzamy przyciskiem „Szukaj” (4),**

5. Zostaniemy przekierowani do strony gdzie możemy wybrać wyszukiwaną restaurację (a).



1.1.3. Konto



- 1. Aby przejść do panelu logowania wybieramy opcję „Zaloguj” (1) na górze strony, bądź przycisk „Konto” (2) z paska dostępu.**
- 2. Jeśli posiadamy konto wpisujemy login w pole (a) i hasło w (b) oraz zatwierdzamy przyciskiem zaloguj (c).**
- 3. Jeśli zapomnieliśmy hasła przechodzimy do panelu odzyskiwania hasła poprzez kliknięcie „Tutaj” w linii (3).**

This screenshot shows a password reset form. It asks for the user's login and provides a 'Dalej' (Next) button. A red arrow labeled '5' points to the 'test' input field.

a. Tutaj wpisujemy swój login (5) i wybieramy opcję Dalej,

Podaj odpowiedź na pytanie wybrane w czasie rejestracji aby zresetować hasło.

Odzyskiwanie hasła

Login: test

Pytanie: pyt

Odpowiedź:

6

Dalej



b. Trafimy na stronę gdzie będziemy musieli odpowiedzieć na pytanie, które wybraliśmy podczas rejestracji – odpowiedź należy wpisać w pole (6) i zatwierdzić kliknięciem „Dalej”,

Hasło zostało wysłane na adres email podany przy rejestracji. Jeżeli nie dostaniesz hasła w przeciągu 12 godzin, ponów próbę lub skontaktuj się z administratorem.
[Powrót do strony logowania.](#)

c. Zostaniemy poinformowani o wysłaniu nowego hasła na nasz adres poczty elektronicznej (j.w).

4. Aby założyć konto wybieramy opcję „tutaj” (4).

Utwórz swoje konto, jeśli jeszcze go nie posiadasz.

Dane rejestracji

Wprowadź swoje dane, a następnie kliknij 'Załóż konto' aby w pełni wykorzystać możliwości serwisu.

Login	login
Email	langusiek1@wp.pl
Powtórz email	langusiek1@wp.p
Pola Email oraz Powtórz email nie są zgodne.	
Hasło (Minimum 6 znaków.)	•••
Hasło musi zawierać przynajmniej 6 znaków.	
Powtórz hasło	••••••
Pola Hasło oraz Powtórz hasło nie są zgodne.	
Pytanie do przywracania hasła	pytanie
Odpowiedz	odpowiedz
Imię	imie
Nazwisko	nazwisko
Adres	adres 1
Miasto	tczew
Kod pocztowy	83-110
Kraj	Polska
Data urodzenia	01/12/2012
Płeć	Mężczyzna
Numer telefonu	123456
<input type="button" value="Załóż konto"/>	

a **b** **c** **1**



2

- a. Zostaniemy przekierowani do formularza rejestracji – należy wypełnić pola (1) zgodnie z opisem (w innym przypadku pojawią się błędy (a, b, c)) i zatwierdzamy przyciskiem „Załóż konto” (2),
- b. Gdy wszystkie dane są poprawne zostaniemy przeniesieni do strony głównej, jako zalogowany użytkownik.

1.2. Zalogowany użytkownik

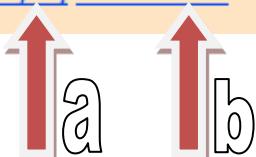


1. Podstrony „Strona główna” i „Szukaj” działają tak samo jak w przypadku niezalogowanego użytkownika.

2. Panel użytkownika – Konto

a. Dane użytkownika

Dane Użytkownika
Kliknij przycisk 'Edytuj', aby wprowadzić zmiany do swoich danych, lub przycisk 'Zmień hasło', aby zmienić hasło do swojego konta.
Login: login
Adres email: langusiek1@wp.pl
Imię: imie
Nazwisko: nazwisko
Adres: adres 1
Miasto: Tczew
Kod pocztowy: 83-110
Kraj: Polska
Data urodzenia: 01/12/2012
Płeć: Mężczyzna
Numer telefonu: 123456
• [Edytuj](#) [Zmień hasło](#)



- **Przechodzimy do strony „Dane Użytkownika” (1)**

1.2.1. Edycja danych

Aby Edytować dane użytkownika wciskamy (a) na powyższym obrazku.

The screenshot shows a user profile editing interface with three tabs at the top: 'Dane użytkownika' (selected), 'Historia zamówień', and 'Komentarze'. The main area is titled 'Edycja danych' and contains instructions: 'Wprowadź nowe dane, a następnie kliknij 'Zapisz''. Below are input fields for various personal details:

Email	paul474@wp.pl
Imię	Jan
Nazwisko	Kowalski
Adres	Główna 3
Miasto	Tczew
Kod pocztowy	83-110
Kraj	Polska ▾
Data urodzenia	13/06/1990
Płeć	Mężczyzna
Numer telefonu	0987654321

Annotations:

- An arrow labeled '1' points to the 'Email' field.
- A large red double-headed arrow labeled '2' points to the 'Zapisz' button.

Edytujemy interesujące nas dane (1) i zatwierdzamy przyciskiem (2).

1.2.2. Zmiana hasła

- **Aby zmienić hasło wciskamy (b) na obrazku z danymi użytkownika,**

Zmiana hasła
Wprowadź dotychczasowe i nowe hasło oraz kliknij 'Zatwierdź' w celu zmiany.

Dotychczasowe hasło

Nowe hasło (Minimum 6 znaków.)

Powtórz hasło

Zatwierdź

The form consists of four input fields. The first field is labeled 'Dotychczasowe hasło'. The second and third fields are grouped together under the label 'Nowe hasło (Minimum 6 znaków.)' and have a curly brace between them. The fourth field is labeled 'Powtórz hasło'. Below the fields is a grey button labeled 'Zatwierdź'. Three red arrows point from the text labels '1', '2', and '3' to the respective fields and button. A curly brace groups the 'Nowe hasło' and 'Powtórz hasło' fields.

Wpisujemy dotychczasowe hasło (1) następnie wpisujemy nowe i ponownie poniżej (2), zatwierdzamy przyciskiem (3).

1.2.3. Historia zamówień

Dane użytkownika Historia zamówień 1
Od: 02/12/2012 Do: 09/12/2012 Filtruj b
▼ Numer:35 Camargo Razem: 30,40 zł Płatność przy odbiorze Status: Oczekujące a
Dane restauracji:
Adres: Gdańska 33 - Tczew (83-110)
Kontakt: 585312233
Przewidywany czas dostawy: 00:45:00
Koszt dostawy: 2,50 zł
Data złożenia zamówienia: 05/12/2012 10:29:32
Produkty:
Kebab x 1 - Czosnkowy
Skrzydełka z grilla x 1 - Ziemiaki
Pomidorowa x 1
▼ Numer:36 Cztery Pory Roku Razem: 23,50 zł Płatność przy odbiorze Status: Oczekujące
▼ Numer:38 Camargo Razem: 15,00 zł Płatność przy odbiorze Status: Oczekujące

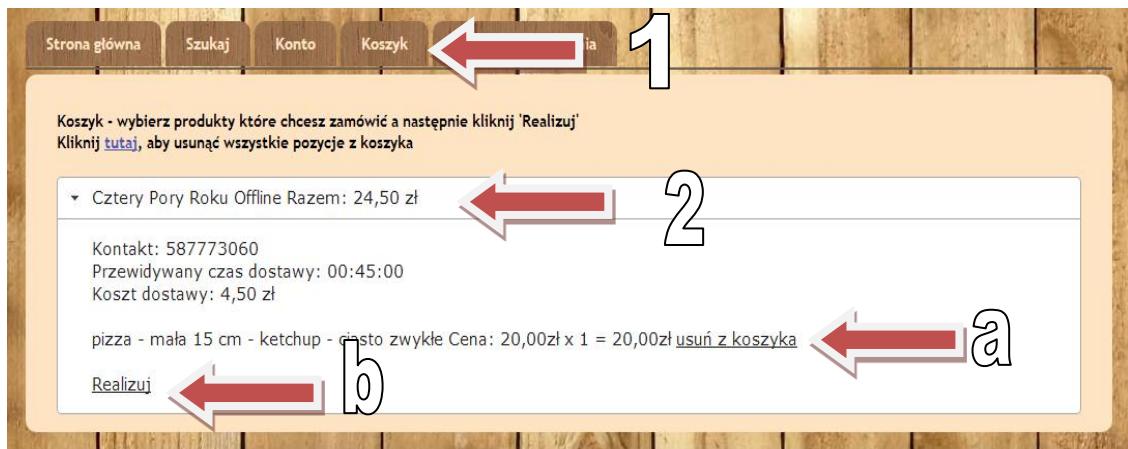
- **Wybieramy opcję „Historia zamówień” (1)**
- **Z listy możemy wybrać nasze zamówienie (a). Wyświetla się jego szczegóły. Z tego miejsca możemy również sprawdzić status zamówienia.**
- **W polu (b) możemy wybrać datę interesujących nas zamówień oraz zatwierdzić przyciskiem filtruj.**

1.2.4. Komentarze

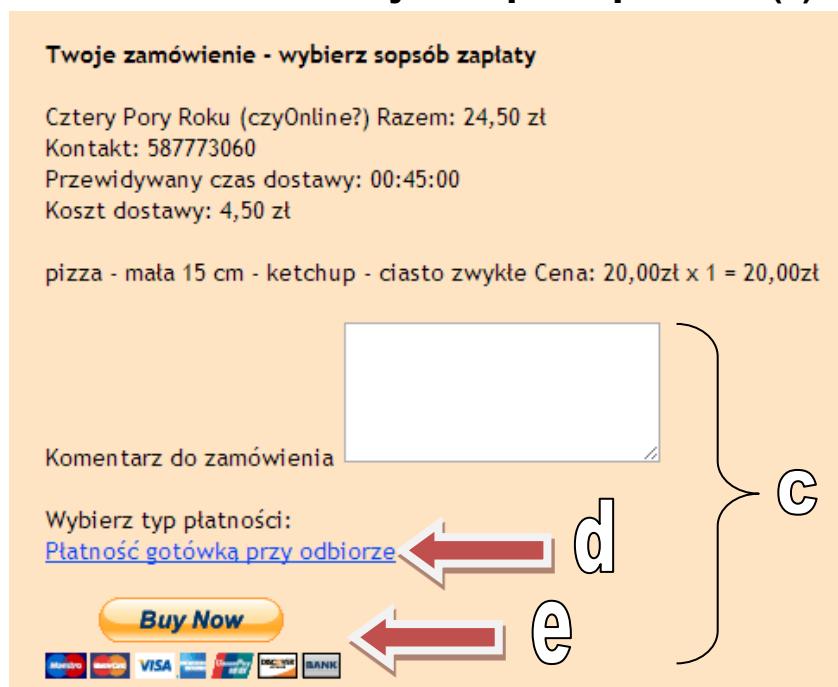
Dane użytkownika Historia zamówień Komentarze 1
Numer komentatora: 6. Wstawiony dla: Camargo (Tczew) 09/12/2012 17:41:18
Ocena: 2
Treść komentarza:
Jedzenie bardzo dobre, jednak lokal stanowczo za mały.
[Usuń](#) 2
Numer komentatora: 7. Wstawiony dla: Restauracja Gdańsk (Gdańsk) 09/12/2012 17:40:36
Ocena: 5
Treść komentarza:
Wspaniała Restauracja - serdecznie polecam.
[Usuń](#)

Wybieramy opcje „Komentarze” (1). Poniżej wyświetlają się nasze komentarze. Możemy usunąć wybrane komentarze wciskając przycisk „Usuń” (2).

1.2.5. Koszyk

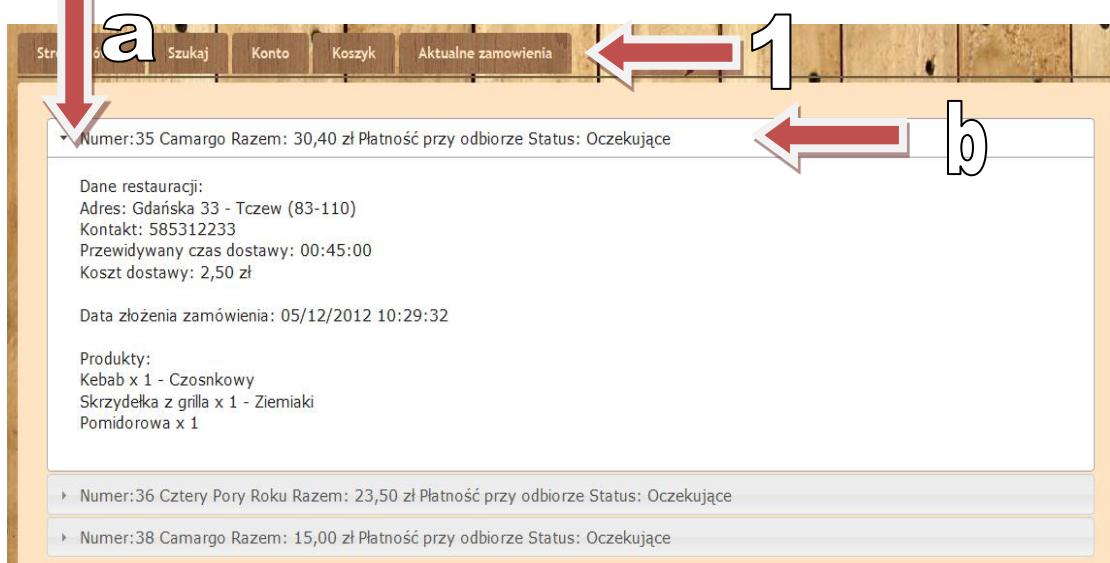


- 1. Aby przejść do koszyka wybieramy opcję „Koszyk” (1)**
- 2. W tym miejscu widoczne są nasze zamówienia przyporządkowane do odpowiednich restauracji (2).**
- 3. Możemy usunąć zamówienia klikając „tutaj” (a) bądź wysłać zamówienie do realizacji klikając (b) – następnie możemy zostawić komentarz i wybrać sposób płatności (c).**



- 1. Po wyborze płatności „Płatność gotówką przy odbiorze” (d) zostanie wyświetcone powiadomienie o zakończeniu transakcji (PaySuccess).**
- 2. Po wyborze płatności „PayPal” zostaniemy przekierowani na stronę płatności PayPal.**

1.2.6. Aktualne Zamówienia



- 1. Do naszych zamówień przechodzimy przyciskiem „Aktualne zamówienia” (1).**
- 2. Aby zobaczyć jego szczegóły możemy wybrać nasze zamówienie: numer (a) oraz status (b).**

2. Strona Restauracji

Gdy znajdziemy interesującą nas restaurację (wykorzystując stronę główną, bądź opcję szukaj) możemy przejść do strony restauracji.

2.1. Strona główna restauracji

Tutaj możemy zobaczyć podstawowe informacje na temat restauracji takie jak krótka informacja, promocje, czy też godziny otwarcia.

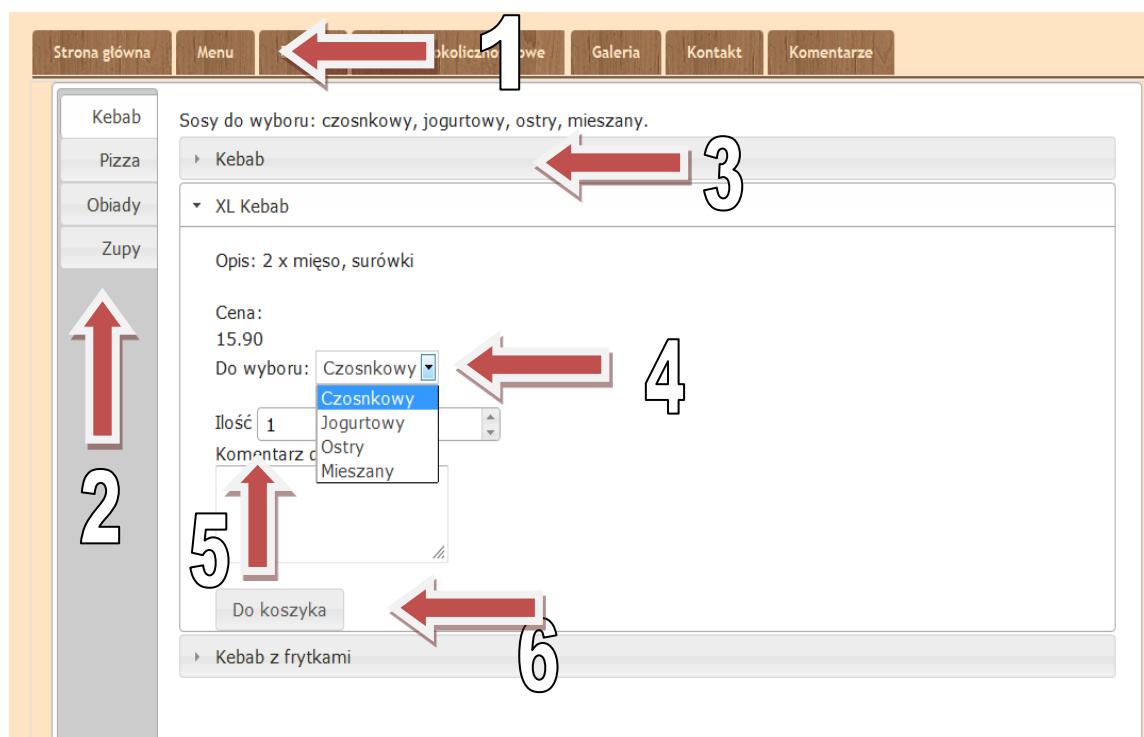
The screenshot shows a website for a restaurant named CAMARGO. At the top, there is a navigation bar with links: Strona główna, Menu, Dowóz, Imprezy okolicznościowe, Galeria, Kontakt, and Komentarze. Below the navigation bar, there are two main content boxes. The left box contains text about the restaurant's philosophy and facilities. The right box features a logo and sections for promotions and opening hours.

CAMARGO jest Tczewskim Barem Restauracyjnym. Staramy się, aby każde danie było smaczne i wyjątkowe. Używamy wyłącznie świeżych produktów. Stale poszerzamy nasze menu, aby jak najlepiej zadowolić gusta naszych klientów. CAMARGO posiada małą salę barową, średnią salę do 40 osób oraz dużą salę do 100 osób, dzięki czemu mogą Państwo u nas zarezerwować termin na wyjątkową okoliczność, taką jak np. chrzciny czy wesele.

PROMOCJE
Przy zamówieniu powyżej 100 zł - **dowóz** na terenie miasta **gratis**.
Przy zamówieniu powyżej 55 zł - **1 litr soku gratis**.
Przy zamówieniu powyżej 110 zł - **2 litr soku gratis**.

GODZINY OTWARCIA
Pon - Czw: od 10:00 do 22:00
Pt - Sob: od 10:00 do 23:00
Niedziela: od 11:00 do 22:00

2.2. Menu



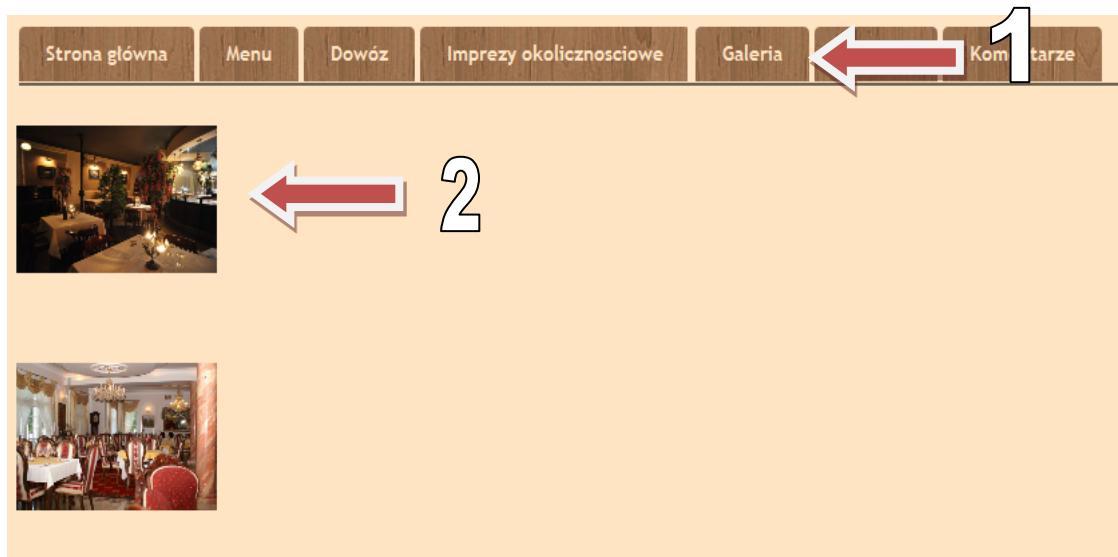
1. Wybieramy ze strony restauracji opcję „Menu” (1).
2. Z skategoryzowanego menu możemy wybrać rodzaj dania (2).
3. Następnie z listy rozwijanej (3) danie.
4. Należy również wybrać dodatkowe opcje takie jak „sos”, „dodatki” itp. (4), oraz ilość zamawianego dania (5).
5. Aby przenieść zamówienie do koszyka wybieramy opcję (6).

2.3 Dowóz, Imprezy Okolicznościowe, Kontakt

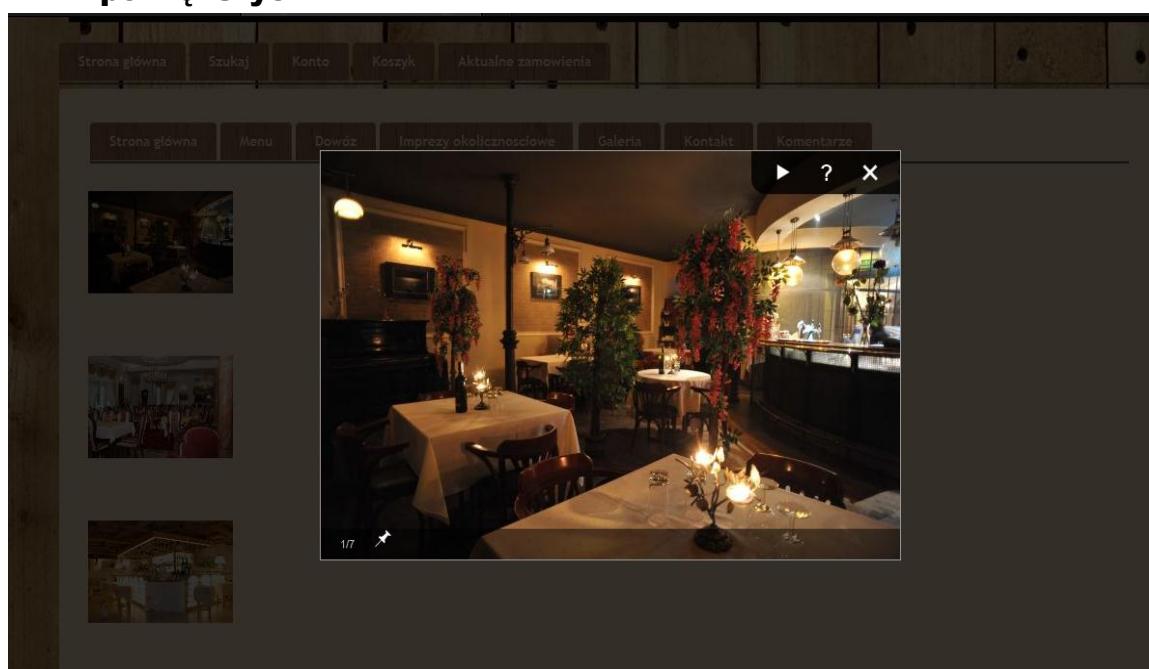


Aby wyświetlić informacje o dowozie wybieramy opcję Dowóz na pasku panelu restauracji. Podobnie postępujemy w przypadku informacji na temat imprez okolicznościowych oraz kontaktu (1).

2.4. Galeria



- 1. Wybieramy opcję „Galeria” (1), aby przejść do listy zdjęć.**
- 2. Wybieramy interesujące nas zdjęcie i klikamy na nie (2), aby powiększyć:**



2.5. Komentarze

The screenshot shows a web page for adding a comment. At the top, there is a navigation bar with links: Strona główna, Menu, Dowóz, Imprezy okolicznościowe, 1 (highlighted with a red arrow), Kontakt, and Komentarze. Below the navigation bar, the main content area has a light orange background. It contains a form titled "Dodaj komentarz". The form fields are: "Ocena:" with a 5-star rating scale (2, highlighted with a red arrow), "Treść komentarza:" (3, highlighted with a red arrow), and a "Wyślij" button (4, highlighted with a red arrow). Below the form, the submitted comment is displayed: "Numer komentarza: 3. Wstawiony przez: test 05/12/2012 11:29:04", "Ocena: 3,5", and "Treść komentarza:". A large curly brace (5) groups the displayed comment information.

- 1. Aby wystawić komentarz restauracji należy wybrać opcję „Komentarze” (1).**
- 2. Następnie wybrać ocenę z 5-cio gwiazdkowej skali (2), wpisać treść komentarza (3) oraz zaakceptować wszystko wybierając opcję wyślij (4).**
- 3. Istniejące już komentarze i ogólna ocena restauracji wyświetlona jest poniżej formularza wystawiania. (5)**



Dla Menadżerów

Szanowny Kliencie

Opracowanie to zostało stworzone, aby ułatwić państwu korzystanie z usług systemu E-Restauracja.

Prosimy o korzystanie z oprogramowania w sposób opisany w poniższej instrukcji obsługi. Wszelkie odstępstwa od niej mogą prowadzić do usterek.

Oprogramowanie zostało dokładnie sprawdzone oraz przetestowane pod względem bezpieczeństwa oraz funkcjonalności.

Przed rozpoczęciem korzystania z serwisu należy uważnie przeczytać dostarczoną do niego dokumentację i zachować ją w celu użycia w przyszłości.

Wstęp

Portal E-Restauracja to doskonałe narzędzie do zakładania i zarządzania restauracjami. W bardzo wygodny sposób można edytować menu, stworzyć galerię, zarządzać pracownikami oraz wszelkimi danymi pracowników i restauracji. Zostało one opracowane i zaprojektowane w celu usprawnienia oraz ułatwienia działania państwa restauracji.

Mamy nadzieję, że będą Państwo zadowoleni z korzystania z naszego oprogramowania.

1. Zakładanie konta Menagera



1. Wybieramy opcję „Dodaj swoją restaurację” (1)

2. Zostaniemy przekierowani do formularza rejestracji menadżera:

Uwaga!
Aby dodać nową restaurację należy stworzyć konto menadżera, które umożliwia nie tylko dodawanie nowych lokali ale również zarządzanie nimi oraz personelem. Wykorzystując w tym celu istniejące konto, stracisz możliwość wykonywania zamówień.

Utwórz nowe konto menadżera wypełniając poniższy formularz lub wykorzystaj [istniejące konto](#)

Dane rejestracji
Wprowadź swoje dane, a następnie kliknij 'Załóż konto'.

Login	<input type="text"/>
Email	<input type="text"/>
Powtórz email	<input type="text"/>
Hasło (Minimum 6 znaków.)	<input type="password"/>
Powtórz hasło	<input type="password"/>
Pytanie do przywracania hasła	<input type="text"/>
Odpowiedz	<input type="text"/>
Imię	<input type="text"/>
Nazwisko	<input type="text"/>
Adres	<input type="text"/>
Miasto	<input type="text"/>
Kod pocztowy	<input type="text"/>
Kraj	Afganistan
Data urodzenia	<input type="text"/>
Płeć	Mężczyzna
Numer telefonu	<input type="text"/>
Załóż konto	

A red arrow points to the 'Załóż konto' button, which is highlighted with a red box. A large number '1' is positioned above the input fields, and a large number '2' is positioned below the 'Załóż konto' button.

3. W tym miejscu wypełniamy dane (1) i zatwierdzamy przyciskiem „Załóż konto” (2).

Uwaga!

Aby dodać nową restaurację należy stworzyć konto menadżera, które umożliwia nie tylko dodawanie nowych lokali, ale również zarządzanie nimi oraz personelem.

Wykorzystując w tym celu istniejące konto, stracisz możliwość wykonywania zamówień.

2. Logowanie do systemu

The screenshot shows the E.Restauracja website's login form. At the top, there is a navigation bar with a pizza icon, the text "E.Restauracja", and several menu items: "Strona główna", "Szukaj", "Konto", "Koszyk", and "Aktualne zamówienia". A large red arrow labeled "1" points to the "Zaloguj" button on the right side of the header. Below the header, the main content area has a light orange background. It contains instructions: "Zaloguj się aby w pełni wykorzystać możliwości serwisu." and "Dane logowania: Wprowadź swój login oraz hasło." There are two input fields: "Login" and "Hasło", each with a red double-headed arrow labeled "2" and "3" respectively. Below these fields is a checkbox labeled "Zapamiętaj mnie." and a "Zaloguj" button, which is also marked with a red double-headed arrow labeled "4". At the bottom of the form, there is a link "Zapomniałeś hasła? [Tutaj](#) możesz odzyskać dostęp do konta." and another link "Jeśli nie posiadasz jeszcze konta możesz je założyć [tutaj](#)".

Aby zalogować się do systemu należy:

- 1. Wybrać opcję (1) „Zaloguj” w prawym górnym rogu strony lub kliknąć w zakładkę „Konto” (a),**
 - 2. Wpisać przyznany wcześniej Login (2),**
 - 3. Wpisać hasło (3),**
- *Istnieje możliwość zapamiętania ustawień logowania.**
- 4. Wcisnąć przycisk zaloguj (4).**

Aby wylogować menadżera wybieramy opcję wyloguj (1).



3. Panel Menadżera

The screenshot shows the homepage of the E-Restauracja Manager Panel. At the top right, it says "Witaj test2!" and "Wyloguj". Below the header is a navigation bar with four tabs: "Strona główna", "Twoje restauracje", "Pracownicy", and "Konto". The main content area has a yellow background and contains the following text:

Witamy w panelu menadżera.
Zakładka "Twoje restauracje" pozwala na dodawanie nowych restauracji oraz zarządzanie nimi.
Po dodaniu restauracji masz możliwość zarządzania zawartością strony restauracji, do której dostęp mają klienci.
Intuicyjne w użyciu edytory tekstu pozwolą Ci sformatować treść strony, a możliwość dodawania zdjęć do galerii pozwoli na prezentowanie wystoju Twojego lokalu.
Możliwość dodawania kategorii oraz produktów pozwala odwzorować niemal każde menu.
Sekcja "Komentarze" zapewnia szybki podgląd komentarzy i ocen jakie wystawili klienci twojej restauracji.
Jeśli treść komentarza jest obraźliwa lub w inny sposób nie odpowiadająca wybierz opcję "Zgłoś nadużycie" a sprawą zajmą się administratorzy strony.
W zakładce "Pracownicy" możesz dodawać nowych pracowników jednocześnie dając im dostęp do Panelu obsługi restauracji.

At the bottom of the page, there is a footer with links to "Pomoc", "Informacje", and "Zgłaszenie błędów". It also includes the copyright notice "© 2012 Erestauracja".

3.1. Twoje Restauracje

The screenshot shows the "Twoje Restauracje" section of the manager panel. A large red arrow labeled "1" points to the "Twoje restauracje" tab in the top navigation bar. Another red arrow labeled "4" points to the text "Wybierz restaurację, którą chcesz zarządzać lub kliknij tutaj, aby dodać nową.". A third red arrow labeled "2" points to the section title "▼ Restauracja Cztery Pory Roku". A fourth red arrow labeled "3" points to the "Edytuj dane Zmiana hasła Zarządzaj" link at the bottom of the page. The page displays the details of the restaurant "Cztery Pory Roku", including its login information, address, phone number, and delivery times.

- 1. Wybieramy opcję „Twoje Restauracje” (1), otworzy się lista restauracji.**
- 2. Z listy wybieramy nazwę restauracji (2), aby zobaczyć jej szczegóły.**
- 3. Mamy do wyboru następujące opcje (3):**
 - a. Edytuj dane,**
 - b. Zmiana hasła,**
 - c. Zarządzaj.**
- 4. Możemy również wybrać opcję „tutaj” (4) i wypełnić formularz (a), aby dodać nową restaurację.**

Wypełnij poniższy formularz oraz kliknij 'Zapisz', aby dodać nową restaurację.

Dane rejestracji

Login	
Adres email	
Powtórz email	
Hasło (Minimum 6 znaków.)	
Powtórz hasło	
Pytanie do przywracania hasła	
Odpowiedz	
Nazwa firmy	
Nazwa wyświetlana	
Adres lokalu	
Miasto	
Kod pocztowy	
Kraj	Afganistan
Numer telefonu	
NIP	
REGON	
Czas dostawy	
Cena dostawy	

Zapisz

3.1.1. Edycja danych

Dane restauracji
Wprowadź nowe dane, a następnie kliknij 'Zapisz', aby edytować dane restauracji.

Nazwa firmy	Restauracja Cztery Pory Roku
Nazwa wyświetlana	Cztery Pory Roku
Adres lokalu	ul. Jagiellońska 55
Miasto	Tczew
Kod pocztowy	83-110
Kraj	Polska
Numer telefonu	587773060
NIP	123-456-32-18
REGON	123456785
Czas dostawy	00:45:00
Cena dostawy	3,50
Widoczna dla klientów	<input checked="" type="checkbox"/>

Zatwierdź zmiany  **2**

1

Na tej stronie można edytować pojedyncze dane restauracji (1), i zatwierdzić je przyciskiem „Zatwierdź zmiany” (2).

3.1.2. Zmiana hasła

Należy postępować zgodnie z instrukcją (1).

Zmiana hasła
Wprowadź dotychczasowe i nowe hasło oraz kliknij 'Zatwierdź' w celu zmiany.

Dotychczasowe hasło

Nowe hasło (Minimum 6 znaków.)

Powtórz hasło

Zatwierdź

1

3.1.3. Zarządzanie

Strona główna Menu Dowóz Imprezy okolicznościowe Galeria Kontakt

Strona główna - kliknij [tutaj](#) aby edytować

1

2

Nasza restauracja znajduje się przygotowanie posiłków kuchni greckiej oraz włoskiej. Dania można zamawiać telefonicznie jak i zjeść na miejscu w bardzo klimatycznym lokalu. Nasza jedyna restauracja w mieście oferujemy piec z pieca opalanego drewnem, co zapewnia najwyższą jakość oraz jej niepowtarzalny smak.

Godziny otwarcia

Pn.-Czw. 12.00-22.00
Pt. - Sb. 12.00-23.00
Nd. 12.00-21.00

W sezonie letnim
Nd. 12.00-22.00



Promocje

Przy każdym zamówieniu powyżej 50 zł 1 litr soku i dowóz gratis

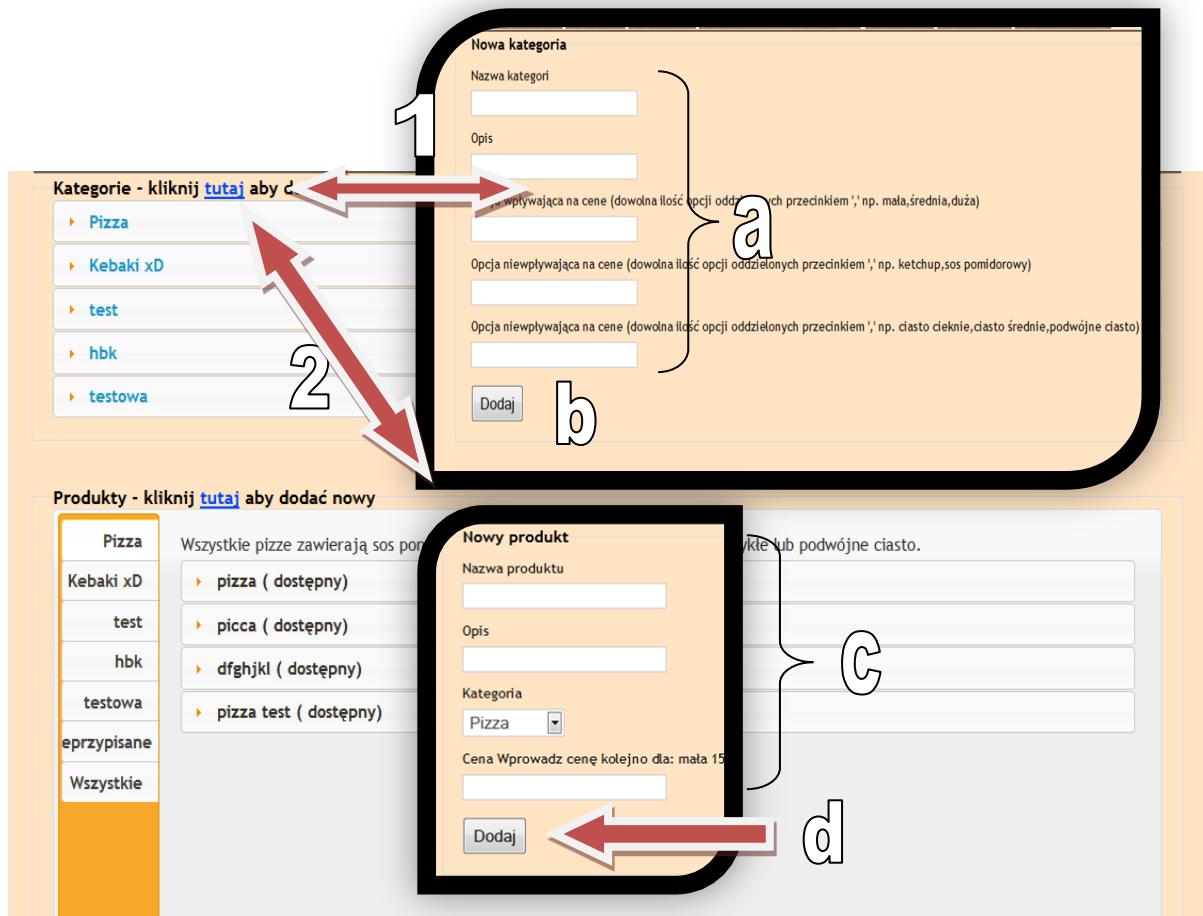
- 1. Po wybraniu opcji zarządzaj możemy wybrać stronę (1),**
- 2. Następnie klikamy przycisk „tutaj” (2), aby edytować poszczególne strony.**

3.1.3.1. Strona główna



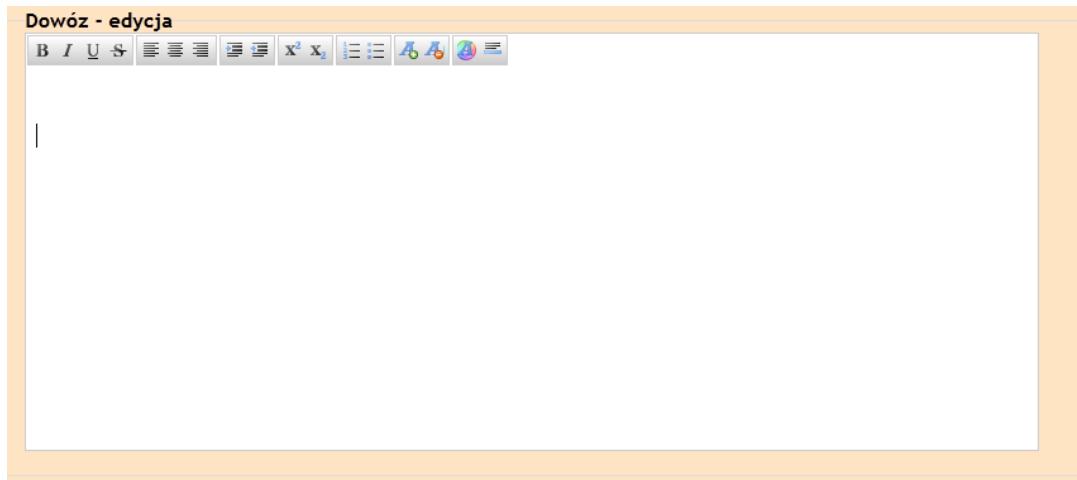
- Możemy edytować tekst z informacjami i promocjami (1) (do dyspozycji mamy panel tekstowy),**
- Możemy dodać zdjęcie główne, bądź je usunąć (2).**

3.1.3.2. Menu



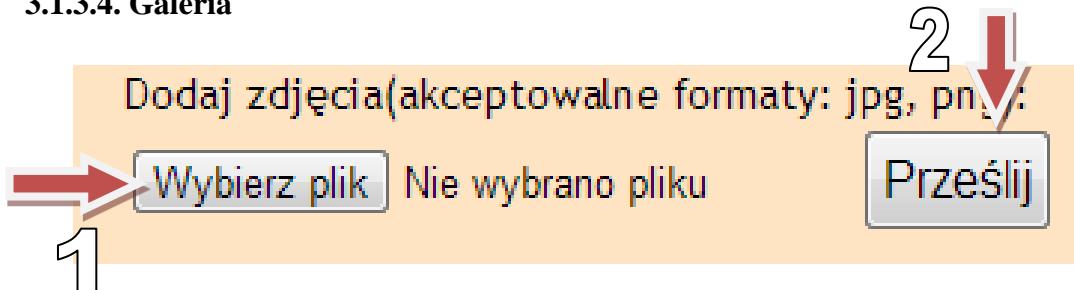
- Aby dodać kategorie wybieramy „tutaj” (1), a następnie wypełniamy pola (a) oraz zatwierdzamy przyciskiem „Dodaj” (b).**
- Aby dodać produkt wybieramy „tutaj” (2), a następnie wypełniamy pola (c) oraz zatwierdzamy przyciskiem „Dodaj” (d).**

3.1.3.3. Dowóz, Imprezy Okolicznościowe, Kontakt



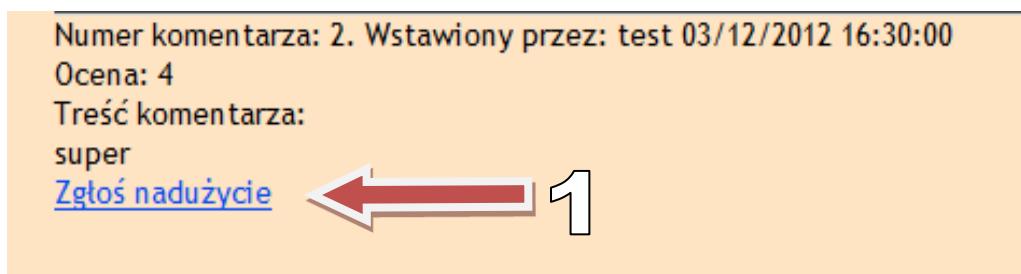
- **Po wybraniu opcji edytuj dostajemy dostęp do edycji tekstu o dowozie, imprezach okolicznościowych oraz kontaktcie.**

3.1.3.4. Galeria

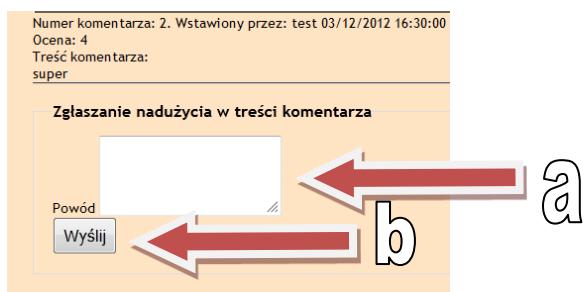


- **Aby dodać zdjęcia wybieramy opcję „Wybierz plik” (1) wybieramy lokalnie zdjęcie, które chcemy dodać i zatwierdzamy przyciskiem „Prześlij” (2).**

3.1.3.5. Komentarze

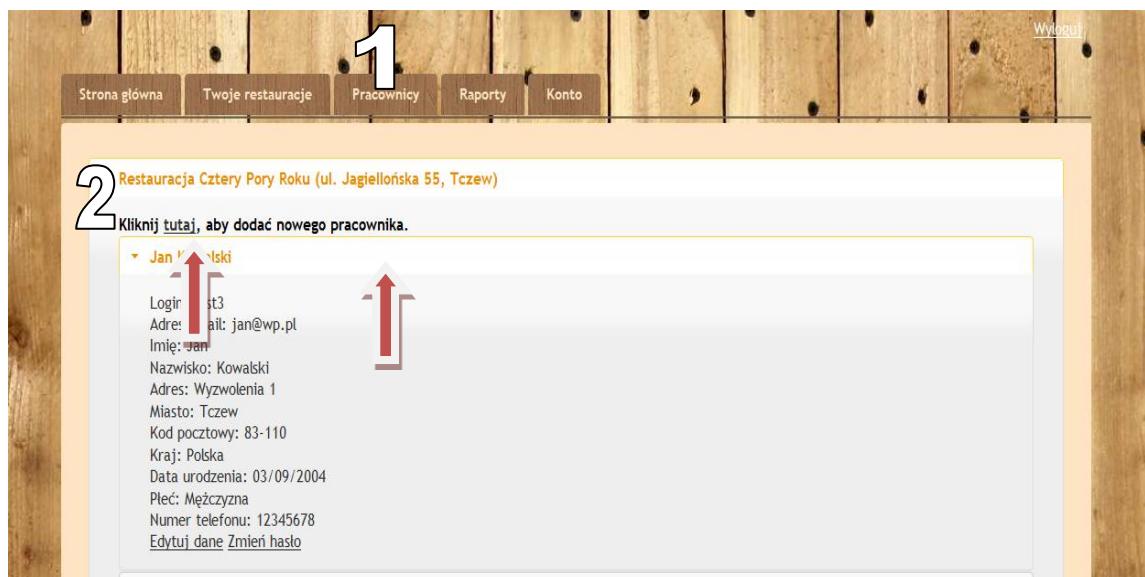


- **W tym miejscu możemy zgłosić do administratora nadużycie (1) zawarte w komentarzu. W tym celu klikamy na „Zgłoś nadużycie”.**



- wpisujemy treść komentarza dotyczącego nadużycia (a)
- wysyłamy komentarz przyciskiem „Wyślij” (b).

3.2. Pracownicy



- 1. Przechodzimy do panelu pracowników wybierając opcję (1),**
- 2. Z listy wybieramy restaurację (2),**
- 3. Po wybraniu restauracji wyświetlona zostanie lista jej pracowników. Z listy można wybrać danego pracownika lub dodać nowego.**

3.2.1. Edycja danych

W tym oknie można edytować dane, a następnie zatwierdzić je kliknięciem przycisku zapisz (1).

Edycja danych
Wprowadź nowe dane, a następnie kliknij 'Zapisz'.

Email jan@wp.pl

Imię Jan

Nazwisko Kowalski

Adres Wyzwolenia 1

Miasto Tczew

Kod pocztowy 83-110

Kraj Polska

Data urodzenia 03/09/2004

Płeć Mężczyzna

Numer telefonu 12345678

Zapisz  1

3.2.2. Zmiana hasła

Należy wpisać hasło, wpisać je ponownie i zatwierdzić przyciskiem (1).

Zmiana hasła pracownika
Wprowadź hasło, a następnie kliknij "Zapisz hasło".

Hasło (Minimum 6 znaków.)

Powtórz hasło

Zapisz hasło  1

3.3. Konto



1. Wybieramy przycisk „Konto” (1) i dostajemy dostęp do strony „Dane użytkownika” z opcjami:

a. Edytuj dane

b. Zmień hasło

W pkt. a. i b. można edytować dane oraz zmienić hasło jak w przypadku pracowników (3.3.2)

3.4. Zgłaszanie błędów

Wybierając opcję zgłoszenia błędów (1) zostaniemy przekierowani do formularza zgłoszenia błędów.

Podaj swój adres email oraz treść zgłoszenia, a następnie kliknij "Wyślij", aby powiadomić nas o znalezionym błędzie.
Formularz zgłoszenia

Email 2

Treść zgłoszenia 3

Wyślij 4

 [Pomoc](#) • [Informacje](#) • [Zgłoś błąd](#) 1

© 2012 Erestauracja

- **Wpisujemy swój adres poczty elektronicznej (2)**
- **Wpisujemy treść zgłoszenia (3), tj. szczegóły błędu**
- **Zatwierdzamy przyciskiem wyślij (4)**

E.Restauracja

Instrukcja Obsługi



Dla Restauracji

Szanowny Kliencie

Opracowanie to zostało stworzone, aby ułatwić państwu korzystanie z usług systemu E-Restauracja.

Prosimy o korzystanie z oprogramowania w sposób opisany w poniższej instrukcji obsługi. Wszelkie odstępstwa od niej mogą prowadzić do usterek.

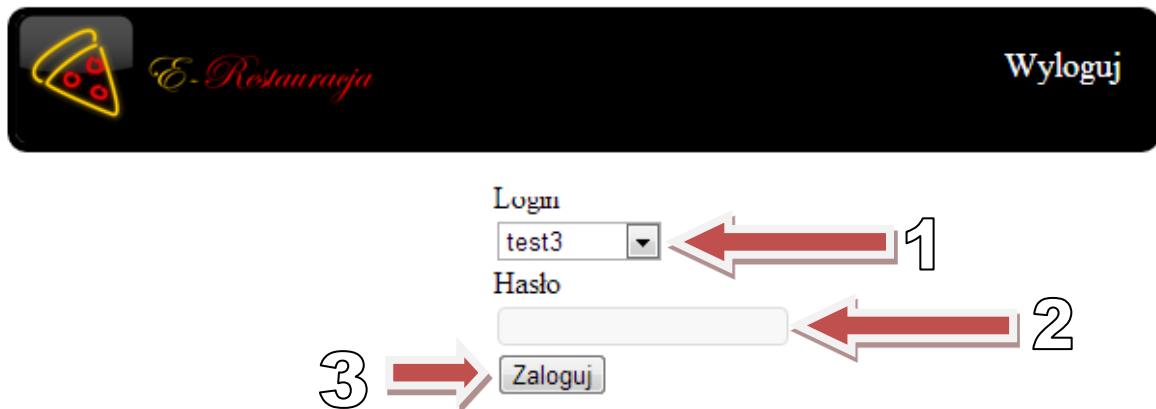
Oprogramowanie zostało dokładnie sprawdzone oraz przetestowane pod względem bezpieczeństwa oraz funkcjonalności.

Przed rozpoczęciem korzystania z serwisu należy uważnie przeczytać dostarczoną do niego dokumentację i zachować ją w celu użycia w przyszłości.

Wstęp

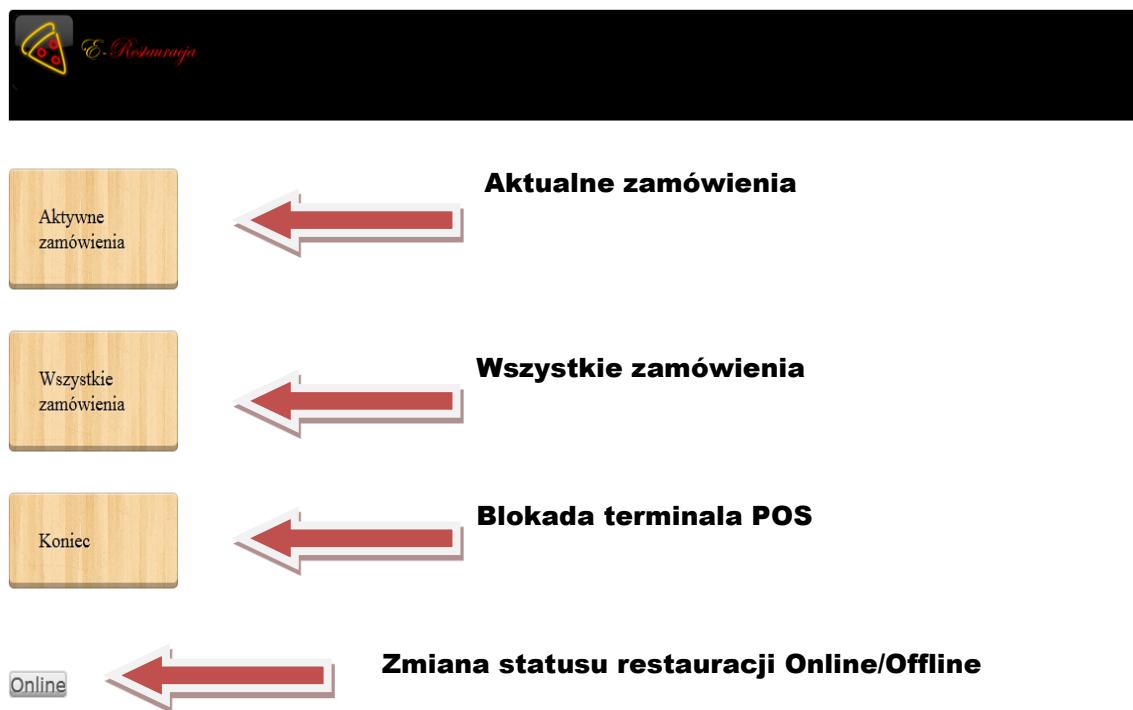
Portal E-Restauracja to doskonałe narzędzie do zarządzania zamówieniami składanymi przez klientów restauracji. W bardzo wygodny sposób można przyjmować zamówienia, zmieniać ich status oraz przeglądać historię zamówień. Zostało one opracowane i zaprojektowane w celu usprawnienia oraz ułatwienia działania państwa restauracji oraz pracy personelu restauracji. Mamy nadzieję, że będą Państwo zadowoleni z korzystania z naszego oprogramowania.

1. Restauracja

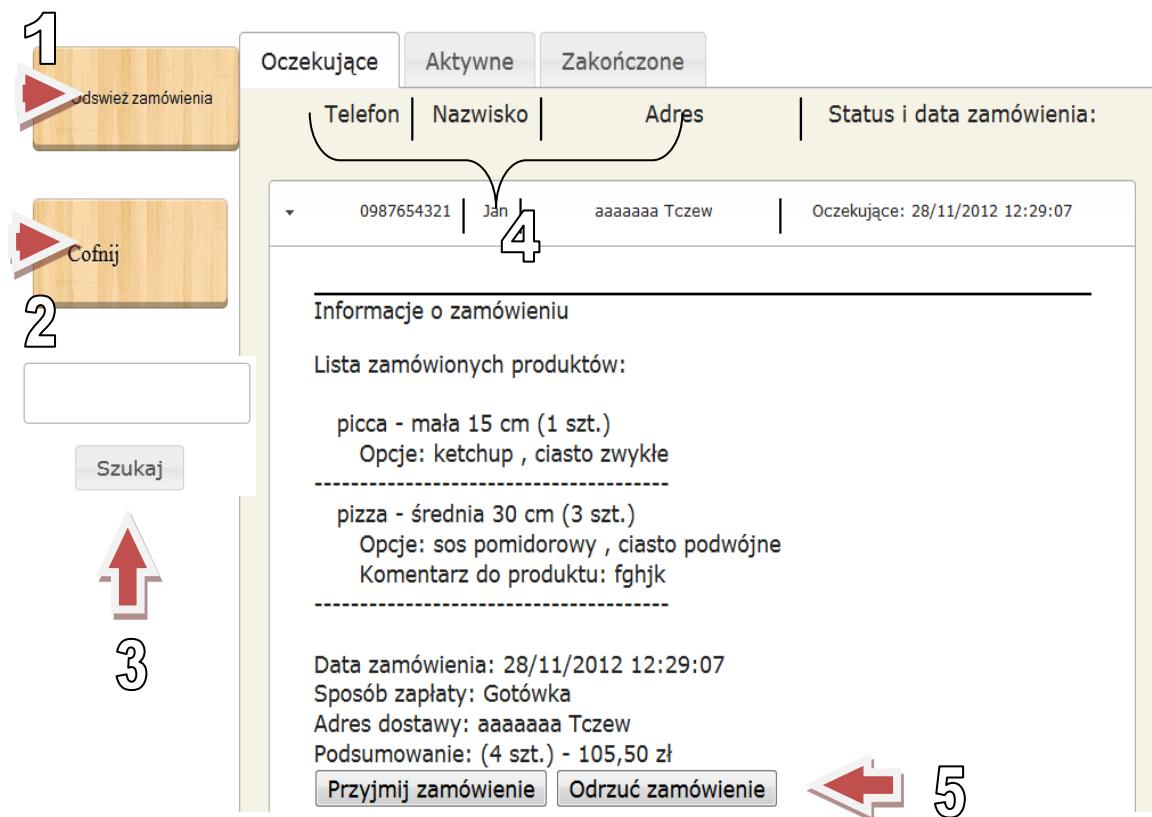


1.1. Panel Restauracji

Aby zalogować się należy z rozwijanej listy (1) wybrać swój login, a następnie wpisać hasło użytkownika (2) i wcisnąć przycisk zaloguj (3).
Z tego miejsca mamy dostęp do dotykowego panelu restauracji.



1.2. Aktywne zamówienia



- 1 – Możliwość odświeżenia listy zamówień.**
- 2 – Wycofanie do głównego panelu POS.**
- 3 – Wyszukiwanie konkretnego zamówienia, poprzez wpisanie telefonu, nazwiska lub adresu i zatwierdzenie przyciskiem „szukaj”.**
- 4 – Możliwość wyboru zamówień oczekujących, aktywnych lub zakończonych.**
- 5 – Zmiana statusu zamówienia poprzez wybranie opcji „Przyjmij zamówienie” bądź „Odrzuć zamówienie”**

1.3. Wszystkie zamówienia

The screenshot shows the E-Restauracja POS software interface. At the top, there's a logo of a pizza slice and the text "E-Restauracja". On the right, there's a "Wyloguj" (Logout) button. Below the header, there are search fields for "Telefon" (Phone), "Nazwisko" (Last Name), "Adres" (Address), and "Status i data zamówienia" (Order Status and Date). To the left, there's a "Cofnij" (Cancel) button and a large red arrow pointing upwards labeled "2". Underneath the search fields, there are date filters: "Od: 02/12/2012" and "Do: 26/12/2012", followed by a "Filtruj" (Filter) button. The main area displays order details for two items:

Produkt	Opis	Opis Opcji
picca - mała 15 cm	(1 szt.)	Opcje: ketchup , ciasto zwykłe
pizza - mała 15 cm	(1 szt.)	Opcje: ketchup , ciasto zwykłe

Below the products, the order summary is shown:

Data zamówienia: 03/12/2012 14:26:57
Sposób zapłaty: Gotówka
Komentarz do zamówienia: asdfg
Adres dostawy: Główna 3 Tczew
Podsumowanie: (2 szt.) - 35,50 zł

At the bottom, there are fields for "123456 | imie | adres 1 Tczew" and "Zakończone: 05/12/2012 10:34:07". A curly brace on the right side of the order details section is labeled "3".

- 1 – Wycofanie do głównego panelu POS.**
- 2 – Możliwość filtrowania (poprzez wybranie daty oraz zatwierdzenie przyciskiem „filtruj”).**
- 3 – Podgląd szczegółów zamówienia poprzez wybranie konkretnego zamówienia.**



Instrukcja Obsługi



Dla Administratorów

Szanowny Kliencie

Opracowanie to zostało stworzone, aby ułatwić państwu korzystanie z usług systemu E-Restauracja.

Prosimy o korzystanie z oprogramowania w sposób opisany w poniższej instrukcji obsługi. Wszelkie odstępstwa od niej mogą prowadzić do usterek.

Oprogramowanie zostało dokładnie sprawdzone oraz przetestowane pod względem bezpieczeństwa oraz funkcjonalności.

Przed rozpoczęciem korzystania z serwisu należy uważnie przeczytać dostarczoną do niego dokumentację i zachować ją w celu użycia w przyszłości.

Wstęp

Panel administracyjny portalu E-Restauracja umożliwia zarządzanie wszystkimi danymi w systemie. Przy użyciu phpMyAdmina wygodnie można edytować dowolne pola w bazie danych. Zostały również dodatkowo zaimplementowane podstawowe funkcje tworzenia oraz usuwania użytkowników i ról. Został on opracowany i zaprojektowany w celu wsparcia pracy administratora witryny.

Mamy nadzieję, że będą Państwo zadowoleni z korzystania z naszego oprogramowania.

1. Administrator

1.1. Tworzenie nowego użytkownika

The screenshot shows a user creation form. On the left, there's a sidebar with 'Users' and 'Create user' selected. The main area has a title 'Nowy użytkownik' (1). It contains fields for 'Login', 'Email', 'Powtórz email', 'Hasło (Minimum 6 znaków.)', 'Powtórz hasło', 'Pytanie do przywracania hasła', 'Odpowiedz', 'Imię', 'Nazwisko', 'Adres', 'Miasto', 'Kod pocztowy', 'Kraj' (with 'Afganistan' selected), 'Data urodzenia', 'Płeć' (with 'Mężczyzna' selected), and 'Numer telefonu'. A large red arrow labeled '1' points to the 'Create user' link in the sidebar. A curly brace labeled '2' groups all the input fields. A red arrow labeled '3' points to the 'Załóż konto' button at the bottom.

Aby stworzyć nowego użytkownika:

1. Wybieramy opcję „Create user” (1),
2. Należy wypełnić wszystkie pola (2),
3. Wybieramy opcję „Załóż konto” (3),

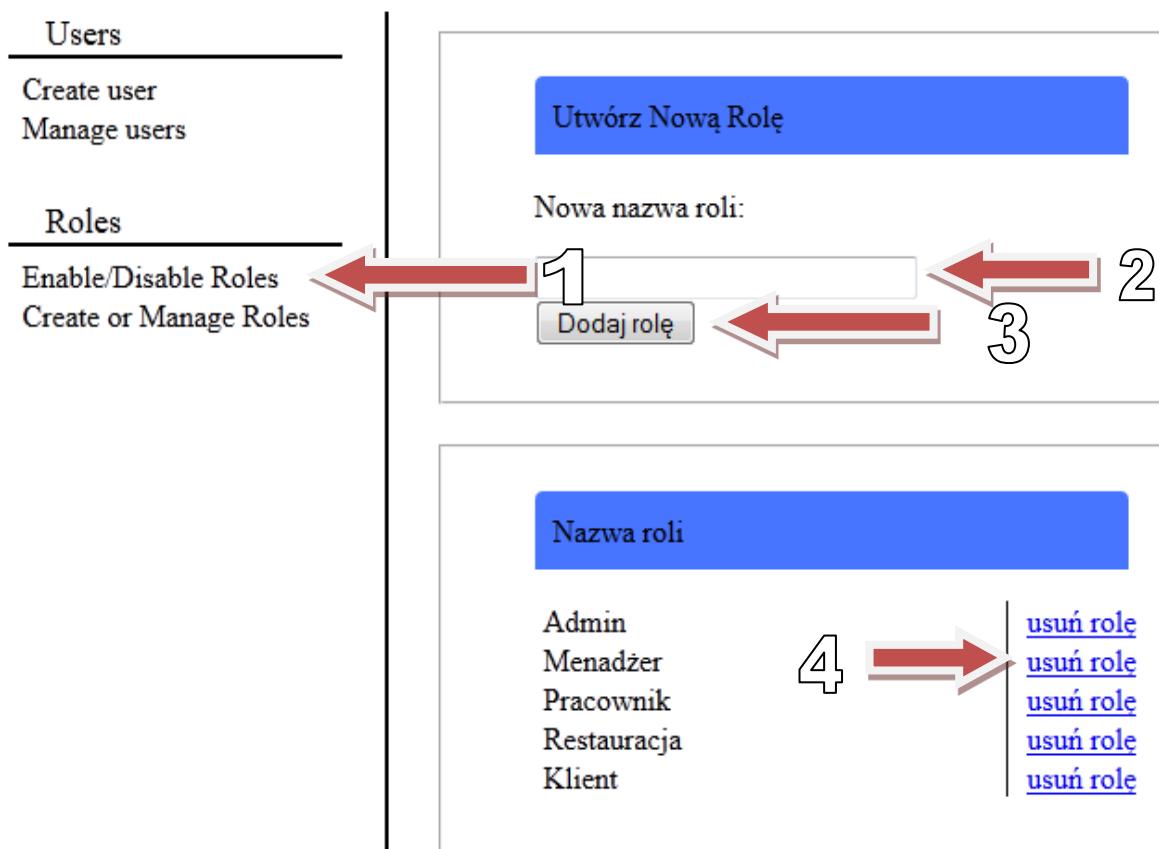
1.2. Zarządzanie użytkownikami

The screenshot shows a user management interface. On the left, there's a sidebar with 'Users' and 'Manage users' selected. The main area lists user names: '4pory', 'admin', 'barcamargo', and 'camargo'. To the right of each name is a blue 'usuń' (Delete) link. A red arrow labeled '1' points to the 'Manage users' link in the sidebar. A red arrow labeled '2' points to one of the 'usuń' links.

Nazwa użytkownika	
4pory	usuń
admin	usuń
barcamargo	usuń
camargo	usuń

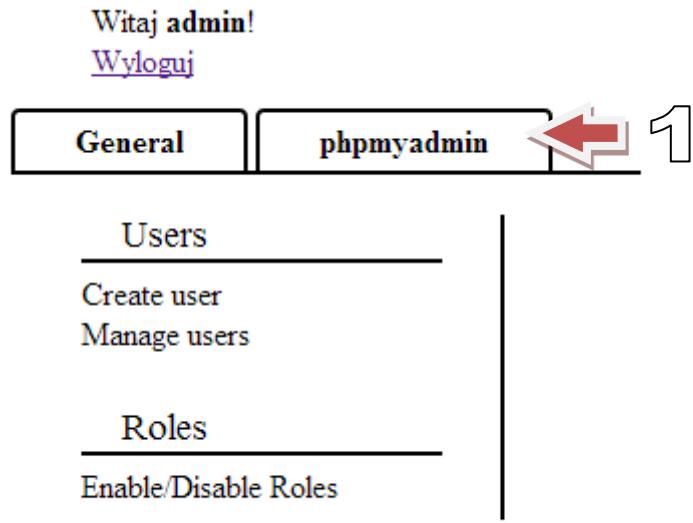
1. Wybieramy opcję „Manage users” (1),
2. Aby usunąć użytkownika wybieramy opcję „usuń” przy jego nazwie (2).

1.3. Dodawanie/usuwanie ról użytkowników



- 1. Wybieramy opcję „Enable/Disable Roles” (1), aby przejść do panelu ról.**
- 2. Aby utworzyć nową rolę wpisujemy jej nazwę w polu (2). Po wpisaniu wybieramy opcję „Dodaj rolę” (3).**
- 3. Aby usunąć rolę wybieramy opcję „usuń rolę” przy nazwie tej roli.**

1.4. phpMyAdmin



Panel administratora pozwala również na dostęp do bazy danych za pomocą narzędzia phpMyAdmin, do którego można wejść poprzez kliknięcie w link (1).