

# System Zarządzania Rezerwacjami Sal

## Technologia

- Język: C#
- UI: WPF
- Baza danych: MariaDB/MySQL
- Backend: Node.js (REST API)
- IDE: Visual Studio
- Kontrola wersji: Git (repo na GitHub)

## Funkcjonalności

### 1. Logowanie i rejestracja

- Rejestracja użytkownika (klient/admin). ✓
- Logowanie (rola decyduje o widoku funkcji). ✓
- Hasła przechowywane w formie zaszyfrowanej (np. hash). ✓

### 2. Zarządzanie salami (admin)

- Lista statych sal do wyboru – np.:
  - Sala A (ul. Piłsudskiego 5, Wrocław)
  - Sala B (ul. Jana Pawła II 10, Warszawa)
  - Sala C (ul. Długa 15, Kraków)
- Admin może edytować te dane (nazwa, adres).

### 3. Tworzenie rezerwacji (klient)

- Formularz rezerwacji:
  - Wybór sali z listy.
  - Data i godzina.
- Po wyborze sali aplikacja automatycznie pobiera współrzędne tej sali z Google Maps API.
- Możliwość podania adresu domowego użytkownika (start podróży).
- Wyświetlenie mapki z trasą dojazdu do sali – Google Maps Static API lub Embedded API.

### 4. Przegląd rezerwacji

- Klient widzi swoje rezerwacje.

- Admin widzi wszystkie rezerwacje, filtrowane np. po sali, dacie, użytkowniku.

## 5. Graficzna prezentacja (dla admina)

- Wykres: liczba rezerwacji w miesiącu.
- Statystyki: najczęściej wybierane sale, dni tygodnia itp.

## 6. Dane w bazie (MariaDB/MySQL)

- Tabela Użytkownicy: ID, Imię, Email, Hasło, Rola. ✓
- Tabela Sale: ID, Nazwa, Adres, Współrzędne (Latitude, Longitude). ✓
- Tabela Rezerwacje: ID, Użytkownik, Sala, Data, Godzina, Status. ✓

## 7. Integracja z Google Maps API

- Po zapisaniu rezerwacji aplikacja pobiera:
  - Współrzędne sali (Geocoding API).
  - Mapę statyczną z zaznaczoną salą (Static Maps API) lub interaktywną mapę (Embedded Maps API).
  - Opcjonalnie: szacowany czas dojazdu (Directions API).
- Przykład scenariusza:
  1. Klient wybiera „Sala A – Wrocław”.
  2. System pobiera współrzędne tej sali z Google Maps.
  3. Klient może opcjonalnie wpisać swój adres domowy.
  4. System wyświetla mapę z trasą z domu do sali.
  5. Trasa zapisywana jako część rezerwacji.

## 8. Walidacja danych

- Pola nie mogą być puste.
- Data w przyszłości.
- Adres poprawny (sprawdzany przez Google Maps Geocoding API).
- E-mail poprawny.

## 9. Dokumentacja i testy

- Dokumentacja XML + np. DocFX.
- Testy jednostkowe dla walidacji i operacji na bazie.

## 10. Persystencja danych

- Wszystkie dane w bazie MariaDB/MySQL
- Po ponownym uruchomieniu dane są dalej widoczne.

**API – jakiego użyć?**

- Geocoding API – zamiana adresu na współrzędne.
- Static Maps API – prosta mapka do wyświetlenia w aplikacji (obrazek z zaznaczoną lokalizacją).
- Directions API (opcjonalnie) – obliczenie trasy i czasu dojazdu.
- Maps Embed API (jeśli chcesz klikającą mapę w oknie WPF).