WYŻSZA SZKOŁA
INFORMATYKI i ZARZĄDZANIA
z siedzibą w Rzeszowie

KOLEGIUM INFORMATYKI STOSOWANEJ

Kierunek: INFORMATYKA

Dawid Gdowik
Nr albumu studenta 72413

System obsługi biura podróży

Prowadzący: mgr inż. Ewa Żesławska

Dokumentacja projektu

Rzeszów 2026

Spis treści

Rozdział 1

Wstęp i założenia projektu

Przedmiotem projektu jest system obsługi biura podróży, zrealizowany w języku C#. Wybrano formę aplikacji konsolowej, co pozwoliło na skoncentrowanie się na logice biznesowej oraz poprawnej komunikacji z relacyjną bazą danych MySQL. System został zaprojektowany jako narzędzie wspomagające codzienną pracę biura, umożliwiając zarządzanie ofertami wycieczek, ewidencję klientów oraz monitorowanie procesów rezerwacyjnych i finansowych. Wszystkie dane są składowane na serwerze bazy danych, co zapewnia ich trwałość i bezpieczeństwo po zakończeniu sesji programu.

Cele projektu

Głównym celem było opracowanie stabilnego systemu zarządzania danymi o wysokim stopniu powiązania. Wybór tematyki biura podróży pozwolił na praktyczne zastosowanie relacyjnej bazy danych, w której informacje o klientach, wycieczkach i hotelach wzajemnie na siebie oddziałują. Istotnym założeniem było również wykorzystanie paradygmatów programowania obiektowego, takich jak dziedziczenie czy enkapsulacja, w celu zapewnienia przejrzystości i modularności kodu źródłowego.

Wymagania funkcjonalne (WF)

W ramach funkcjonalności systemu zdefiniowano następujące operacje, które zostaną szczegółowo opisane w dalszej części dokumentacji:

- **WF1 Zarządzanie ofertami:** Możliwość dodawania nowych wycieczek, aktualizacji ich parametrów (np. ceny, terminu) oraz usuwania rekordów z bazy.
- **WF2 Ewidencja klientów:** Rejestrowanie danych osób korzystających z usług biura oraz zarządzanie ich profilami.
- **WF3 Obsługa rezerwacji:** Proces łączenia klienta z wybraną ofertą oraz przypisanie pracownika odpowiedzialnego za daną transakcję.
- **WF4 Rozliczenia finansowe:** Rejestrowanie wpłat, monitorowanie statusu płatności i automatyczne wyliczanie pozostałych należności dla każdej rezerwacji.

Wymagania niefunkcjonalne (WN)

Z punktu widzenia technicznego, system spełnia następujące wymagania jakościowe:

- **WN1 Trwałość danych (Persystencja):** Wykorzystanie serwerowego systemu MySQL gwarantuje trwałe przechowywanie informacji.
- **WN2 Walidacja danych:** Implementacja mechanizmów sprawdzających poprawność wprowadzanych informacji (np. blokada ujemnych cen, weryfikacja formatów tekstowych).
- **WN3 Kontrola wersji:** Wykorzystanie systemu Git do zarządzania postępami prac oraz archiwizacja kodu technicznego na platformie GitHub.
- **WN4 Odporność na błędy:** Zastosowanie bloków obsługi wyjątków (try-catch), co zapobiega awaryjnemu zamykaniu aplikacji w przypadku problemów z połączeniem z bazą danych.

Rozdział 2

Opis techniczny i struktura projektu

W tym rozdziale przedstawiono wykorzystane technologie, strukturę bazy danych oraz kluczowe elementy kodu źródłowego aplikacji.

2.1 Specyfikacja technologiczna i narzędzia programistyczne

Proces wytwórczy oprogramowania oparto na nowoczesnym stosie technologicznym, który gwarantuje wysoką wydajność oraz bezpieczeństwo typów.

Język programowania i platforma uruchomieniowa

Jako główny język programowania wykorzystano **C# w wersji .NET 8.0**. Wybór ten podyktowany był dojrzałością platformy oraz wsparciem dla programowania obiektowego i operacji asynchronicznych podczas komunikacji z bazą danych.

Zintegrowane środowisko i system kontroli wersji

Implementację logiki biznesowej przeprowadzono w **Microsoft Visual Studio 2022**. Całość prac monitorowano przy użyciu systemu kontroli wersji **Git**, a kod umieszczono w repozytorium na platformie **GitHub**.

System zarządzania bazą danych

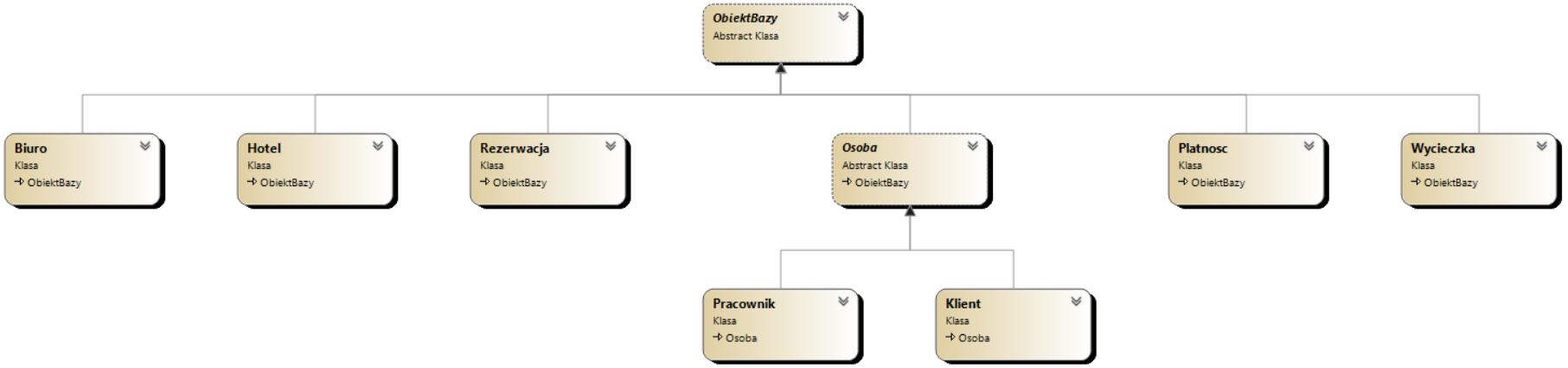
Warstwę trwałości danych oparto na serwerze **MySQL 8.0**. Do fizycznej administracji bazą oraz wykonywania zapytań SQL wykorzystano środowisko **MySQL Workbench**, natomiast schemat logiczny (diagram ERD) na potrzeby dokumentacji opracowano w narzędziu **Draw.io**.

2.2 Architektura systemu i analiza struktury klas

Projekt systemu składa się z klas powiązanych relacjami dziedziczenia oraz implementujących interfejsy dostępowe.

Analiza hierarchii klas i paradygmatu OOP

W systemie wyodrębniono następujące poziomy zależności:



Rysunek 2.1: Szczegółowy diagram klas systemu z uwzględnieniem dziedziczenia i klas abstrakcyjnych.

- **Klasa abstrakcyjna `ObiektBazy`**: Definiuje wspólne właściwości (np. identyfikator) dla wszystkich elementów zapisywanych w bazie danych.
- **Klasa abstrakcyjna `Osoba`**: Służy jako szablon dla wszystkich użytkowników systemu, przechowując dane teleadresowe.
- **Klasy `Klient` oraz `Pracownik`**: Specjalizują klasę nadrzędną `Osoba` o atrybuty specyficzne dla swoich ról.
- **Klasy operacyjne**: `Wycieczka, Hotel, Biuro` oraz `Platnosc` – odpowiadają za konkretne domeny biznesowe.
- **Klasa `Rezerwacja`**: Pełni rolę klasy asocjacyjnej, integrującej obiekty klas `Klient`, `Wycieczka` oraz `Pracownik`.

2.3 Projekt bazy danych (MySQL)

Model danych (diagram ERD) zaprezentowany poniżej został przygotowany w programie **Draw.io**, odwzorowując strukturę zaimplementowaną fizycznie na serwerze MySQL. Cały system składa się z siedmiu tabel, które są ze sobą powiązane logicznie. Każda tabela odpowiada za inny obszar działalności biura podróży – od danych klientów, przez ofertę wycieczek, aż po finanse.

Poniżej znajduje się szczegółowy opis każdej z tabel oraz zawartych w nich kolumn:

1. Tabela `rezerwacje`

Jest to najważniejsza tabela w systemie, znajdująca się w centrum diagramu. Łączy ona wszystkie pozostałe informacje w jedną całość.

- `id_rezerwacji` – unikalny numer każdej rezerwacji (klucz główny).
- `data_rezerwacji` – data, kiedy klient dokonał zakupu wycieczki.
- `id_klienta` – informacja, kto dokonał rezerwacji (klucz obcy do tabeli klientów).
- `id_wycieczki` – informacja, co zostało kupione (klucz obcy do tabeli wycieczek).
- `id_pracownika` – identyfikator osoby z obsługi, która przyjęła zamówienie.

2. Tabela `klienci`

Przechowuje dane osobowe osób korzystających z usług biura.

- `id_klienta` – unikalny identyfikator klienta w bazie.
- `imie` oraz `nazwisko` – podstawowe dane personalne.
- `email` – adres poczty elektronicznej do kontaktu i wysyłki biletów.
- `telefon` – numer telefonu kontaktowego.

3. Tabela `wycieczki`

To katalog dostępnych ofert turystycznych.

- `id_wycieczki` – numer identyfikacyjny oferty.
- `cel` – nazwa miejsca docelowego lub nazwa wycieczki.
- `cena` – koszt wycieczki dla jednej osoby. Użyłem tu typu `DECIMAL(10,2)`, aby dokładnie przechowywać kwoty walutowe.
- `id_hotelu` – przypisanie konkretnego hotelu do danej oferty.

4. Tabela `hotele`

Służy jako baza miejsc noclegowych wykorzystywanych w wycieczkach.

- `id_hotelu` – identyfikator hotelu.
- `nazwa_hotelu` – pełna nazwa obiektu.
- `lokalizacja` – miasto lub region, w którym znajduje się hotel.
- `standard` – liczba gwiazdek lub ocena standardu obiektu.

5. Tabela `pracownicy`

Lista osób zatrudnionych w biurze podróży.

- `id_pracownika` – numer identyfikacyjny pracownika.
- `imie` oraz `nazwisko` – dane pracownika.
- `id_biura` – informacja, w której placówce stacjonarnej pracuje dana osoba.

6. Tabela `biura`

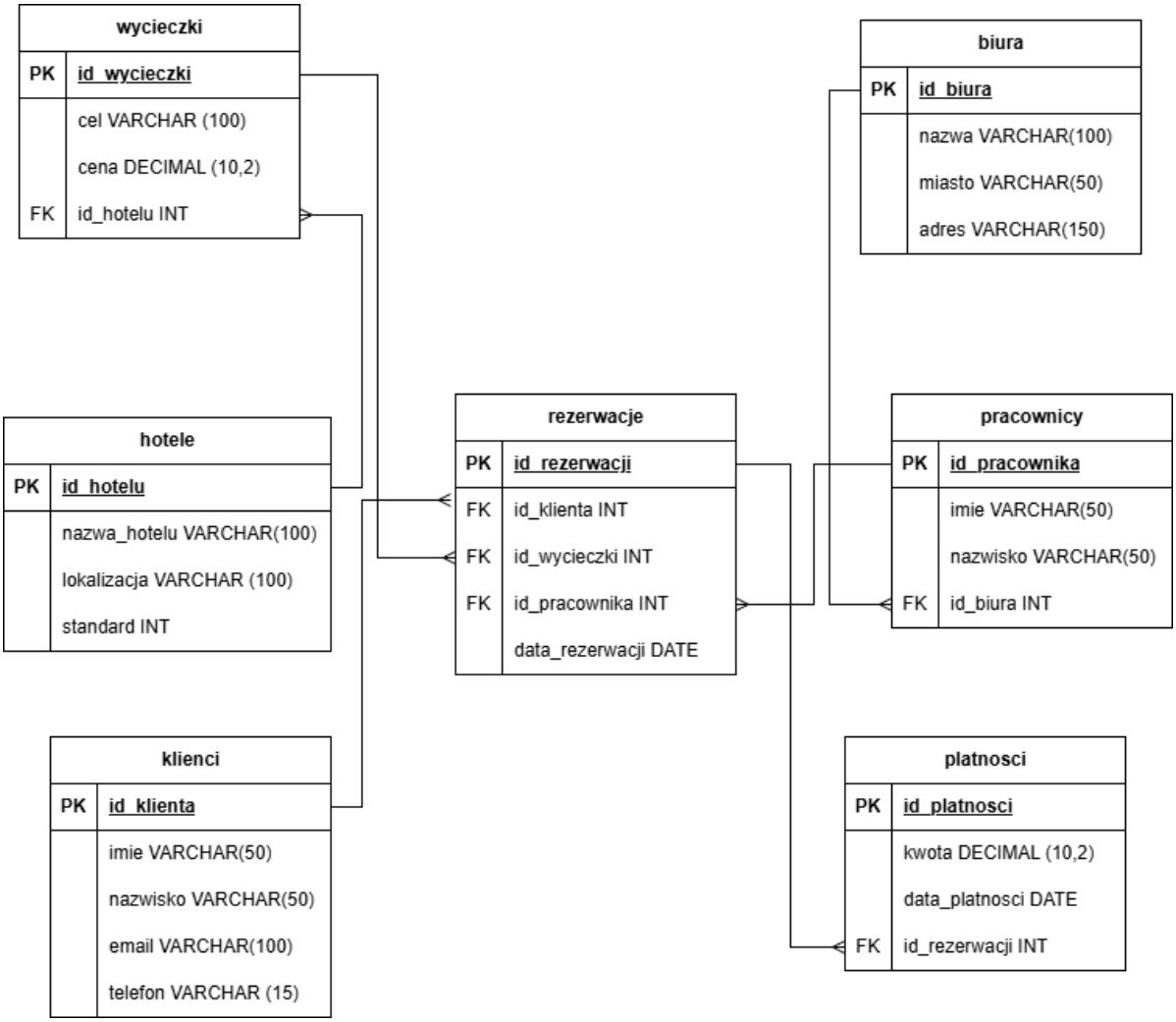
Spis fizycznych placówek biura podróży.

- `id_biura` – numer identyfikacyjny placówki.
- `nazwa` – nazwa własna oddziału.
- `miasto` oraz `adres` – dokładna lokalizacja biura.

7. Tabela `platnosci`

Rejestruje wpłaty dokonywane przez klientów.

- `id_platnosci` – unikalny numer transakcji.
- `kwota` – wpłacona suma pieniędzy (typ `DECIMAL`).
- `data_platnosci` – dzień zaksięgowania wpłaty.
- `id_rezerwacji` – przypisanie płatności do konkretnego zamówienia.



Rysunek 2.2: Fizyczny model danych (ERD) – struktura tabel w systemie MySQL.

2.4 Wymagania systemowe i sprzętowe

W celu zapewnienia stabilnej i wydajnej pracy systemu, środowisko docelowe musi spełniać określone parametry techniczne. Poniżej przedstawiono minimalną konfigurację niezbędną do uruchomienia aplikacji klienckiej oraz nawiązania połączenia z bazą danych.

Wymagania sprzętowe

- **Procesor**: Jednostka wielordzeniowa o architekturze x64, taktowanie minimum 2.0 GHz (zalecane serie Intel Core i5 lub odpowiedniki AMD Ryzen).
- **Pamięć RAM**: Minimum 4 GB (zalecane 8 GB dla zapewnienia płynności działania systemu operacyjnego i bazy danych).
- **Przestrzeń dyskowa**: Minimum 500 MB wolnej przestrzeni na dysku twardym (zalecany dysk SSD) na pliki aplikacji i logi systemowe.
- **Wyświetlacz**: Monitor o rozdzielczości minimalnej 1366x768 pikseli (zalecane Full HD 1920x1080).
- **Urządzenia wejścia**: Klawiatura oraz mysz komputerowa.

Wymagania programowe

- **System operacyjny**: Microsoft Windows 10 (wersja 20H2 lub nowsza) lub Windows 11.
- **Platforma uruchomieniowa**: Zainstalowane środowisko **.NET Desktop Runtime 8.0** (lub nowsze).
- **Serwer bazy danych**: Dostęp do instancji **MySQL Server 8.0** (lokalnie lub sieciowo).
- **Połączenie sieciowe**: Wymagane aktywne połączenie sieciowe w przypadku, gdy baza danych znajduje się na zdalnym serwerze.

Rozdział 3

Prezentacja warstwy użytkowej projektu

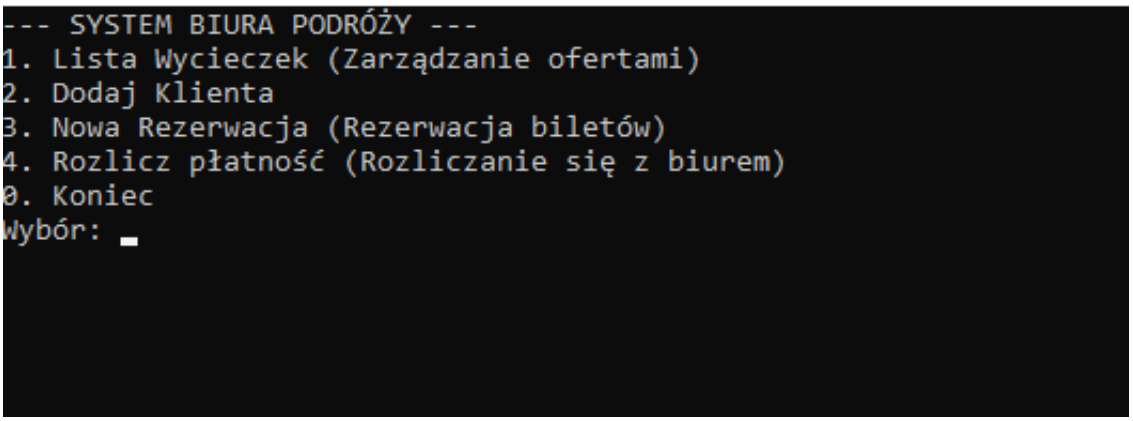
Warstwa wizualna aplikacji stanowi główny kanał komunikacji z systemem. Kluczowym aspektem interfejsu jest jego pełna integracja z silnikiem bazy danych MySQL – przedstawione poniżej widoki nie są jedynie makietami, lecz w pełni zaimplementowanymi funkcjonalnościami operującymi na rzeczywistych rekordach.

3.1 Menu główne i nawigacja

Główny panel sterowania stanowi punkt wyjścia dla wszystkich operacji biznesowych. Nawigacja opiera się na wyborze numerycznym, a każda wybrana opcja inicjuje połączenie z bazą danych i pobranie aktualnych informacji.

Interfejs panelu głównego

Po uruchomieniu programu użytkownik widzi listę dostępnych modułów (1-4). Wybór konkretnej cyfry wywołuje odpowiednią metodę w kodzie C#, która odpowiada za komunikację z konkretną tabelą w bazie danych. System został zabezpieczony przed wyborem nieprawidłowej opcji – w przypadku naciśnięcia klawisza spoza zakresu, mechanizm `default` w instrukcji sterującej poinformuje użytkownika o błędzie i pozwoli na ponowny wybór.



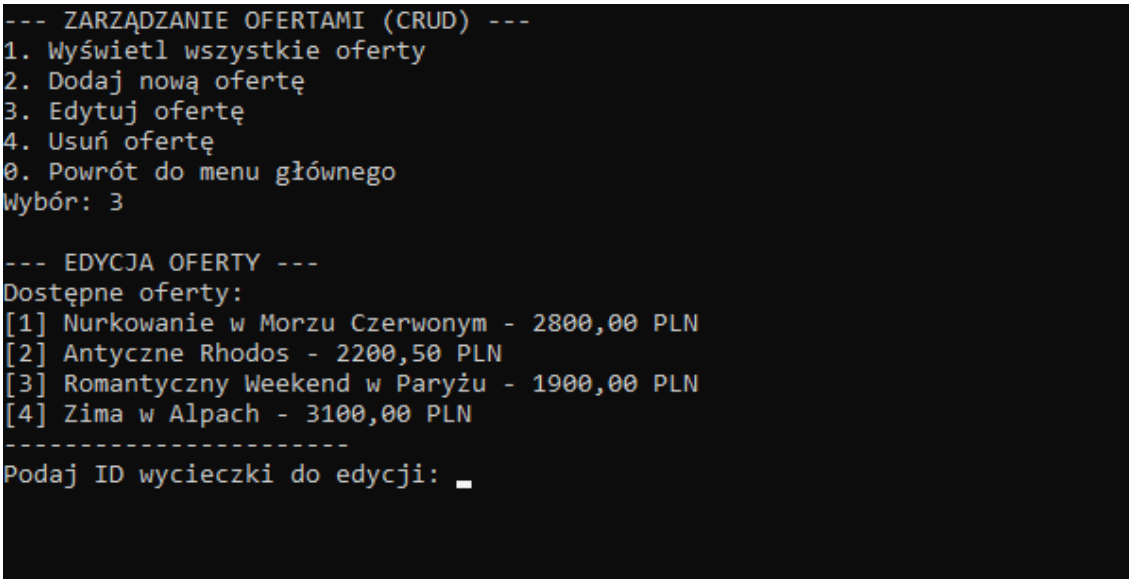
Rysunek 3.1: Główne menu nawigacyjne aplikacji CLI połączonej z bazą MySQL.

3.2 Zarządzanie zasobami biura

W tej sekcji opisano moduły odpowiedzialne za obsługę ofert wycieczkowych oraz bazę osób zarejestrowanych w systemie.

Opcja 1: Zarządzanie ofertami wycieczek (Dane z bazy)

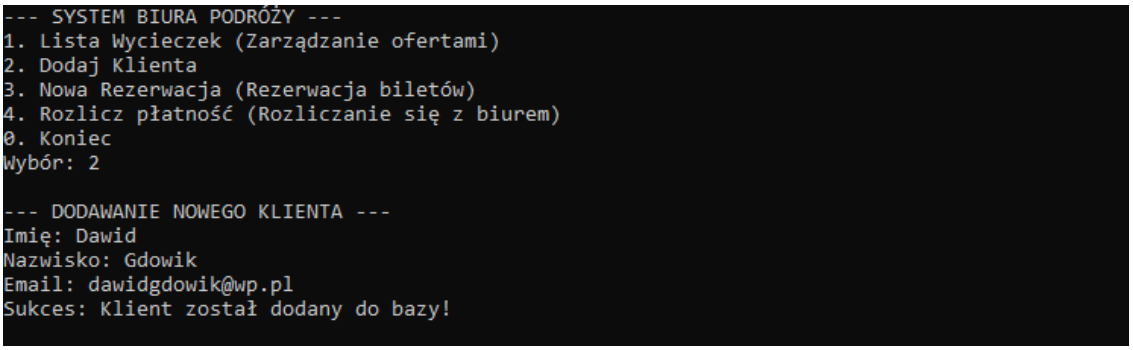
Moduł ten wyświetla listę wycieczek pobieraną bezpośrednio z tabeli `wycieczki`. Widoczne na zrzucie ekranu pozycje, takie jak „Nurkowanie w Morzu Czerwonym", są rekordami zaimplementowanymi w bazie. Funkcje edycji i usuwania natychmiastowo aktualizują stan bazy danych. Dodatkowo, operacja usuwania jest chroniona blokiem `try-catch`, co zapobiega awarii programu w przypadku próby usunięcia wycieczki powiązanej z istniejącą rezerwacją.



Rysunek 3.2: Podmenu zarządzania ofertami z listą wycieczek pobraną z bazy danych.

Opcja 2: Rejestracja nowych klientów

Funkcja ta pozwala na dodanie nowego klienta do tabeli `klienci`. Po wpisaniu danych przez pracownika, aplikacja wysyła zapytanie `INSERT`, a silnik bazy danych automatycznie generuje unikalny numer ID zgodnie z regułą klucza podstawowego. Przed wysłaniem danych do bazy zaimplementowano prostą walidację, która sprawdza, czy kluczowe pola (takie jak imię czy nazwisko) nie są puste, co zapewnia integralność danych.



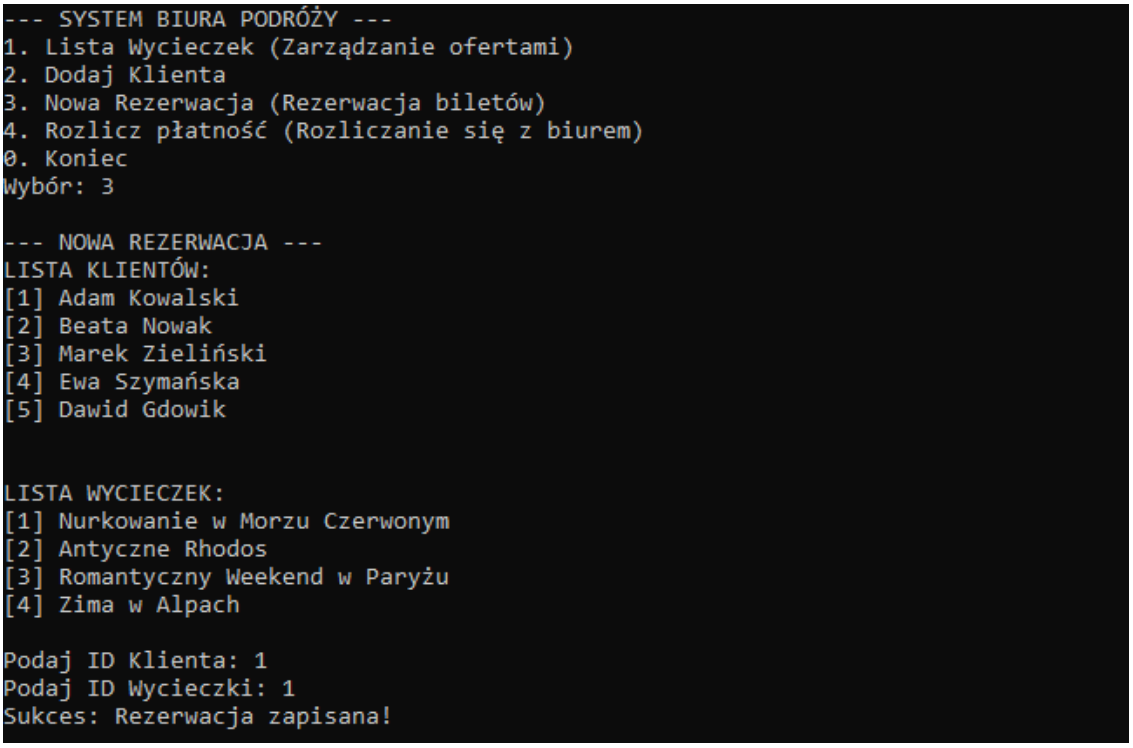
Rysunek 3.3: Proces wprowadzania danych nowego klienta do systemu.

3.3 Procesy rezerwacyjne i finansowe

Ostatni etap prezentacji obejmuje kluczową logikę aplikacji, czyli łączenie klientów z ofertami oraz finalizację płatności.

Opcja 3: Tworzenie nowej rezerwacji

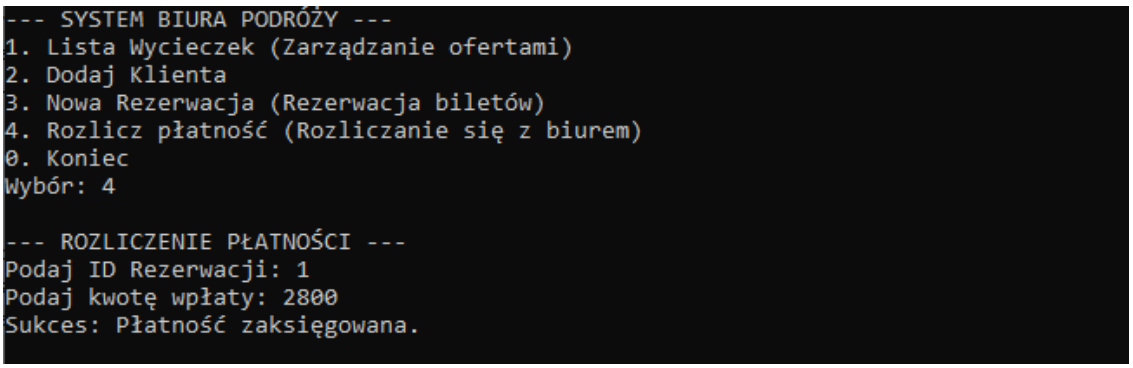
Moduł rezerwacji tworzy powiązania w tabeli asocjacyjnej `rezerwacje`. Wykorzystywane są tutaj dane zaimplementowane w poprzednich krokach (ID klienta i ID wycieczki), co pozwala na zachowanie spójności referencyjnej między wszystkimi elementami systemu. Jeśli użytkownik poda ID, które nie istnieje w bazie (np. ID nieistniejącego hotelu), system przechwyci wyjątek SQL i wyświetli stosowny komunikat, zamiast przerywać działanie aplikacji.



Rysunek 3.4: Interfejs procesu rezerwacji biletów dla klienta.

Opcja 4: Rozliczanie płatności

System finansowy operuje na precyzyjnych kwotach typu `DECIMAL(10,2)`. Moduł płatności umożliwia przypisanie wpłaty do konkretnej rezerwacji, co aktualizuje stan tabeli `platnosci`. Jest to ostateczny etap ścieżki transakcyjnej zaimplementowanej w projekcie. Dzięki zastosowaniu odpowiedniego typu danych, system jest odporny na błędy zaokrągleń typowe dla operacji zmiennoprzecinkowych.



Rysunek 3.5: Okno obsługi płatności za rezerwację.

Podsumowanie pracy z interfejsem

Zastosowanie tekstowego interfejsu użytkownika w pełni integruje się z logiką bazy danych MySQL. Każda zaimplementowana funkcja została przetestowana pod kątem poprawności przesyłu danych oraz odporności na podstawowe błędy użytkownika, co gwarantuje, że system jest stabilnym narzędziem wspomagającym pracę w biurze podróży.

Rozdział 4

Harmonogram realizacji projektu

Proces wytwórczy oprogramowania został precyzyjnie rozplanowany w czasie, co pozwoliło na terminową i rzetelną implementację wszystkich warstw systemu. Realizacja projektu została podzielona na logiczne etapy, począwszy od analizy wymagań, przez projektowanie architektury, aż po implementację i testy.

4.1 Szczegółowy podział zadań

Harmonogram prac został zaprezentowany w dwóch uzupełniających się formach: jako zestawienie tabelaryczne zawierające dokładne daty graniczne oraz jako wizualizacja graficzna na osi czasu. Zarówno tabela z danymi, jak i sam wykres Gantta zostały opracowane w programie Microsoft Excel przy użyciu profesjonalnego szablonu „Elastyczny wykres Gantta", co pozwoliło na precyzyjne odwzorowanie ram czasowych i postępów prac.

Przyjęta strategia planowania opierała się na realistycznym oszacowaniu pracochłonności poszczególnych zadań, co pozwoliło na zachowanie odpowiedniego marginesu czasowego na wypadek wystąpienia nieprzewidzianych trudności technicznych. Systematyczna aktualizacja statusu poszczególnych etapów wewnątrz arkusza kalkulacyjnego zapewniała pełną kontrolę nad przebiegiem procesu wytwórczego i umożliwiła terminowe zakończenie prac nad kluczowymi modułami aplikacji.

Projekt

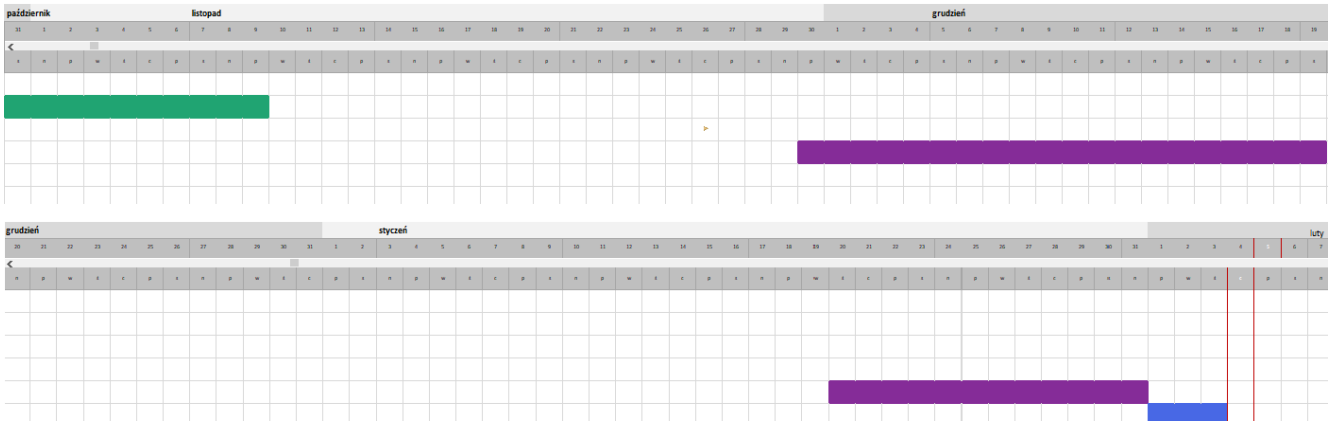
Data rozpoczęcia projektu: 12.10.2025

Przyrost przewijania: 12

Opis punktu kontrolnego	Kategoria	Postęp	Rozpoczęcie	Dni
Analiza tematu i wymagań	Zgodnie z planem	100%	12.10.2025	30
Zgłoszenie tematu i założeń	Punkt kontrolny	100%	27.11.2025	1
Projektowanie i implementacja bazy	Wysokie ryzyko	100%	01.12.2025	20
Implementacja aplikacji (Kod C#)	Wysokie ryzyko	100%	21.01.2026	12
Finalizacja dokumentacji	Średnie ryzyko	100%	02.02.2026	3

Rysunek 4.1: Szczegółowy harmonogram i status realizacji zadań

W zestawieniu tabelarycznym (Rys. 4.1) przypisano stopnie ryzyka do poszczególnych zadań. Etapy takie jak projektowanie bazy danych oraz implementacja kodu oznaczono jako „Wysokie ryzyko", ze względu na ich kluczowe znaczenie dla stabilności całego systemu – ewentualne błędy na tych etapach mogłyby skutkować koniecznością przebudowy znacznej części aplikacji.



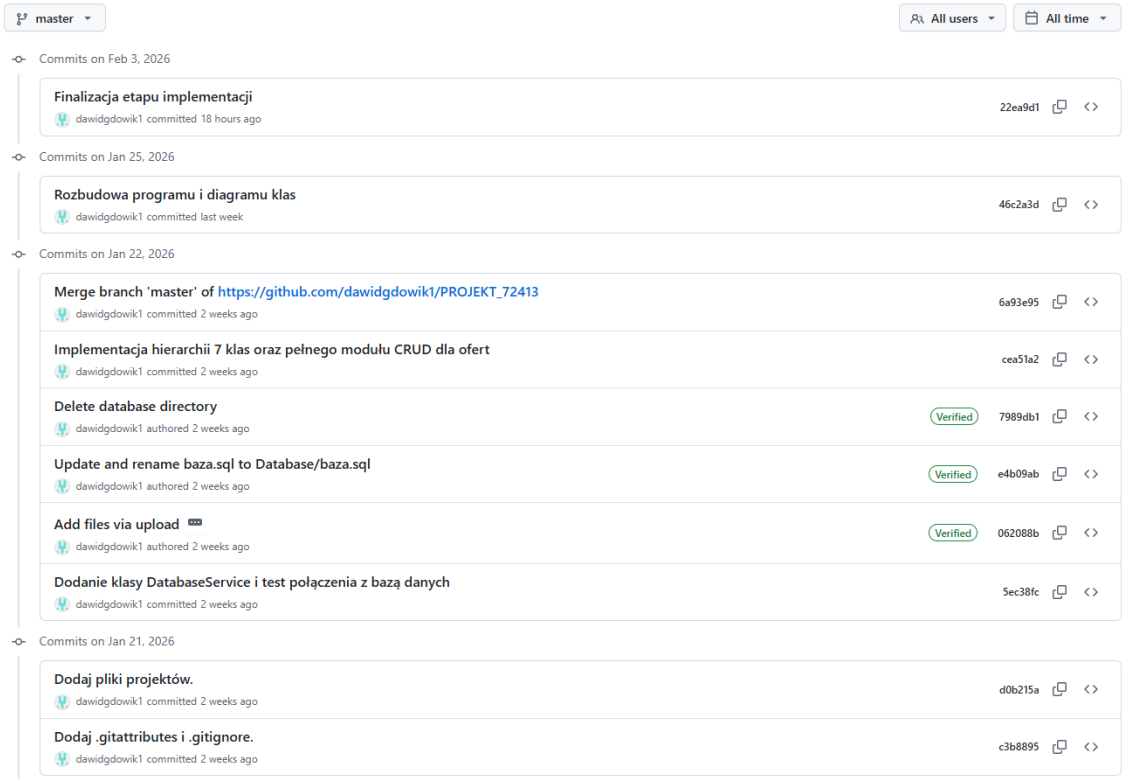
Rysunek 4.2: Harmonogram prac projektowych w postaci wykresu Gantta

Zastosowanie podziału na konkretne etapy pozwoliło uniknąć chaosu podczas pracy. W ramach fazy analitycznej, trwającej 30 dni, opracowano nie tylko wymagania funkcjonalne, ale również stworzono wstępne makiety interfejsu użytkownika oraz diagramy przypadków użycia. Pozwoliło to na uniknięcie błędów logicznych przed przystąpieniem do pisania kodu w języku C#.

Wykres Gantta z kolei pokazuje tzw. ścieżkę projektu: wyraźnie widać, że zakończenie fazy projektowej było warunkiem koniecznym do rozpoczęcia prac programistycznych.

4.2 Weryfikacja realizacji (Git)

Potwierdzeniem realizacji harmonogramu zgodnie z założeniami jest historia zmian w repozytorium kodu źródłowego. Systematyczność prac odzwierciedlają daty poszczególnych zatwierdzeń (commitów), które pokrywają się z zaplanowanymi etapami.



Rysunek 4.3: Historia zmian w repozytorium GitHub potwierdzająca przebieg prac

Jak pokazano na Rysunku 4.3, historia zmian odzwierciedla konkretne kamienie milowe, takie jak implementacja modułu CRUD dla ofert czy konfiguracja połączenia z bazą danych (DatabaseService). Każdy etap widoczny w harmonogramie znajduje swoje bezpośrednie odzwierciedlenie w aktywności w repozytorium. Taki sposób pracy gwarantuje bezpieczeństwo kodu oraz udowadnia, że projekt powstawał sukcesywnie i systematycznie.

W repozytorium udostępniono kompletny zestaw plików projektowych, obejmujący kod źródłowy aplikacji (C#), skrypty SQL definiujące strukturę i dane początkowe bazy danych oraz pliki źródłowe niniejszej dokumentacji przygotowanej w systemie LaTeX. Adres repozytorium:

`https://github.com/dawidgdowik1/PROJEKT_72413.git` [dostęp: 04.02.2026]

Rozdział 5

Podsumowanie

Celem zrealizowanego projektu było zaprojektowanie i implementacja systemu informatycznego wspomagającego kluczowe procesy w biurze podróży. Zakres zrealizowanych prac objął analizę wymagań funkcjonalnych, zaprojektowanie relacyjnej bazy danych oraz stworzenie aplikacji umożliwiającej zarządzanie ofertą turystyczną i bazą klientów.

Aplikacja została wykonana zgodnie z przyjętym harmonogramem. Główny nacisk położono na poprawność logiczną operacji biznesowych (np. rezerwacja wycieczki) oraz spójność danych. System pozwala na efektywną ewidencję klientów oraz obsługę procesu sprzedaży, co stanowi realizację głównego celu projektowego.

Planowane dalsze prace rozwojowe

Projekt posiada architekturę otwartą na modyfikacje. W ramach dalszego rozwoju systemu zaplanowano następujące rozszerzenia:

- **Moduł dostępu dla klienta (Web)** – stworzenie strony internetowej zintegrowanej z tą samą bazą danych, umożliwiającej klientom samodzielne przeglądanie ofert i dokonywanie wstępnych rezerwacji z domu.
- **Integracja z bramką płatności** – automatyzacja procesu księgowania wpłat poprzez podłączenie zewnętrznego dostawcy płatności (np. BLIK, PayU).
- **System powiadomień i marketingu** – automatyczne wysyłanie e-maili lub SMS-ów z przypomnieniem o terminie zapłaty lub informacją o nowych ofertach promocyjnych ("Last Minute") dopasowanych do preferencji klienta.
- **Rozbudowa modułu raportowania** – implementacja zaawansowanych wykresów statystycznych dla menedżera biura, obrazujących trendy sprzedaży w poszczególnych sezonach.

Stworzone oprogramowanie stanowi solidny fundament pod wyżej wymienione funkcjonalności.

Bibliografia

[1] E. Żesławska, *Materiały do zajęć z przedmiotu Programowanie Obiektowe*, WSIiZ Rzeszów.

[2] B. Fryc, *Programowanie Obiektowe – materiały wykładowe*, WSIiZ Rzeszów.

[3] Matulewski J., *C#: lekcje programowania: praktyczna nauka programowania dla platform .NET i .NET Core*, Helion, Gliwice 2021 lub nowsze.

[4] W3Schools, *C# Tutorial*, `https://www.w3schools.com/cs/` [dostęp: 04.02.2026].

Spis rysunków