



POLITECHNIKA WROCŁAWSKA

ZASTOSOWANIE INFORMATYKI W GOSPODARCE

Aplikacja do rezerwacji miejsc w restauracjach

*Hubert Dus
Jędrzej Kozal
Eliza Mocek
Piotr Montewka*

prowadzący
Dr inż. Marek WODA

9 czerwca 2018

1 Wstęp

1.1 Cel projektu

Celem projektu realizowanego w ramach kursu, jest stworzenie aplikacji biznesowej, umożliwiającej rezerwacje miejsc w wybranych restauracjach. Zakłada się, że tworzona aplikacja będzie umożliwiała rezerwację miejsc w restauracji w porozumieniu z właścicielem i obsługą. Analogicznym pomysłem może być rezerwacja miejsc w kinie, która najczęściej odbywa się drogą elektroniczną. Tworzona aplikacja ma ułatwić pracę restauratorom i pozwolić na lepsze zarządzanie dostępnym miejscem oraz pośrednio zaopatrzeniem i personelem.

Ponadto istotnym celem projektu, jest zapoznanie się z realiami pracy nad dużym projektem informatycznym oraz analiza i próba rozwiązania podstawowych problemów jakie są związane z tym zagadnieniem. Projekt może być wymagający na poziomie technicznym, jak i komunikacyjnym. Przed przystąpieniem do aktywności zawodowej nie jest łatwo zdobyć doświadczenie w zakresie pracy w większym zespole inżynierskim.

1.2 Zakres projektu

Podstawowy zakres funkcjonalności można rozważyć z perspektywy klienta, chcącego zamówić miejsce w restauracji, oraz właściciela i obsługi rezerwacji. Klient dzięki aplikacji powinien mieć zdolność zarezerwowania miejsca w wybranej przez siebie restauracji. Restauratorzy powinni mieć zdolność dodawania własnych rezerwacji oraz potwierdzania rezerwacji danych użytkowników. Aplikacja ma ułatwić i zautomatyzować komunikację między klientami a restauracjami. Warto zaznaczyć, że aplikacja nie udostępnia narzędzi umożliwiających zarządzanie restauracjami. W celu osiągnięcia przedstawionego celu należy stworzyć stronę internetową umożliwiającą dostęp do wybranych funkcjonalności, połączoną z aplikacją webową z dostępem do bazy danych.

2 Analiza wymagań

2.1 Analiza rynkowa

Głównym celem projektu jest ożywienie branży gastronomicznej oraz usprawnienie i automatyzację niektórych procesów w restauracjach. Aplikacja ma na celu częstsze odwiedzanie lokalu przez ludzi, a także do odciążenie właścicieli restauracji i eliminację ksiąg rezerwacyjnych z rezerwacją przez telefon. Aplikacja pozwoli na promocję restauracji poprzez opinie użytkowników czy ilość rezerwacji. Na podstawie ilości rezerwacji w ciągu dnia system może prognozować zapełnienie restauracji, a tym samym dostosować odpowiednią ilość pracowników obsługi restauracji w zależności od dnia tygodnia czy pory dnia.

2.1.1 Dostępne rozwiązania

Na rynku jest dostępnych kilka aplikacji oferujących zbliżony zakres funkcjonalności do przedstawionego. Poniżej przedstawiono pobiczną analizę dostępnych rozwiązań.

gastrobooking.pl Firma Adluna jest polską firmą, która stworzyła serwis gastrobooking.pl umożliwiający rezerwację stolików w restauracjach na terenie Krakowa. Strona dostarcza prosty interfejs użytkownika w którym po wybraniu restauracji, zatwierdzeniu liczby rezerwowanych miejsc i wpisaniu dokładnej daty jest możliwa rezerwacja. Strona pozwala także na wystawienie oceny użytkownika. Dodatkowym atutem strony jest wprowadzone punkty lojalnościowe, które użytkownik po uzbieranej odpowiedniej ilości może wymienić na zniżki w restauracjach. Serwis ma 77 zarejestrowanych restauracji i jest jedynym działającym serwisem na terenie Polski.

quandoo.com Kolejnym przykładem strony rozwiązującej problem rezerwacji stolików w restauracjach jest serwis internetowy quandoo.com stworzony przez firmę Quandoo GmbH. Funkcjonalności oferowane przez Quandoo są bardzo podobne do tych przedstawionych w powyższym przykładzie z tym, że strona jako całość prezentuje się o wiele ładniej i bardziej intuicyjnie od poprzedniego przykładu. Dodatkowym atutem jest aplikacja mobilna, dzięki której jest możliwa rezerwacja za pomocą telefonu. Strona posiada 17 tysięcy zarejestrowanych restauracji w 12 krajach m. in. w Europie i Azji. Przyciąga to wiele użytkowników dzięki czemu serwis jest w stanie funkcjonować w tak wielu krajach.

zomato.com Serwis Zomato.com jest serwisem internetowym posiadającym największą bazę restauracji z całego Świata, także z Polski, jednakże funkcję rezerwacji stolików udostępnia tylko w kilku krajach. Większość funkcji jest taka sama jak w pozostałych przykładach, jedyną różnicą jest to że serwis pokazuje współczynnik popularności danej restauracji w danym tygodniu.

opentable.com Opentable jest jedną z najlepiej ocenianych i najczęściej używanych stron internetowych oferujących rezerwację stolików w restauracjach. Działa ona na terenie 14 krajów i posiada zarejestrowanych 40 000 restauracji. Najwięcej użytkowników znajduje się w Stanach Zjednoczonych.



Rysunek 1: Firmy konkurencyjne.

	gastrobooking.pl	quandoo.com	zomato.com	opentable.com
Popularne	W Polsce	W 12 krajach	W 24 krajach	W 14 krajach
Rezerwacja stolików w polskich restauracjach	Kraków	Niedostępna	Niedostępna	2 restauracje
Recenzje i ocena	✓	✓	✓	✓
Menu	✓	✓	✓	✓
Lokalizacja	✓	✓	✓	✓
Galeria zdjęć	✓	✓	✓	✓
Wskaźnik popularności w danym tygodniu	✗	✗	✓	✓
Wersja mobilna	✗	✓	✓	✓
Ilość zarejestrowanych restauracji	77	17 000	-	40 000

Rysunek 2: Porównanie firm konkurencyjnych.

2.1.2 Analiza wymagań biznesowych

Wymienione w poprzednim paragrafie serwisy nie występują w Polsce, lub są słabo rozpowszechnione. W porównaniu do konkurencji podstawową zaletą aplikacji ma być jej niska cena, oraz prostota. Zwiększa to zakres firm, który mogłyby sobie pozwolić na wdrożenie naszej aplikacji, co na dłuższą metę może stanowić o większej popularności.

Główną grupą docelową naszego produktu są restauratorzy oraz obsługa restauracji z dużą liczbą rezerwacji. Dzięki przygotowanej aplikacji mogą skorzystać z najnowszych rozwiązań technicznych, aby lepiej zarządzać swoją dostępnymi miejscami, oraz personelem. Zintegrowanie opracowanego systemu z innymi systemami umożliwiającym zarządzanie personelem, kosztami czy zamówieniami znacząco ułatwiałoby zarządzanie i podejmowanie właściwych decyzji na poziomie managerskim.

Ryzyka projektu

W projekcie przyjęto oszacowanie ryzyka na podstawie określenia poziomu ryzyka jako iloczynu prawdopodobieństwa wystąpienia i stopnia ryzyka. Wyróżniono następujące prawdopodobieństwa wystąpienia:

1. praktycznie niemożliwe
2. bardzo mało prawdopodobne
3. zdarza się rzadko
4. może się zdarzyć
5. duże prawdopodobieństwo zdarzenia

Wyróżniono następujące stopnie ryzyka:

1. bardzo mały

2. mały
3. średni
4. duży
5. bardzo duży

Stosując opisaną metodykę, wybrano kila najważniejszych czynników i oszacowano poziom ryzyka.

Tabela 1: oszacowanie poziomu ryzyka

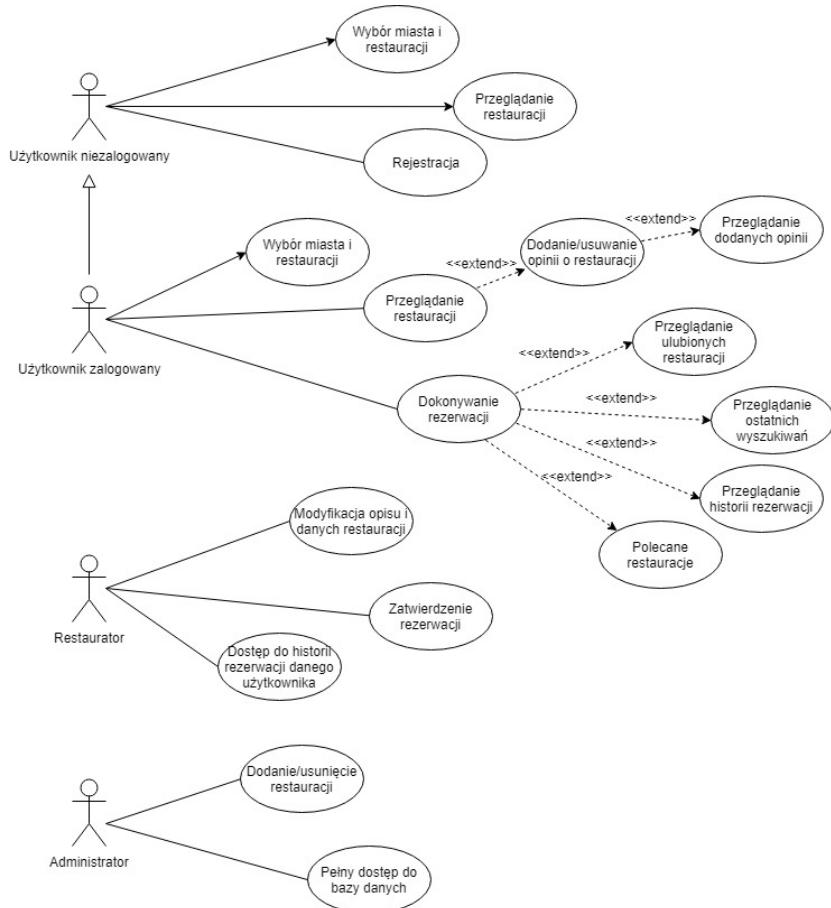
Ryzyko	P(w)	Stopień ryzyka	poziom ryzyka
Nie zaimplementowanie funkcji ze względu na napotkane problemy	4	4	16
Złe oszacowanie harmonogramu projektu (diagram Gantta)	3	4	12
Brak współpracy w zespole , Problem z komunikacją	4	4	16
Awaria sprzętu	3	4	12
Zdarzenia przy projekcie (Utrata danych, złośliwe oprogramowanie , brak zasilania)	2	4	8
Źle dobrane technologie	2	4	8
Przekroczenie budżetu	4	4	16
Złe decyzje projektowe podjęte na początku, mające duży wpływ na ostatnią fazę projektu i powodujące zmiany w systemie	4	5	20
Niedostateczne zabezpieczenie danych	4	4	16
Kłopoty związane z regulacjami prawnymi	2	4	8

2.2 Wymagania funkcjonalne

2.2.1 Podstawowe przypadki użycia

Użytkowników aplikacji można podzielić na 3 grupy. Każda z grup posiada własne przypadki użycia.

1. klient restauracji, niezalogowany użytkownik
 - (a) przeglądanie profilu restauracji
Niezalogowany użytkownik może wyświetlać profil restauracji i ma dostęp do wszystkich informacji zawartych w profilu.
 - (b) rejestracja
Większość przypadków użycia jest dostępnych po rejestracji i zalogowaniu.
2. klient restauracji, zalogowany użytkownik



Rysunek 3: Schemat przypadków użycia systemu.

(a) złożenie rezerwacji

Klient ma możliwość wyboru daty i godziny rezerwacji. Na podstawie informacji o dostępności stolików system daje informację zwrotną, czy rezerwacja w wybranych godzinach jest możliwa. Klient dokonuje rezerwacji, która następnie musi być zatwierdzona przez restaurację.

(b) dodawanie opinii o restauracji

Użytkownik ma możliwość dodatnia opinii, która będzie widoczna dla wszystkich użytkowników w profilu wybranej restauracji.

(c) przeglądanie dodanych opinii

Klient ma dostęp do wcześniej dodanych opinii w swoim profilu.

(d) przeglądanie ulubionych restauracji

Na podstawie historii rezerwacji prezentowana jest lista ulubionych restauracji, która jest prezentowana użytkownikowi po zalogowaniu.

(e) przeglądanie historii rezerwacji

Użytkownik ma dostęp do historii własnych rezerwacji.

(f) wysyłanie wiadomości do restauracji

Klient może się komunikować bezpośrednio z restauracją przez stronę internetową.

(g) polecane restauracje

Na postawie historii rezerwacji, oraz preferencji użytkowników o podobnych cechach prezentowana jest lista restauracji, które mogą się spodobać użytkownikowi. Do wyznaczania listy stosuje się algorytm kNN.

3. restaurator

(a) zmiana informacji odnośnie restauracji

Restaurator posiada możliwość zmiany danych prezentowanych na profilu restauracji. Zaliczają się do nich informacje o nazwie, adresie i zdjęcia. Restaurator nie ma wpływu na opinie widoczne na profilu jego restauracji.

(b) potwierdzenie rezerwacji

Po złożeniu rezerwacji przez użytkownika, musi zostać ona zatwierdzona przez restaurację. W trakcie ewaluacji danej rezerwacji pomocne może być przeglądanie profilu klienta.

(c) wysyłanie wiadomości do klienta Opcja bezpośredniej komunikacji przez stronę internetową.

(d) przeglądanie profilu klienta Restaurator ma dostęp do profilu klienta.

4. administrator

(a) dodanie restauracji Dodawanie restauracji powinno być kontrolowane, aby uniknąć udostępniania nieprawdziwych bądź niewłaściwych informacji użytkownikom. W celu ułatwienia procesu weryfikacji danych, tylko administrator ma możliwość dodania restauracji do systemu.

- (b) pełny dostęp do bazy danych Administrator może dodawać, usuwać, modyfikować i odczytywać wszystkie tabele w bazie.

2.3 Wymagania niefunkcjonalne

2.3.1 Wykorzystane technologie i narzędzia

Projekt został zrealizowany z wykorzystaniem języka C# oraz framework'a ASP.NET MVC. Do realizacji frontendu zostały wykorzystane HTML5, CSS 3.0 oraz JavaScript. Jako system zarządzania bazą danych wykorzystano Microsoft SQL Server.



Rysunek 4: Wykorzystane technologie.

- **Język programowania C#**

C# jest obiektowym językiem programowania, który oficjalnie przedstawiony został w 2000 roku. Powstał na zlecenie firmy Microsoft, aby ułatwić tworzenie oprogramowania dla systemu operacyjnego Windows. W początkowej fazie swojego istnienia język był wspierany wyłącznie przez Microsoft, dlatego też rozpatrując język nieodzownym elementem było platforma programistyczna .NET. Jednak jego dobrze przemyślana struktura sprawiła, że język stał się popularny i powstało dla niego kilka open source standardów (m.in. Mono). Od 2016 roku Microsoft zmieniło swoją politykę i stworzyło .Net Core, które stało się praktycznie w pełni open source projektem.

- **Platforma .NET**

Do roku 2016 .NET rozwijały się dwa podejścia co do środowisk uruchomieniowych, czyli oficjalną wersję opublikowaną przez Microsoft, jak i tworzoną przez pasjonatów wersja open source. Jednak wraz z publikacją .NET Core 1.0 rozwój platformy został połączony w jedno. Wprowadzono ujednolicony standard platformy (.NET Standard Library). Dzięki temu tworzone od teraz biblioteki są możliwe w użyciu również dla .NET Core, Xamarin i .NET Framework. Platforma .NET wspiera nie tylko język C# ale również takie języki jak m.in. F#, Visual Basic, C++. .NET Framework udostępnia specyficzne dla systemu operacyjnego Windows API takie jak Windows Forms i WPF (tworzenie aplikacji okienkowych). .NET Core zakłada całkiem odmienne podejście do przeznaczenia od wcześniejszego standardu. Głównym założeniem jest wieloplatformowość. Przykładem

API są: ASP.NET Core (aplikacje webowe) oraz Universal Windows Platform (docelowo reużywalne oprogramowanie działające zarówno na komputerach, urządzeniach mobilnych, jak i dla aplikacji wspierającej Internetu rzeczy itd.). Mono for Xamarin w tej chwili wersja .NET dla aplikacji działających pod iOS oraz Android. Historycznie opensource'owa wersja standardu .NET Framework.

- **ASP.NET Core 2.0**

ASP.NET to API umożliwiające tworzenie aplikacji webowych[?], czyli takie, które do swojego działania potrzebują jedynie przeglądarki internetowej. Nie wymagają od użytkownika instalowania zewnętrznego oprogramowania. Dzięki temu stworzone aplikację działają na wielu platformach jednocześnie. Dodatkowo wszelkie zmiany wprowadzane są wyłącznie po stronie serwera (nie angażują użytkownika). Do głównych elementów API zaliczamy: Web Forms, MVC (MVC + Web Page + Web API) oraz SignalR. Wprowadzenie standardu .NET Core uporządkowało kwestie związane z różnym podejściem tworzenia aplikacji webowych. Web Forms oraz Web Page praktycznie przestały istnieć.

- **Entity Framework Core**

Entity Framework Core to narzędzie pozwalające przetłumaczyć relacyjne bazy danych na obiekty. Narzędzia te nazywamy ORM (Object Relational Mapping). Dzięki temu podejściu można tworzyć bazy danych bez użycia języka SQL. Ułatwia to zarządzanie danymi oraz przyspiesza pracę z nimi. Ponadto biblioteka ta zapewnia wsparcie nie tylko na poziomie tworzenia zapytań, ale również dzięki podejściu Code First możemy zaprojektować relacyjną bazę danych z poziomu języka obiektowego. Entity Framework pozwala także tworzyć automatycznie obiekty na podstawie już istniejącej bazy. Ostatnim wspieranym podejściem jest tworzenie bazy przy pomocy podejścia Model first polegające na stworzeniu modelu w ADO.NET Entity Data Model Designer, a następnie Entity Framework tłumaczy go na obiekty oraz relacyjną bazę danych.

- **Bootstrap**

Biblioteka zapewniająca głównie style CSS, ale również określa ich zachowanie (wykorzystuje JavaScript). Główną zaletą Bootstrap jest to, że wspomaga tworzenie responsywnych aplikacji webowych, to znaczy, że powinny dobrze wyglądać nie tylko na monitorze, ale również na tablecie, czy telefonie komórkowym. Pomaga w kontrolowaniu zachowania gotowych komponentów HTML.

- **jQuery**

jQuery jest biblioteką JavaScript ułatwiającą manipulację drzewa DOM

(Document Object Model). Wykorzystanie jQuery eliminuje problem interpretacji kodu JS przez różne przeglądarki. Jego niewątpliwą zaletą jest wbudowanie AJAX. Sprawia, że opisane rozwiązanie jest czytelniejsze oraz wprowadza uproszczone odwołanie się do elementów strony, oraz wykorzystania AJAX.

- **SignalR**

SignalR jest biblioteką Microsoft ASP.NET, która pozwala na wysyłanie asynchronicznych powiadomień do aplikacji po stronie klienta. Biblioteka zawiera komponenty JavaScript po stronie serwera i po stronie klienta. Umożliwia dodanie funkcjonalności do aplikacji w czasie rzeczywistym.

3 Projekt systemu

3.1 Architektura systemu

W niniejszym rozdziale przedstawiono ogólny przegląd architektury całej aplikacji, opartej na zalecanych sposobach pracy z wybranymi narzędziami.

3.1.1 ASP.NET MVC

Projekty wykonywane w frameworku ASP.NET MVC wymuszają pewną organizację projektu, która zwiększa uporządkowanie oraz wymusza korzystanie z dobrych praktyk. W projekcie można wyróżnić 4 zasadnicze części: Views, Controllers, Models i Router. Poniżej przedstawiono krótki opis poszczególnych części.

Model Modele są odpowiedzialne za przechowywanie informacji domenowej i stanu aplikacji. Najczęściej są implementowane jako Plain Old CLR Object (POCO) i służą do modelowania danych z bazy. Obiekty te są niezależne od frameworków, systemu zarządzania bazą danych czy mapowań ORM. Modele mogą być łączone w ViewModele, aby ułatwić grupowanie i przekazywanie informacji w aplikacji.

View Jest to frontendowa część projektu, która determinuje wygląd strony pokazywanej użytkownikowi. Technologie wykorzystywane w tym obszarze to HTML, CSS oraz JavaScript. Dodatkowo do widoku są przesyłane dane z innych komponentów w postaci modeli lub viewmodeli. ASP.NET MVC umożliwia dodawanie fragmentów kodu w C# do ułatwienia wykorzystywania danych z modeli.

Controller Najczęściej jest to element implementujący wybraną część logiki biznesowej. Wykorzystuje dane z Modeli, aby wytworzyć i przekierować użytkownika do odpowiedniego Widoku. Każdy controller jest odpowiedzialny za ob-

slugę HTTP Request. Taka organizacja wymusza podział projektu na mniejsze klasy z dobrze zdefiniowanym zakresem odpowiedzialności.

Router Jest to komponent, który odpowiada za mapowanie zapytań HTTP na poszczególne akcje w kontrolerach.

Pierwsze trzy komponenty są dobrze znane ze wzorca projektowego na Model-View-Controller, natomiast ostatni jest często spotykanym dodatkiem w wielu frameworkach sieciowych, umożliwiający łatwe mapowanie. Konstrukcja frameworku MVC pozwala na łatwe rozdzielenie odpowiedzialności komponentów i klas oraz wstrzykiwanie zależności do poszczególnych klas.

3.1.2 Architektura warstwowa

Wzorzec programowania oparty na architekturze warstwowej (ang. tier architecture) jest powszechny w procesie tworzenia aplikacji internetowych. Podejście te rozdziela logikę systemu od interfejsu użytkownika. W podejściu architektury trzywarstwowej wyróżniamy następujące warstwy:



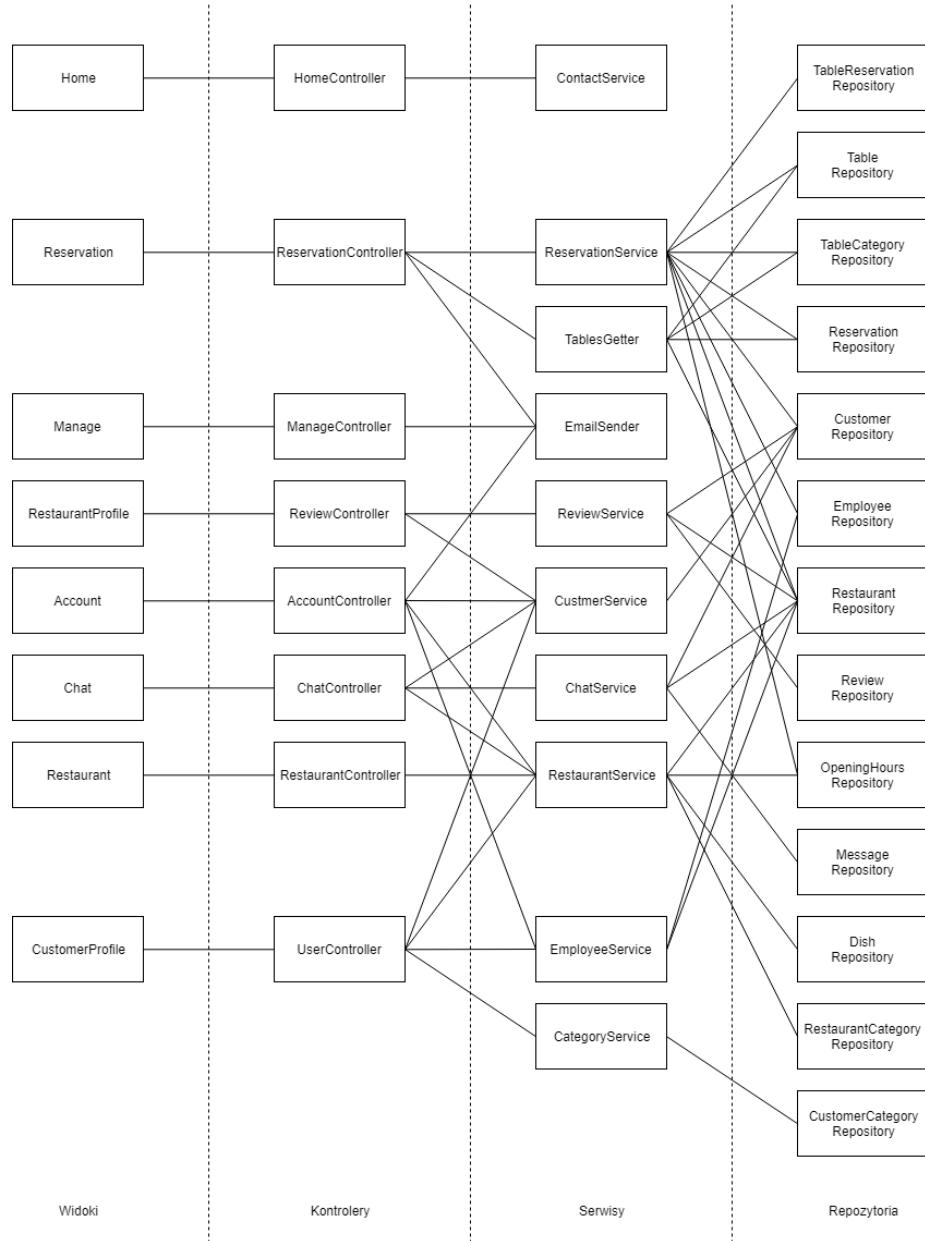
Rysunek 5: Warstwy w architekturze wielowarstwowej.

- **Warstwa prezentacji (ang. presentation tier)** - odpowiedzialna za przedstawienie funkcjonalności użytkownikowi, umożliwiająca wyświetlanie oraz wprowadzanie danych. W tej warstwie ASP.NET używa się oprócz języka platformy (np. C#) języków do opisów wyglądu strony (CSS, HTML, JavaScript).
- **Warstwa biznesowa (ang. business tier)** - w miejscu tym umieszczona jest logika aplikacji. Dane pozyskane z warstwy danych są przetwarzane by były w odpowiedni sposób przygotowane do wysyłania do warstwy prezentacji, jak również pozyskane od użytkownika przekształcane są w oczekiwanej formie, transportowane są do warstwy danych.

- **Warstwa danych (ang. persistance tier)** - odpowiedzialna za komunikacji z zewnętrznymi serwisami zarządzającymi danymi, np. z bazą danych.

Dzięki zastosowaniu tego podejścia zyskujemy łatwość modyfikacji poszczególnych funkcjonalności, ponieważ zmiana w jednej warstwie nie wymusza zmian w całym systemie. Inną zaletą jest fakt, iż tworzone rozwiązania stają się bardziej przejrzyste. Za używaniem tego sposobu projektowania aplikacji przemawia to, że tę samą logikę możemy wykorzystać zarówno przy tworzeniu aplikacji internetowych jak i mobilnych. Wykorzystanie architektury warstwowej pozwala na reużywalność zaimplementowanych komponentów oraz ogranicza liczbę błędów (poprzez użycie wstrzykiwania zależności lub interfejsów), poza bieżącą warstwą nie ma dostępu do wewnętrznych metod. W stworzonym systemie oprócz przedstawionego podziału wprowadzono dodatkowy element przytrzymujący modele wykorzystywane między warstwami. Ich szczególnym przykładem są modele DAO, które używane są wyłącznie przez warstwę danych do komunikacji z bazą danych (mapowane są one przy użyciu Automappera). Dla realizacji założeń każda warstwa zaimplementowana została w osobnym projekcie.

3.2 Diagram elementów systemu

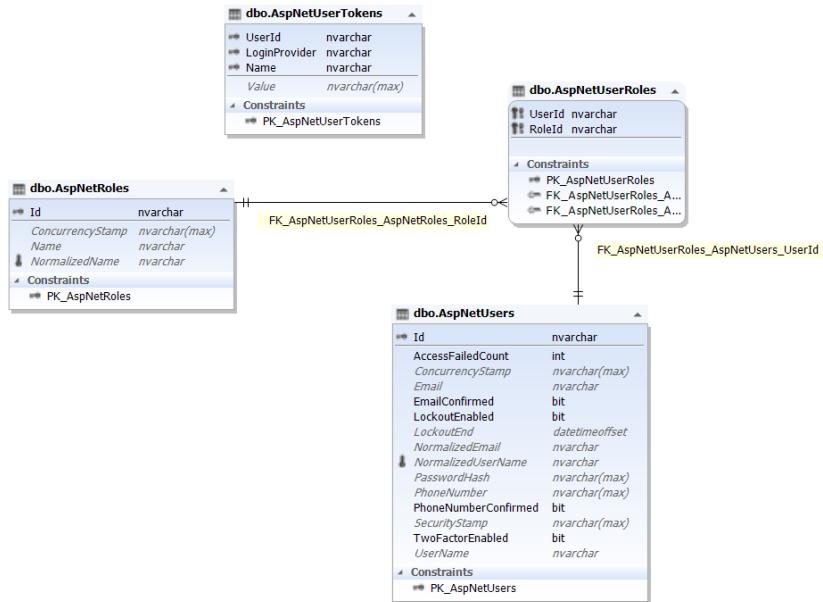


Rysunek 6: Diagram elementów systemu.

Zgodnie z przyjętą architekturą kod projektu można podzielić na cztery części: Widoki, Kontrolery, Serwisy, oraz Repozytoria. Odpowiedzialności poszczególnych części opisano w rozdziale 3.1. Kontrolery uzyskują dostęp do Serwiów poprzez wstrzykiwanie zależności. Ten sam mechanizm wykorzystano przy zapewnieniu dostępu serwisom do Repozytoriów. Poprzez wizualną analizę diagramu 6 można wyróżnić elementy systemu wymagające poprawy. Definitywnie pierwszym kandydatem do procesu refactoryzacji kodu jest ReservationService. Wniosując po ilości powiązań z Repozytoriami klasa ta ma zbyt dużą odpowiedzialność i powinna zostać podzielona na mniejsze komponenty. Podobne wnioski można wyciągnąć na temat RestaurantService. Mimo tych niedoskonałości przy implementacji systemu zachowano warstwową architekturę.

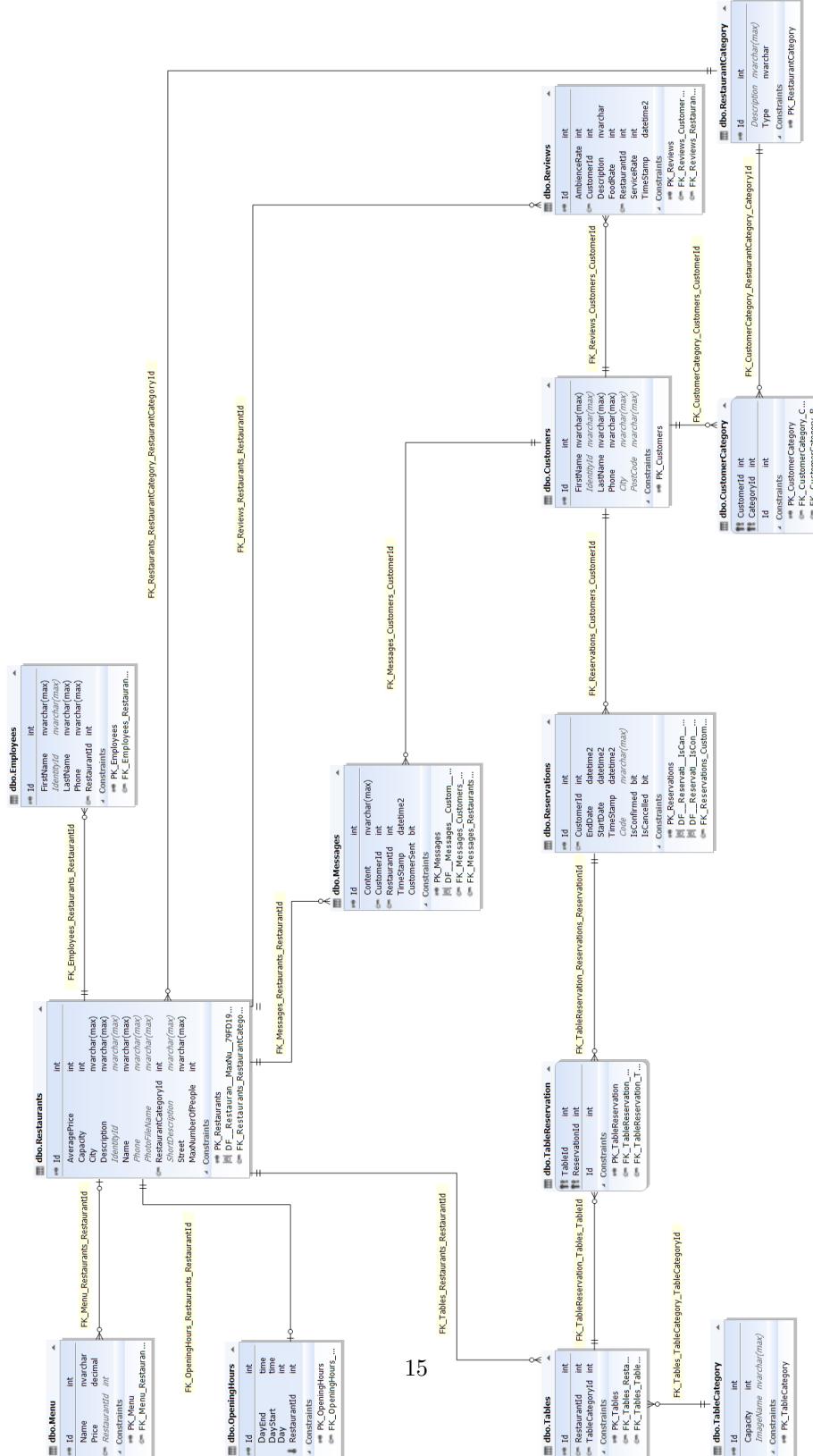
3.3 Projekt bazy danych

Aplikacja wykorzystuje dwie bazy danych. Jedna baza danych (rysunek 5) jest używana do zarządzania użytkownikami - dane logowania, ich role i inne dane potrzebne do weryfikacji. Tabele, jak i relacje w tej bazie danych zostały utworzone przy użyciu biblioteki ASP.NET Core Identity ze względu na szereg udostępnianych przez nią rozwiązań m.in. do logowania, rejestracji, przypomniania czy resetowania haseł.



Rysunek 7: Schemat bazy danych zarządzającej użytkownikami.

Drugą bazą danych jest baza zawierająca tabele potrzebne do obsługi aplikacji. Baza danych 33 jest powiązana przez unikalne pole *IdentityId* w tabeli *Customers/Restaurants* z polem *Email* w bazie danych 32.

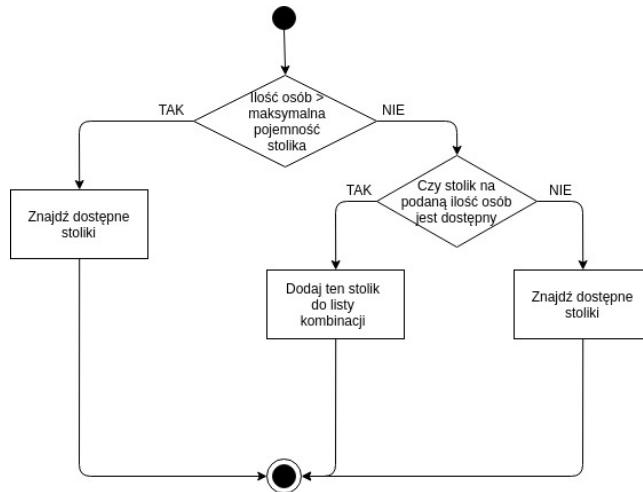


Rysunek 8: Schemat bazy danych aplikacji.

3.4 Algorytm wyszukiwania stolików

W trakcie składania zamówienia użytkownik aplikacji podaje liczbę osób, dla których mają zostać zarezerwowane miejsca. W celu określenia czy złożenie zamówienia w wybranych godzinach jest możliwe opracowano algorytm, tworzący kombinację stolików dostępnych w wybranym czasie. Wejście algorytmu stanowią: idRestaracji, ilość osób, dla których ma zostać zrealizowana rezerwacja, godzina rozpoczęcia rezerwacji oraz czas trwania rezerwacji. Wyjście algorytmu stanowi lista kombinacji stolików dostępnych w wybranym czasie mogących pomieścić wszystkie osoby, dla których została zrealizowana rezerwacja.

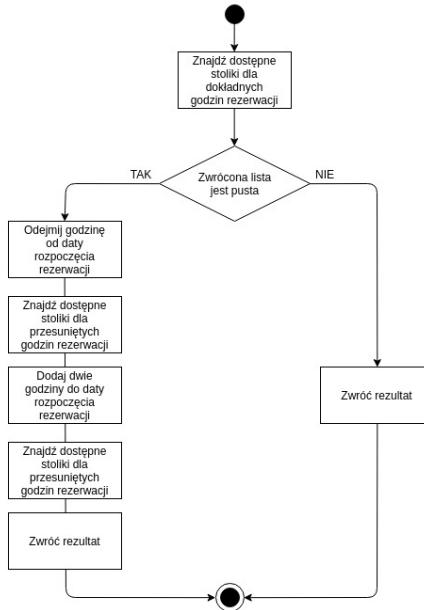
Przy projektowaniu algorytmu wykorzystano technikę dziel i rządź. Na każdym etapie dokonywano odcięć w celu podziału zadania na mniejsze i lepiej określone problemy.



Rysunek 9: Pierwszy krok działania algorytmu

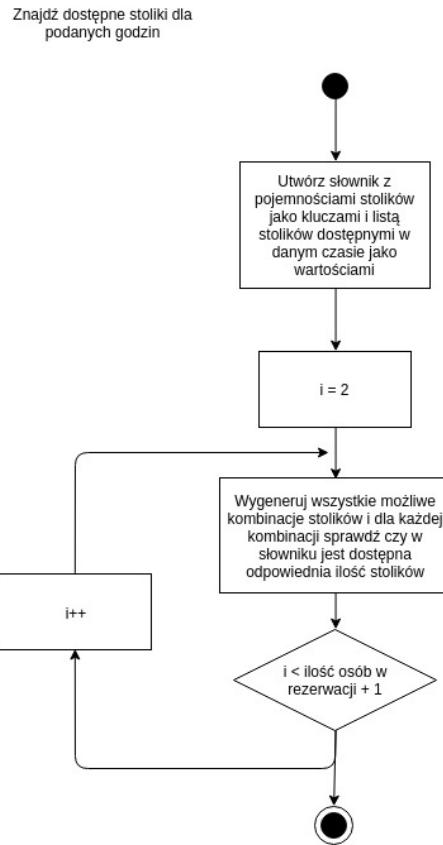
Algorytm w pierwszym kroku (rys. 34) określa czy podana w rezerwacji ilość osób może zostać posadzona przy jednym stoliku. Jeśli tak, sprawdzane jest czy stoliki o pojemności określonej równej ilości osób z rezerwacji są dostępne w wybranym czasie. Jeśli są dostępne algorytm kończy działanie i zwraca znaleziony stolik. Jeśli stolik nie jest dostępny, możliwe jest uzyskanie odpowiedniej ilości miejsc poprzez zajęcie mniejszych stolików (przykładem może być zarezerwowanie dwóch dwuosobowych stolików dla czterech osób). Odpowiedzialna jest za to procedura *Znajdź dostępne stoliki* opisana poniżej. Procedura ta jest także wywoływana w przypadku gdy ilość osób w zamówieniu jest większa od stolika z największą pojemnością.

Znajdź dostępne stoliki



Rysunek 10: Znajdź dostępne stoliki

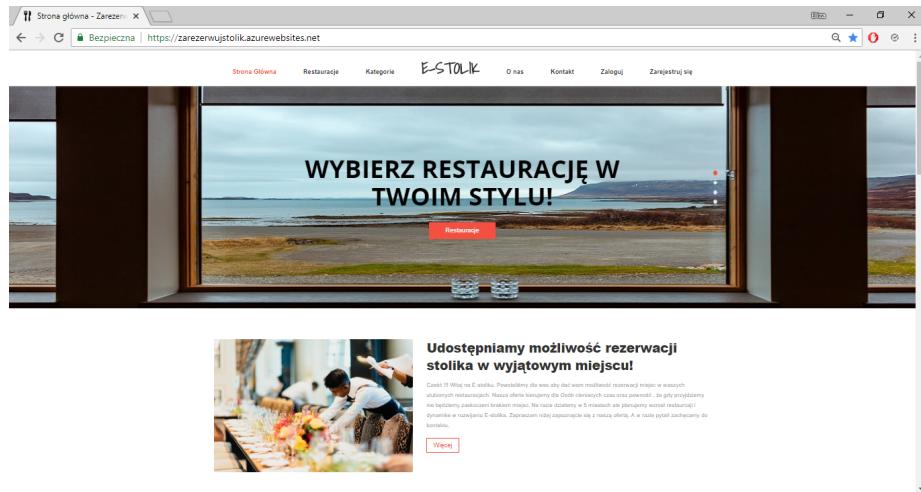
Procedura *Znajdź dostępne stoliki* określa czy są dostępne kombinacje stolików w podanych przez użytkownika godziny (rys. 35). Jeśli zostanie znaleziona dowolna kombinacja będąca w stanie pomieścić ilość osób z rezerwacji procedura kończy działanie. W przeciwnym wypadku procedura modyfikuje godziny rozpoczęcia rezerwacji - dodaje i odejmuje godzinę od oryginalnej godziny rozpoczęcia rezerwacji i szuka kombinacji dla zmodyfikowanych dat.



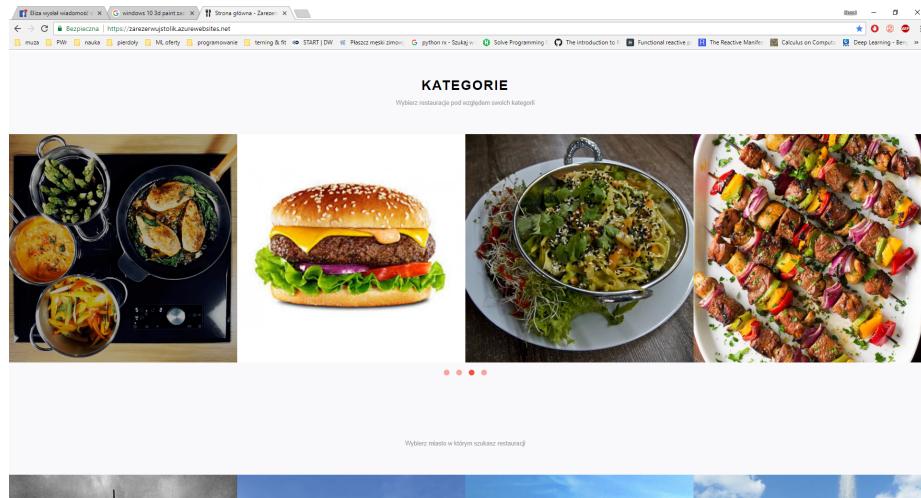
Rysunek 11: Znajdź dostępne stoliki dla podanych godzin

Faktyczne szukanie kombinacji (rys. 36) odbywa się z wykorzystaniem słownika mapującego pojemność stolików na listę stolików dostępnych w danym czasie o danej pojemności. Następnie, iterując po ilości stolików wyznacza się wszystkie możliwe kombinacje stolików (np. realizując zamówienie dla 10 osób przy użyciu dwóch stolików następujące kombinacje mogą zostać użyte dla zrealizowania zamówienia: $\{\{8,2\},\{7,3\},\{6,4\},\{5,5\}\}$). Dla wyznaczonej kombinacji sprawdza się, czy słownik zawiera odpowiednią ilość stolików dostępnych w danym czasie. Jeśli istnieją wolne stoliki potrzebne do utworzenia danej kombinacji w słowniku, kombinacja zostaje dodana do listy rozwiązań.

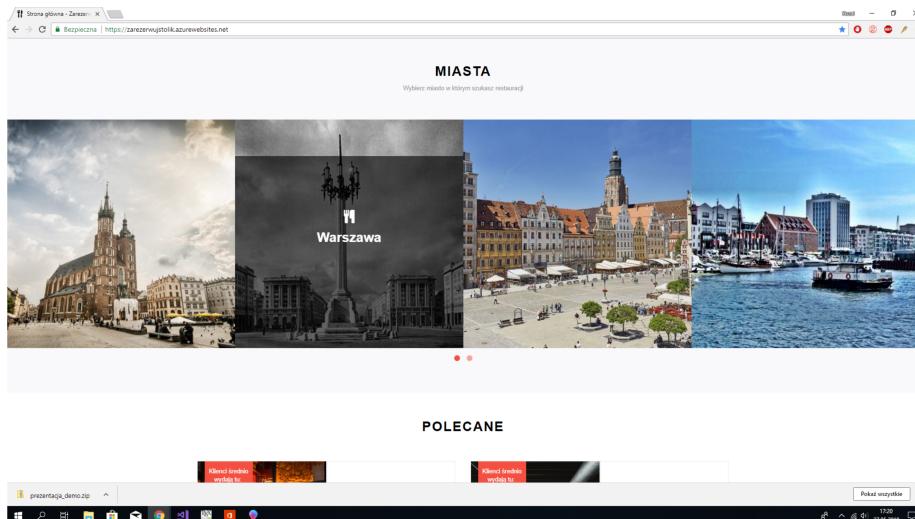
3.5 Projekt frontendu



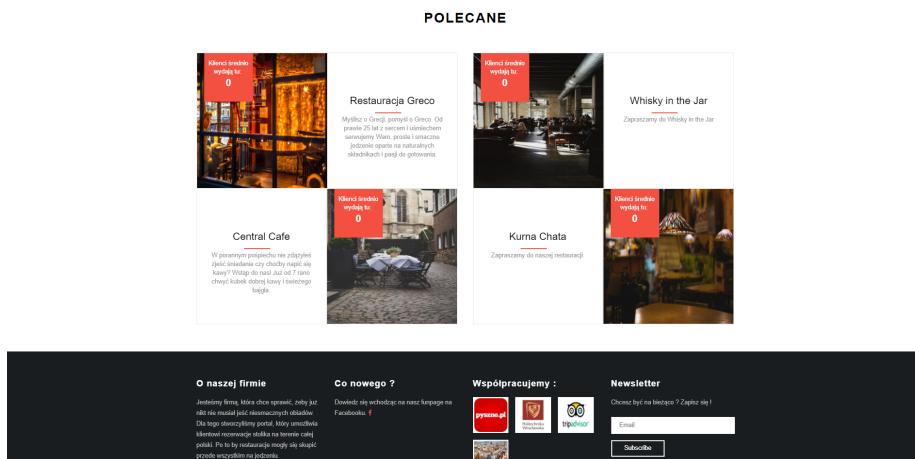
Rysunek 12: Widok strony głównej.



Rysunek 13: Widok kategorii na stronie głównej.



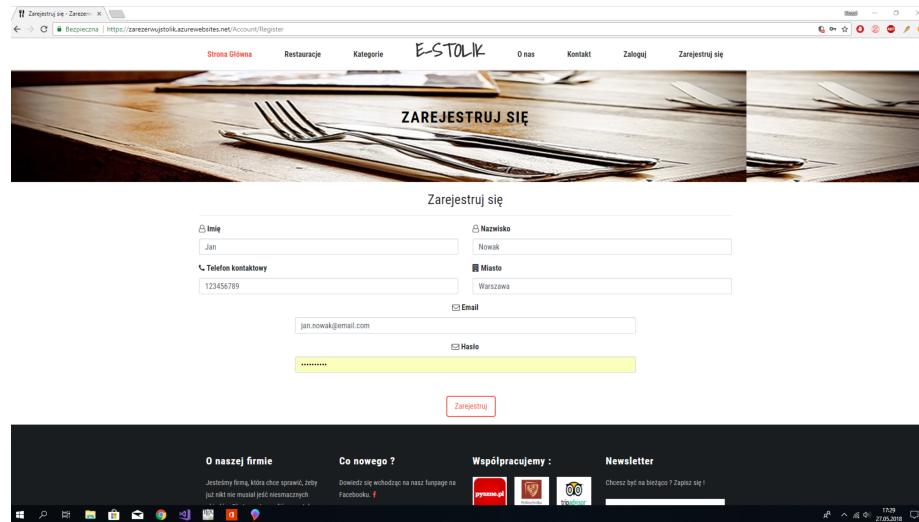
Rysunek 14: Widok miast na stronie głównej.



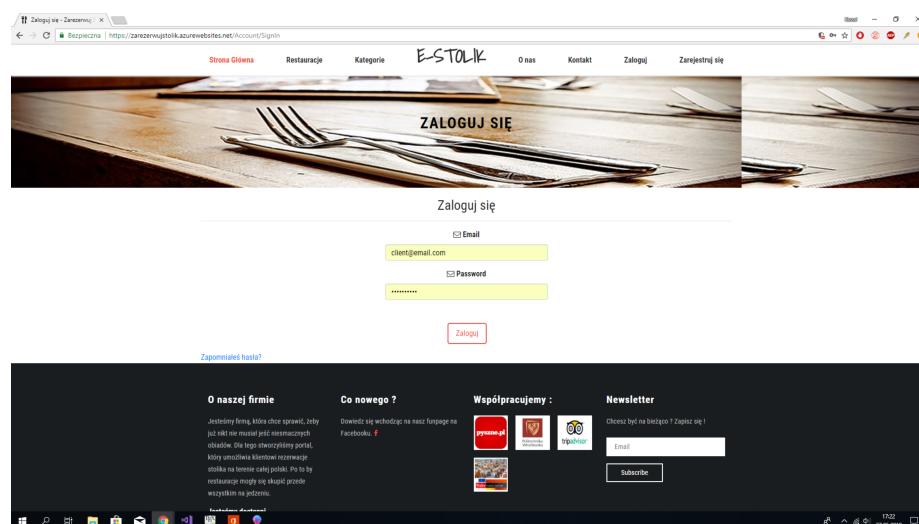
Rysunek 15: Widok polecanych restauracji.

Na stronie głównej znajdziemy odnośniki do wszystkich części systemu. Pasek nawigacyjny u góry strony pozwala na przełączanie się między widokami i łatwe poruszanie się po stronie. Jest on dostępny we wszystkich widokach w systemie. Widok strony głównej (rys. 7-10) zawiera informacje na temat dostępnych kategorii, miast oraz polecanych restauracji. Po wybraniu jednej z kategorii (rys. 8) użytkownik jest przekierowany do widoku zawierającego listę restauracji z wybraną kategorią. Analogicznie, po wybraniu miasta (rys. 9) użytkownik jest przekierowywany do widoku zawierającego listę restauracji w wybranym mieście. Sekcja polecane zawiera listę rekomendowanych restauracji po wybraniu,

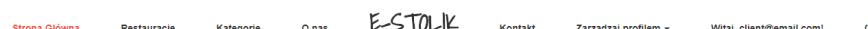
użytkownik jest przekierowywany do widoku ze szczegółami rezerwacji.



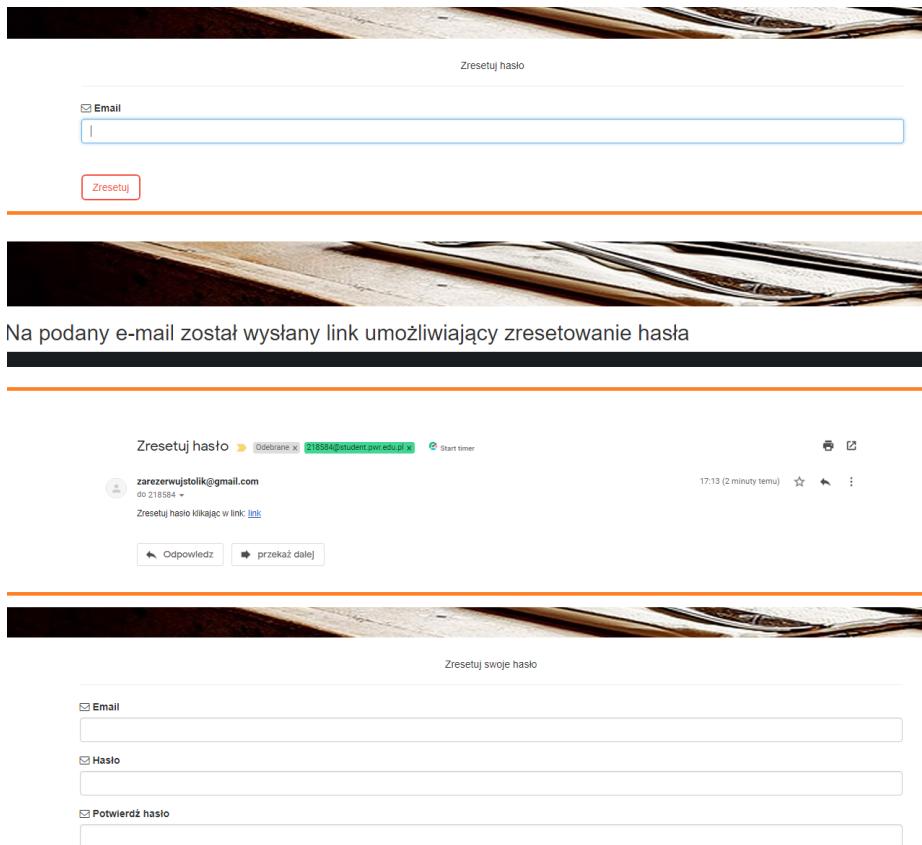
Rysunek 16: Widok rejestracji.



Rysunek 17: Widok logowania.

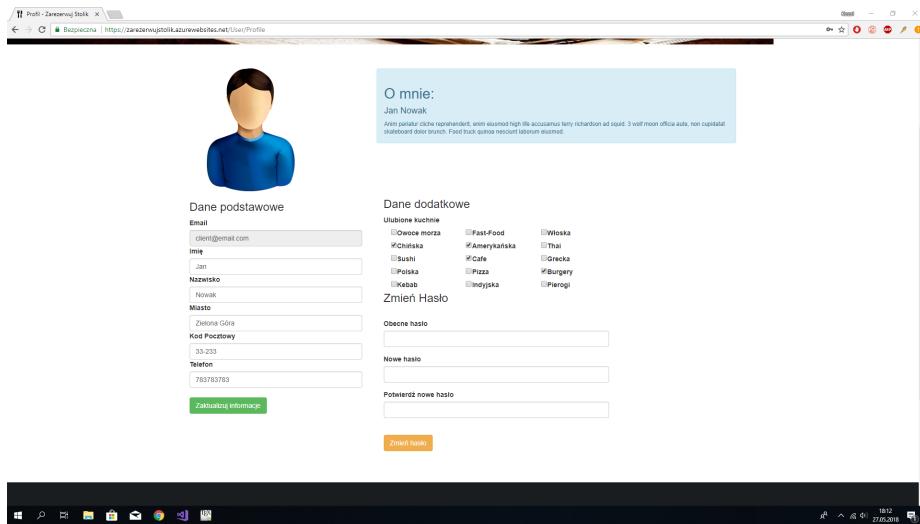


Rysunek 18: Pasek nawigacyjny po zalogowaniu.



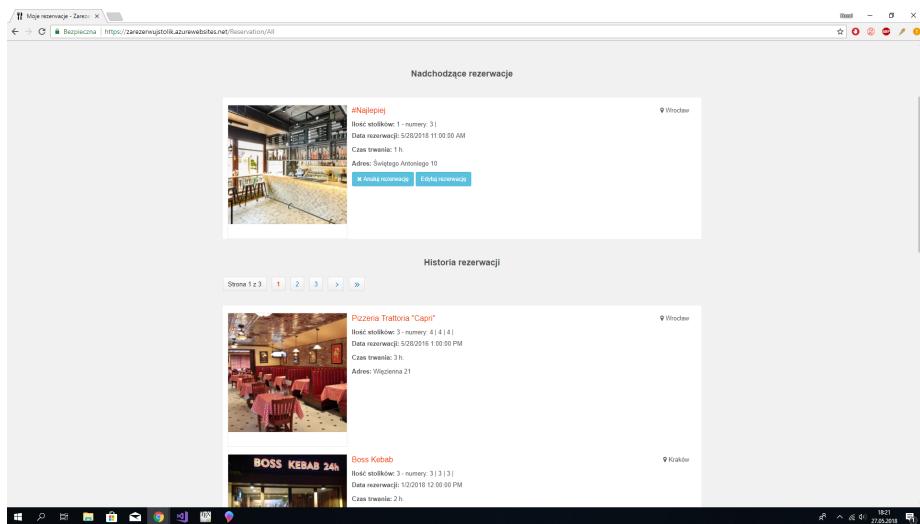
Rysunek 19: Resetowanie hasła.

Aby uzyskać możliwość rezerwowania stolików klient musi być zalogowany. W tym celu należy się zalogować (widok na rys. 12) lub zarejestrować (widok na rys. 11) z potwierdzeniem e-mailowym założenia konta. Po zalogowaniu fakt nazwa użytkownika będzie widoczna zamiast przycisku zaloguj na pasku nawigacyjnym (rys. 13). Jeżeli użytkownik zapomniał hasła może je zresetować¹⁴.



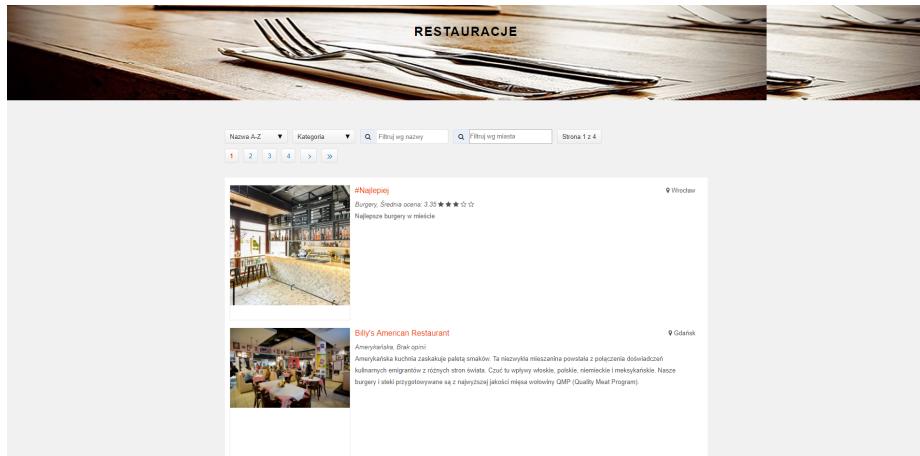
Rysunek 20: Widok szczegółów konta użytkownika.

Po zalogowaniu użytkownik może przeglądać szczegóły swoje konta. W pasku nawigacyjnym należy wybrać "Zarządzaj profilem" a następnie z listy wybranej "Mój profil", aby zyskać dostęp do danych osobowych, oraz wyboru ulubionych kategorii. W tym samym widoku znajduje się również opcja zmiany hasła (rys. 15).



Rysunek 21: Widok historii rezerwacji.

Po wybraniu "Zarządzaj profilem" oraz "Moje rezerwacje" użytkownik może przeglądać nadchodzące rezerwacje oraz historie rezerwacji (rys. 16).



Rysunek 22: Widok dostępnych restauracji.



Amerykańska kuchnia zaskakuje paletą smaków. Ta niezwykła mieszanka powstała z połączenia doświadczeń kulinarnych emigrantów z różnych stron świata. Czuć tu wpływy włoskie, polskie, niemieckie i meksykańskie. Nasze burgery i steki przygotowywane są z najwyższej jakości mięsa wołowniny QMP (Quality Meat Program).

Opinie

Średnia ocena klientów: 3,78

Iwona Matys
Jedzenie ★★★★☆ Atmosfera ★★★★☆ Obsługa ★★★☆☆
25.05.2018 19:47:37

Marcin Kowal
Jedzenie ★★★★★ Atmosfera ★★★★★ Obsługa ★★★★★
Przepyszny burger, duży wybór dobrych piw. Mają bardzo ciekawe opcje hamburgerów na przykład z masłem orzechowym. Fani ostrego jedzenia też znajdą tu dla siebie miejsce. A to za sprawą dostępnego hamburgera z papryczkami habanero. Zdecydowanie polecam to miejsce! :-)
25.05.2018 19:30:29

Jan Nowak
Jedzenie ★★★☆☆ Atmosfera ★★★★★ Obsługa ★★★☆☆
Świetna atmosfera i pyszne jedzenie.
25.05.2018 18:56:40

Zostaw swoją opinię

Jedzenie ★★★☆☆
Obsługa ★★★☆☆
Atmosfera ★★★☆☆

Message

Wyślij

Godziny otwarcia

Dzień Tygodnia	Godziny otwarcia
Poniedziałek	09:00 - 20:00
Wtorek	09:00 - 20:00
Środa	09:00 - 20:00
Czwartek	09:00 - 20:00
Piątek	09:00 - 20:00
Sobota	09:00 - 20:00
Niedziela	09:00 - 20:00

Menu restauracji

#	Menu	Cena
Zarezerwuj stolik		

Adres

Gdańsk, ul. Rajska 10



Popularne godziny



Godzina	Zapelnienie
09:00	0.05
10:00	0.08
11:00	0.15
12:00	0.25
13:00	0.35
14:00	0.45
15:00	0.55
16:00	0.65
17:00	0.55
18:00	0.45
19:00	0.35
20:00	0.25

Billy's American Restaurant

Czy przyjmuję państwo płatności SUDEXO?
client@email.com 20:33:45

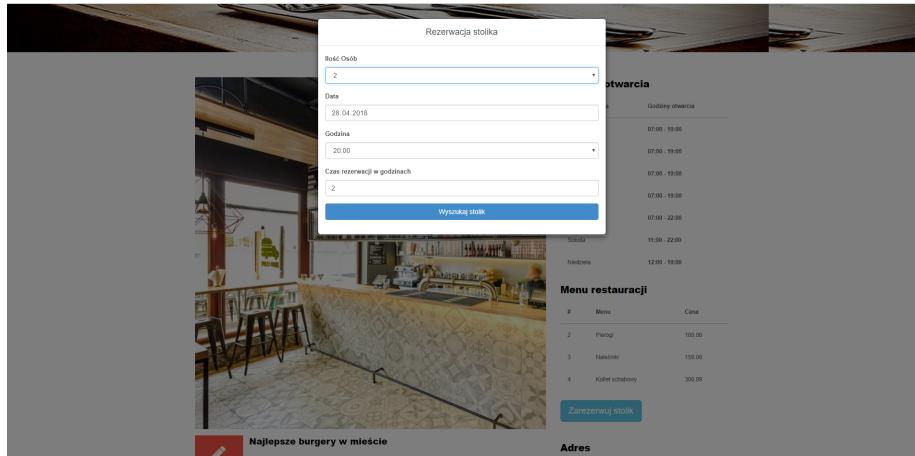
Wyślij

Newsle

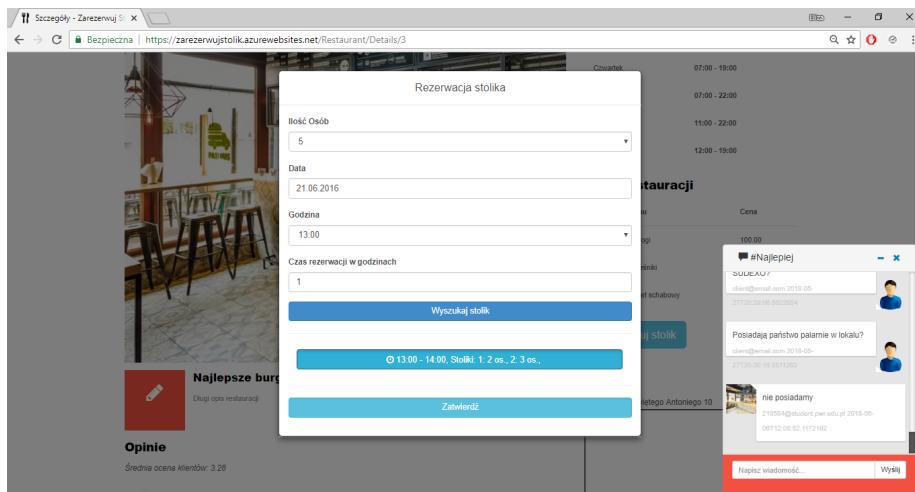
Napisz wiadomość Wyślij

Chcesz być

Rysunek 23: Widok restauracji.



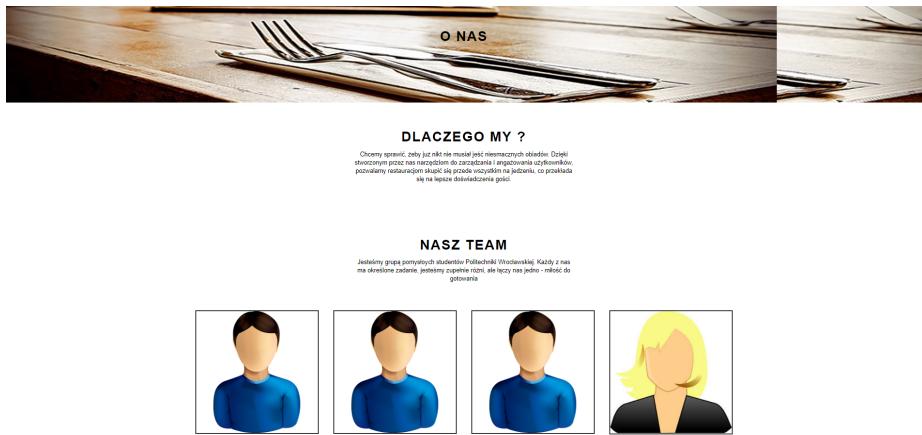
Rysunek 24: Widok rezerwacji stolika.



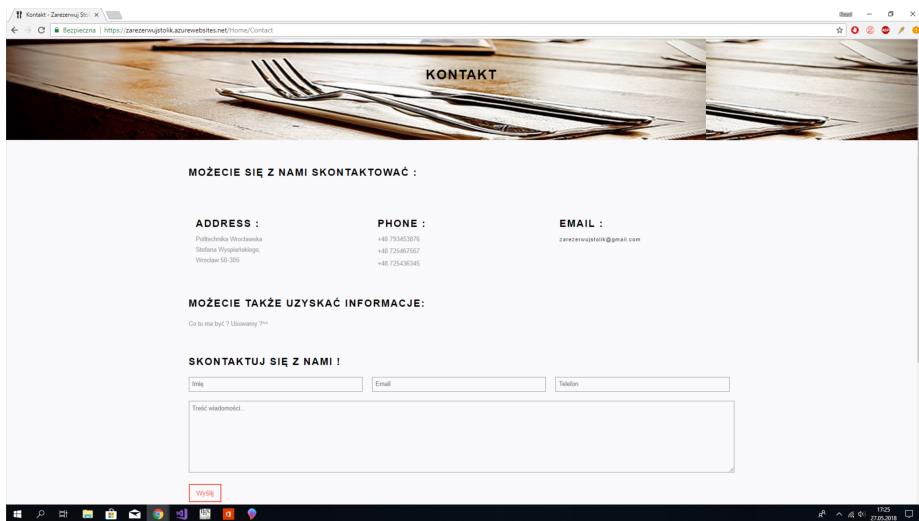
Rysunek 25: Widok rezerwacji stolika - wybór godziny i stolików(propozycje).

Użytkownik może przeglądać dostępne restauracje w widoku restaurants, dostępnym przez przycisk w pasku nawigacyjnym "Restauracje". Po wybraniu restauracji użytkownik jest przekierowywany do widoku zawierającego szczegóły restauracji (ref. 18). Widok zawiera godziny otwarcia rezerwacji, opinie innych użytkowników, dane restauracji, opis, popularne godziny oraz chat. Użytkownik dokonuje rezerwacji przez naciśnięcie "Zarezerwuj stolik". W okienku należy wybrać ilość osób dla których ma zostać złożona rezerwacja, data rezerwacji, godzinę i czas trwania rezerwacji. Po wciśnięciu "Wyszukaj stolik" aplikacja sprawdza czy stoliki są dostępne w danym momencie i prezentuje dostępne kombinacje stolików w danych godzinach. Po zatwierdzeniu rezerwacji użytkownik

otrzymuje maila ze szczegółami rezerwacji. Restaurator musi potwierdzić rezerwację, aby się odbyła.



Rysunek 26: Widok zawierający informacje o zespole tworzącym aplikację.



Rysunek 27: Widok zawierający dane kontaktowe twórców aplikacji.

Widoki dostępne po naciśnięciu w pasku nawigacyjnym przycisku "O nas", oraz "Kontakt" zawierają informacje na temat zespołu tworzącego aplikację (rys. 21), oraz dane kontaktowe (rys. 22).

Strona Główna Restauracje Kategorie O nas E-STOLIK Kontakt Zarządzaj profilem Witaj, client@email.com!

ZAREJESTRUJ RESTAURACJĘ

1 Krok 1 2 Krok 2 3 Krok 3 4 Krok 4

Informacje podstawowe

Nazwa restauracji **Miejsce**

Kategoria **Ulica**

Dalej

Rysunek 28: Pierwszy krok rezerwacji restauracji.

Strona Główna Restauracje Kategorie O nas E-STOLIK Kontakt Zarządzaj profilem Witaj, client@email.com!

ZAREJESTRUJ RESTAURACJĘ

1 Krok 1 2 Krok 2 3 Krok 3 4 Krok 4

Szczegóły

Krótki opis

Dłuższy opis restauracji

Zdjęcie

Dalej

O naszej firmie **Co nowego?** **Współpracujemy:** **Newsletter**

Rysunek 29: Drugi krok rezerwacji restauracji.

Godziny otwarcia	09:00	15:00
Poniedziałek	09:00	15:00
Wtorek	09:00	15:00
Środa	09:00	15:00
Czwartek	09:00	15:00
Piątek	09:00	15:00
Sobota	09:00	12:00
Niedziela	09:00	12:00

Rysunek 30: Trzeci krok rezerwacji restauracji.

Rysunek 31: Czwarty krok rezerwacji restauracji.

Aby dokonać rejestracji restauracji (rys. 23-26) należy podać kolejno: dane podstawowe, opisy restauracji, godziny otwarcia oraz mail, telefon i hasło do konta. Rejestracja jest dostępna na stronie głównej. W celu odfiltrowania fałszywych kont rejestracja restauracji musi być potwierdzona przez administratora systemu.

Użytkownik restauracji ma możliwość tak jak użytkownik przeglądania profili restauracji, przeglądania i anulowania rezerwacji, edycji swojego profilu, komunikowania się z klientami, a także dodawania/usuwania pracowników należących do swojej restauracji. Taki pracownik nie ma możliwości edycji danych restauracji, ale ma pełną kontrolę nad rezerwacjami oraz czatem z klientami.



Rysunek 32: Pasek nawigacyjny restauracji.

The screenshot shows the 'User/RestaurantProfile' page. The top section is titled 'Opis restauracji:' and contains fields for 'ShortDescription' (Najlepsze burgery w mieście) and 'Description' (Długi opis restauracji). Below this is a section for 'Dane podstawowe' (Basic data) with fields for IdentityId (218554@student.pwr.edu.pl), Name (MojLepki), Phone (123456789), City (Wrocław), Street (Ulica Świętego Antoniego 10), and a 'Zaktualizuj informacje' (Update information) button. The 'Dane dodatkowe' (Additional data) section includes 'Kategoria Restauracji' (Burgery), 'Godziny Otwarcia' (Opening hours) for Monday through Sunday, and a 'Zmień Hasło' (Change password) section with fields for 'Obecne hasło', 'Nowe hasło', and 'Potwierdź nowe hasło'. Below these are fields for 'Ilość miejsc' (Number of seats) (6), 'PhotoFileName' (Wybierz plik), and a 'Zaktualizuj informacje' button. The 'Menu restauracji' (Restaurant menu) section lists items: Pierogi (100.00), Naleśniki (150.00), and Kotlet schabowy (300.99). A 'Dodaj' (Add) button is at the bottom of the menu table.

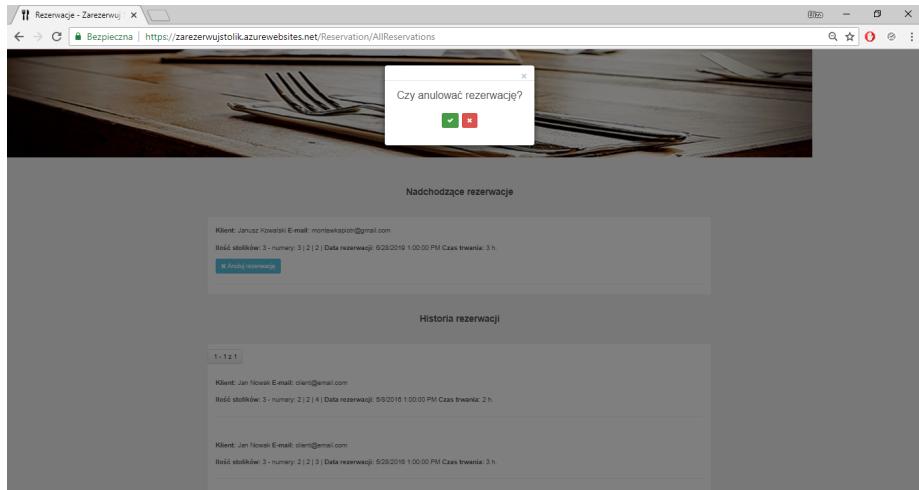
Rysunek 33: Restauracja - zarządzanie profilem.

Imię	Nazwisko	Telefon	E-mail	Usuń
Maria	Kowalska	123456789	employee1@email.com	
Marian	Kowalski	887766554	employee2@email.com	
Maria	Nowak	889988998	employee3@email.com	

Rysunek 34: Lista pracowników.

Czy usunąć konto pracownika?

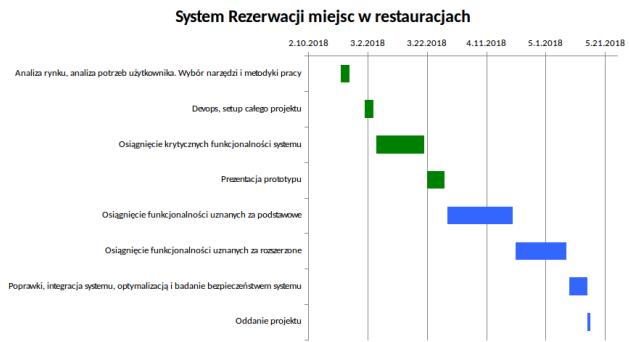
Rysunek 35: Lista pracowników.



Rysunek 36: Rezerwacje klientów w restauracji.

4 Kamienie milowe i plan projektu

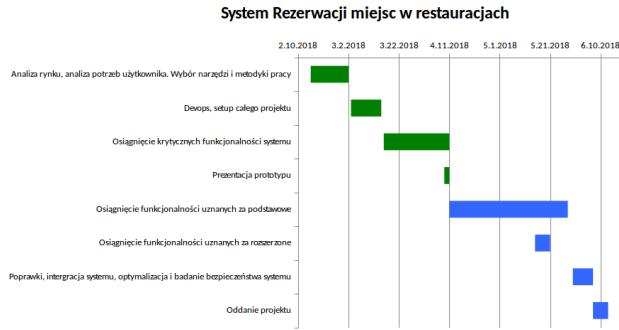
4.1 Wykres Gantta



Rysunek 37: Wykres Gantta.

Na rys. 38 przedstawiono wykres Gantta opracowany przed przystąpieniem do pracy nad projektem. Niestety przy opracowaniu tego wykresu poczyniono zbyt optymistyczne założenia, przez co nie udało się zrealizować planu przedstawionego na wykresie.

Na rys. 38 przedstawiono rzeczywisty wykres Gantta. Na osi czasu można zauważyc przesunięcie spowodowane wydłużeniem pracy nad krytycznymi oraz podstawowymi funkcjonalnościami systemu. Wszystkie opóźnienia spowodowały przesunięcie w czasie oddania projektu o około miesiąc.



Rysunek 38: Rzeczywisty wykres Gantta.

4.2 Rzeczywista kalkulacja kosztów i nakładów pracy

Przed przystąpieniem do pracy nad projektem oszacowano kosztu projektu uwzględniając honorarium czteroosobowego zespołu w wielkości 25 złotych za godzinę, oraz ubezpieczenie i podatki. Jako czas poświęcony na projekt przez każdego członka zespołu przyjęto 90 godzin. Jest to ilość godzin przewidzianych dla tego kursu w karcie przedmiotu. W wyniku obliczeń uzyskano:

łączne wynagrodzenie: 9000 zł
 ubezpieczenie (20.81%): 1872.9 zł
 koszt bezpośredni: 10872.9 zł
 koszt pośredni: 0 zł
 koszt całkowity: 10872.9 zł
 podatek VAT (23%): 2500,767 zł
 łączny koszt projektu: 13373,667 zł

W wyniku opóźnień i błędnej estymacji czas poświęcony na realizację projektu zwiększył się do godzin. Było to powodem wzrostów kosztu projektu. Poniżej umieszczono czas poświęcony na projekt przez członków grupy, oraz rzeczywiste koszta projektu (z uwzględnieniem tej samej stawki):

Eliza Mocek: 130h
 Piotr Monetwka: 100h
 Jędrzej Kozal: 80h
 Hubert Duś: 80h

łączne wynagrodzenie: 9750 zł
 ubezpieczenie (20.81%): 1872.9 zł
 koszt bezpośredni: 11778.98 zł
 koszt pośredni: 0 zł
 koszt całkowity: 11778.98 zł
 podatek VAT (23%): 2709,16 zł
 łączny koszt projektu: 14488,14 zł

Jak widać opóźnienia przyczyniły się do wzrostu łącznego kosztu projektu o ponad 1000 złotych. W związku z nierównym udziałem i nakładem pracy

różnych członków projektu proponuje się uwzględnienie godzinowego nakładu pracy członków zespołu przy wystawianiu oceny za projekt.

5 Wdrożenie projektu

Aplikacja sieci Web jak i baza danych została wdrożona z użyciem usług Microsoft Azure. Utworzony został serwer SQL zawierający dwie bazy danych (pierwszy zawierający bazę danych użytkowników oraz ich uprawnienia, drugi zawierający pozostałe tabele) oraz serwer aplikacji. Od docelowego użytkownika wymagana jest przeglądarka z dostępem do internetu. Aplikacja webowa została przetestowana w przeglądarce Google Chrome wersji 65 oraz Microsoft Edge 41.

5.1 Adres URL

Aplikacja jest dostępna pod adresem:

<https://zarezerwujstolik.azurewebsites.net>

5.2 Konta testowe

Przygotowane zostały dwa konta testowe:

Konto restauracji:

Nazwa użytkownika: restaurant@email.com Hasło: Password1!

Konto klienta:

Nazwa użytkownika: client@email.com Hasło: Password1!

6 Podsumowanie i wnioski

Stworzona aplikacja pozwoliła na zapoznanie się z technologiami C#, oraz ASP.NET MVC. Przyjęcie jasno określonej architektury ułatwia klarowną implementację systemu. Praca bez testów jednostkowych powoduje powstawanie gorszej jakości kodu, który trudno jest utrzymywać. Refaktoryzacja kodu bez testów jednostkowych jest trudna, najbezpieczniejszym sposobem refaktoryzacji jest tworzenie testów jednostkowych i sprawdzanie, czy po wprowadzeniu zmian zachowanie systemu się nie zmieniło.

Dobrą praktyką jest wyszczególnienie zasad formatowania i tworzenia kodu na początku projektu. Pozwala to na uniknięcie nieścisłości i ułatwia zapewnienie lepszej jakości kodu. Dobrą praktyką jest code review przez innych członków zespołu. Pozwala na zapewnienie czytelności kodu i pozwala wychwycić błędy. Testy jednostkowe powinny być częścią systemu ciągłej integracji od początku pracy nad projektem. Wprowadzenie testów jednostkowych w późniejszej fazie projektu może być bardzo trudne.

W trakcie manualnych testów systemu nie zauważono problemów z wydajnością. Nie zbadano skalowalności systemu, ani nie wykonano testów obciążeniowych systemu. W przypadku pojawienia się problemów z wydajnością proponuje się zastosowanie cacheowania w celu zmniejszenia ilości odwołań do bazy danych.

Trudno jest prowadzić projekt zespołowy, kiedy różni członkowie zespołu posiadają niewspółmierne poziomy motywacji. Powoduje to nierówne rozdzielanie obowiązków, oraz konflikty w zespole, co znacząco utrudnia współpracę. Częsta i klarowna komunikacja w zespole ułatwia organizację pracy i umożliwia uniknięcie problemów wynikających z dezinformacji.

Literatura

- [1] Oficjalna strona ASP.NET MVC
<https://www.asp.net/mvc>
- [2] Oficjalna dokumentacja ASP.NET MVC
https://docs.microsoft.com/plpl/aspnet/#pivot-core&panel-core_overview
- [3] Mosh Hamedani *Should you split your ASP.NET MVC project into multiple projects?*.
<https://programmingwithmosh.com/csharp/should-you-split-your-asp-net-mvc-project-into-multiple-projects/>