

POLITECHNIKA CZĘSTOCHOWSKA
WYDZIAŁ INŻYNIERII MECHANICZNEJ I INFORMATYKI



PROJEKT ZESPOŁOWY
Aplikacja „Cmentarz komunalny”

Członkowie zespołu:

Mariusz Jędrzejewski	Nr albumu: 128059 , Kierownik, Back-end Angular 8 i .NET Core 3.1, C#, API, produkcja oraz rozwijanie aplikacji, organizacja zadań, dokumentacja, Tester aplikacji
Kamil Grodzki	Nr albumu: 127986 , Front-end Angular 8, User Interface oraz User Experience, szata graficzna aplikacji, Tester aplikacji
Piotr Bednarek	Nr albumu: 127919 , UML, drobne poprawki
Wiktor Powązka	Nr albumu: 126259 , Praca przy aplikacji, bazy danych T-SQL, integralność oraz poprawność bazy danych, Tester aplikacji

Kierunek: Informatyka

Studia: stacjonarne

Poziom studiów: pierwszego stopnia

Opis projektu

Opis realizowanego projektu:

Projekt miał na celu zrealizowanie aplikacji cmentarza komunalnego. Aplikacja umożliwia użytkownikom odnalezienie informacji na temat zmarłych ludzi spoczywających na tym cmentarzu komunalnym, jak i odnalezienie dokładnej lokalizacji spoczynku.

Przyjęte założenia projektowe i zakładana funkcjonalność:

Dla użytkowników:

- Możliwość podstawowej obsługi aplikacji
- Możliwość przeglądania aktualności na stronie cmentarza komunalnego
- Możliwość przeglądania aktualnych nekrologów na stronie cmentarza komunalnego
- Możliwość przeglądania informacji dotyczących strony oraz cmentarza komunalnego
- Możliwość zobaczenia mapy cmentarza komunalnego
- Możliwość wyszukania zmarłej osoby wpisując jego dane
- Możliwość wyświetlenia informacji dotyczących kontaktu z obsługą cmentarza

Dla pracowników:

- Wszystkie możliwości które posiada domyślny użytkownik aplikacji
- Możliwość dodawania aktualności / nekrologów / zmarłych osób do bazy danych
- Możliwość edycji istniejących w bazie aktualności / nekrologów / zmarłych osób

Dla administracji:

- Wszystkie możliwości który posiada pracownik oraz domyślny użytkownik aplikacji
- Możliwość usuwania aktualności / nekrologów / zmarłych / kwater z bazy danych
- Możliwość tworzenia nowych pracowników lub administratorów
- Możliwość zablokowania pracowników w ich możliwościach
- Możliwość wyświetlenia wszystkich użytkowników zarejestrowanych w systemie

Scenariusze kluczowych czynności

Dostępność dla: Administrator / Pracownik / Użytkownik

Przeglądanie rocznic / aktualności / nekrologów / informacji / wyszukiwarki zmarłych / mapy:

Ścieżka: ~/[rocznice / aktualnosci / nekrologi / informacje / wyszukiwarka-grobow / mapa / kontakt]

- Wybranie za pomocą myszki wybranej zakładki w menu nawigacyjnym
- Przeglądanie informacji

Wyszukiwanie zmarłych:

Ścieżka: ~/wyszukiwarka-grobow

- Podanie danych zmarłego (np. po nazwisku lub dacie śmierci)
- Ukazane zostaną wszystkie osoby zmarłe spełniające wyżej wprowadzone warunki

Wyszukanie informacji na temat wybranej zmarłej osoby:

- Wybranie za pomocą myszki osoby zmarłej z listy, będzie ona zaznaczona
- Użycie przycisku „Szukaj”, poniżej w panelu „Układ sektora” można wyświetlić mapę z podziałem na kwatery w danym sektorze, obok wypisana będzie informacja w jakim sektorze się znajduje oraz jaki jest numer kwatery wybranej zmarłej osoby

Dostępność dla: Administrator / Pracownik

Logowanie w aplikacji:

Ścieżka: ~/Account/Login

- Wprowadzenie adresu email oraz hasła
- Użycie przycisku „Zaloguj” by zalogować się jako administrator / pracownik w aplikacji

Wyświetlenie strony głównej zarządzania:

Ścieżka: ~/Home

Dodanie aktualności:

Ścieżka: ~/panel[administratora / pracownika]

- Wprowadzenie tytułu aktualności (Title)
- Wprowadzenie treści aktualności (NewsContent)
- Wybranie daty za pomocą Date Picker’a (DateOfPublication)
- Użycie przycisku „Dodaj” do zamieszczenia aktualności w bazie danych metodą *POST*

Dodanie nekrologu:

Ścieżka: ~/panel[administratora / pracownika]

- Wprowadzenie imienia i nazwiska zmarłej osoby (Name)
- Wprowadzenie treści nekrologu (ObituaryContent)
- Wybranie daty za pomocą Date Picker’a (DateOfPublication_Obituary)
- Użycie przycisku „Dodaj” do zamieszczenia nekrologu w bazie danych

Uwaga: Nekrologi nie są powiązane z bazą zmarłych osób ze względu na to, że nie we wszystkich przypadkach ludzie bliscy zmarłym osobom chcą by znajdowali się na stronie.

Dodanie zmarłego:

Ścieżka: ~/panel[administratora / pracownika]

- Wprowadzenie imienia i nazwiska (Name)
- Wprowadzenie numeru grobu / kwatery (LodgingId)

- Wybranie daty urodzenia (lub bez daty urodzenia, jeśli nie jest znana) za pomocą Date Picker'a (DateOfBirth)
- Wybranie daty śmierci za pomocą Date Picker'a (DateOfDeath)
- Użycie przycisku „Dodaj” do zamieszczenia zmarłej osoby w bazie danych metodą *POST*

Edycja aktualności:

Ścieżka: ~/panel[administratora / pracownika]

Uwaga: W zależności od tego co chcemy edytować, wprowadzamy tylko te wartości. Nie ma potrzeby wprowadzania wszystkiego od zera.

- Wybranie za pomocą myszki aktualności do edycji
- Wprowadzenie danych które będą edytowane (Title, NewsContent, DateOfPublication)
- Użycie przycisku „Edytuj” do edycji bieżącej aktualności w bazie danych metodą *PUT*

Edycja nekrologu:

Ścieżka: ~/panel[administratora / pracownika]

Uwaga: W zależności od tego co chcemy edytować, wprowadzamy tylko te wartości. Nie ma potrzeby wprowadzania wszystkiego od zera.

- Wybranie za pomocą myszki nekrologu do edycji
- Wprowadzenie danych które będą edytowane (Name, ObituaryContent, DateOfPublication_Obituary)
- Użycie przycisku „Edytuj” do edycji bieżącego nekrologu w bazie danych metodą *PUT*

Edycja zmarłej osoby:

Ścieżka: ~/panel[administratora / pracownika]

Uwaga: W zależności od tego co chcemy edytować, wprowadzamy tylko te wartości. Nie ma potrzeby wprowadzania wszystkiego od zera.

- Wybranie za pomocą myszki zmarłej osoby do edycji
- Wprowadzenie danych które będą edytowane (Name, LodgingId, DateOfBirth, DateOfDeath)
- Użycie przycisku „Edytuj” do edycji bieżącej zmarłej osoby w bazie danych metodą *PUT*

Dostępność dla: Administrator

Usunięcie aktualności / nekrologu / zmarłego:

Ścieżka: ~/paneladministratora

- Wybranie za pomocą myszki rekordu z bazy który chcemy usunąć z wybranej tabeli
- Użycie przycisku „Usuń” w celu usunięcia danego rekordu z bazy danych metodą *DELETE*

Dodanie pracownika:

Ścieżka: ~/Admin/RegisterEmployee

- Podanie adresu email pracownika (Email / UserName)
- Podanie hasła pracownika
- Podanie hasła pracownika ponownie w celu potwierdzenia
- Stworzenie użytkownika o roli pracownika (RoleName: „Employee”)

Warunki: Hasło musi mieć przynajmniej 6 lub maksymalnie 100 znaków, jeden znak musi być cyfrą.

Dodanie administratora:

Ścieżka: ~/Admin/RegisterAdmin

- Podanie adresu email administratora (Email / UserName)

- Podanie hasła administratora
- Podanie hasła administratora ponownie w celu potwierdzenia
- Stworzenie użytkownika o roli administratora (RoleName: „Administrator”)

Warunki: Hasło musi mieć przynajmniej 6 lub maksymalnie 100 znaków, jeden znak musi być cyfrą.

Zablokowanie pracownika:

Ścieżka: ~/Admin/BlockEmployee

- Wybranie za pomocą selecta pracownika/ów do zablokowania
- Użycie przycisku „Zablokuj” który metodą *POST* zmieni rolę pracownika na zablokowanego pracownika w tabeli *AspNetUserRoles* („Employee” -> „BlockedEmployee”)
- Jeśli decyzją nie będzie blokowany pracownik, można cofnąć się przy pomocy przycisku „Powrót”, odprowadzi nas na stronę główną administratora (~/Admin)

Odblokowanie pracownika:

Ścieżka: ~/Admin/BlockEmployee

- Wybranie za pomocą selecta pracownika/ów do odblokowania
- Użycie przycisku „Odblokuj” który metodą *POST* zmieni rolę pracownika na zablokowanego pracownika w tabeli *AspNetUserRoles* („BlockedEmployee” -> „Employee”)
- Jeśli decyzją nie będzie odblokowany pracownik, można cofnąć się przy pomocy przycisku „Powrót”, odprowadzi nas na stronę główną administratora (~/Admin)

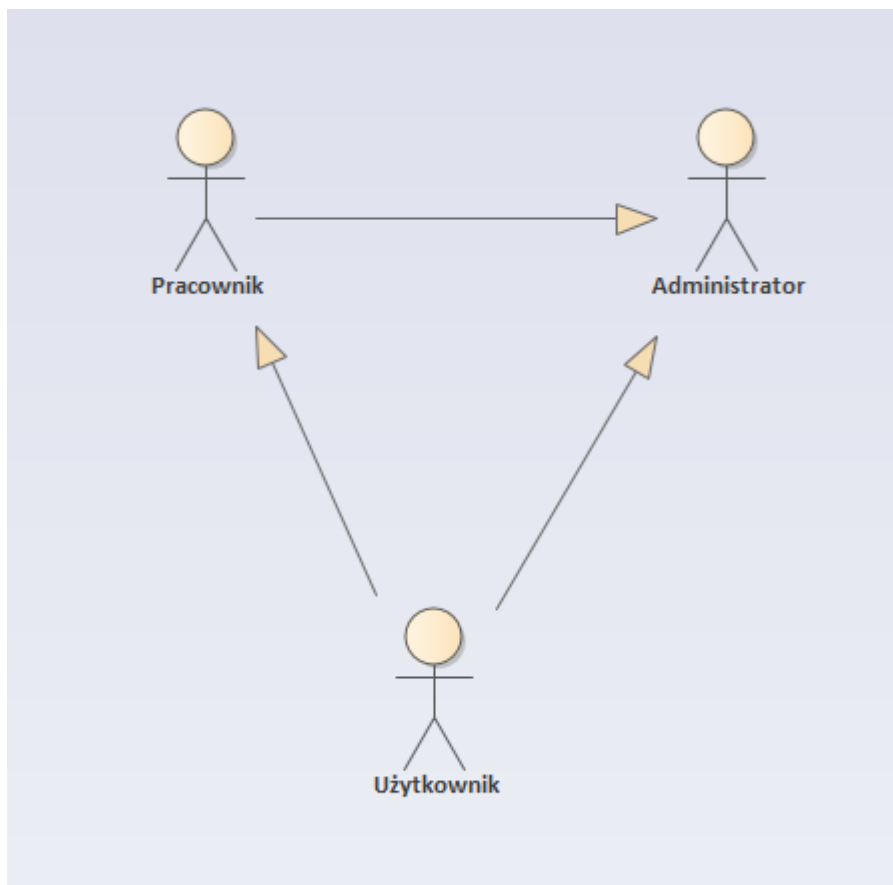
Wyświetlenie wszystkich zarejestrowanych kont:

Ścieżka: ~/Admin/ListUsers

- Wyświetlenie i przejrzenie wszystkich kont zarejestrowanych w serwisie, można wyświetlić tu informacje takie jak UserId, Email oraz UserName (domyślnie UserName jest równy Email)

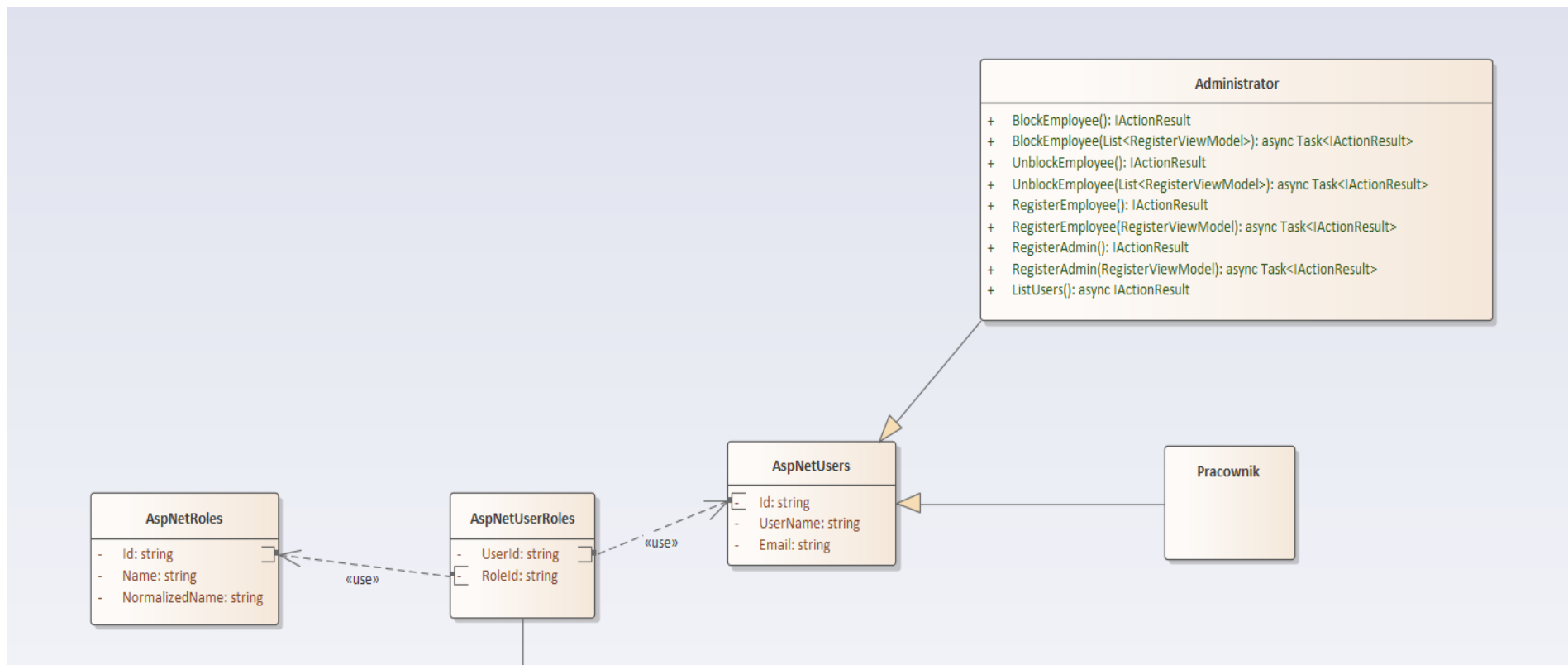
Diagramy UML

Diagram aktorów:

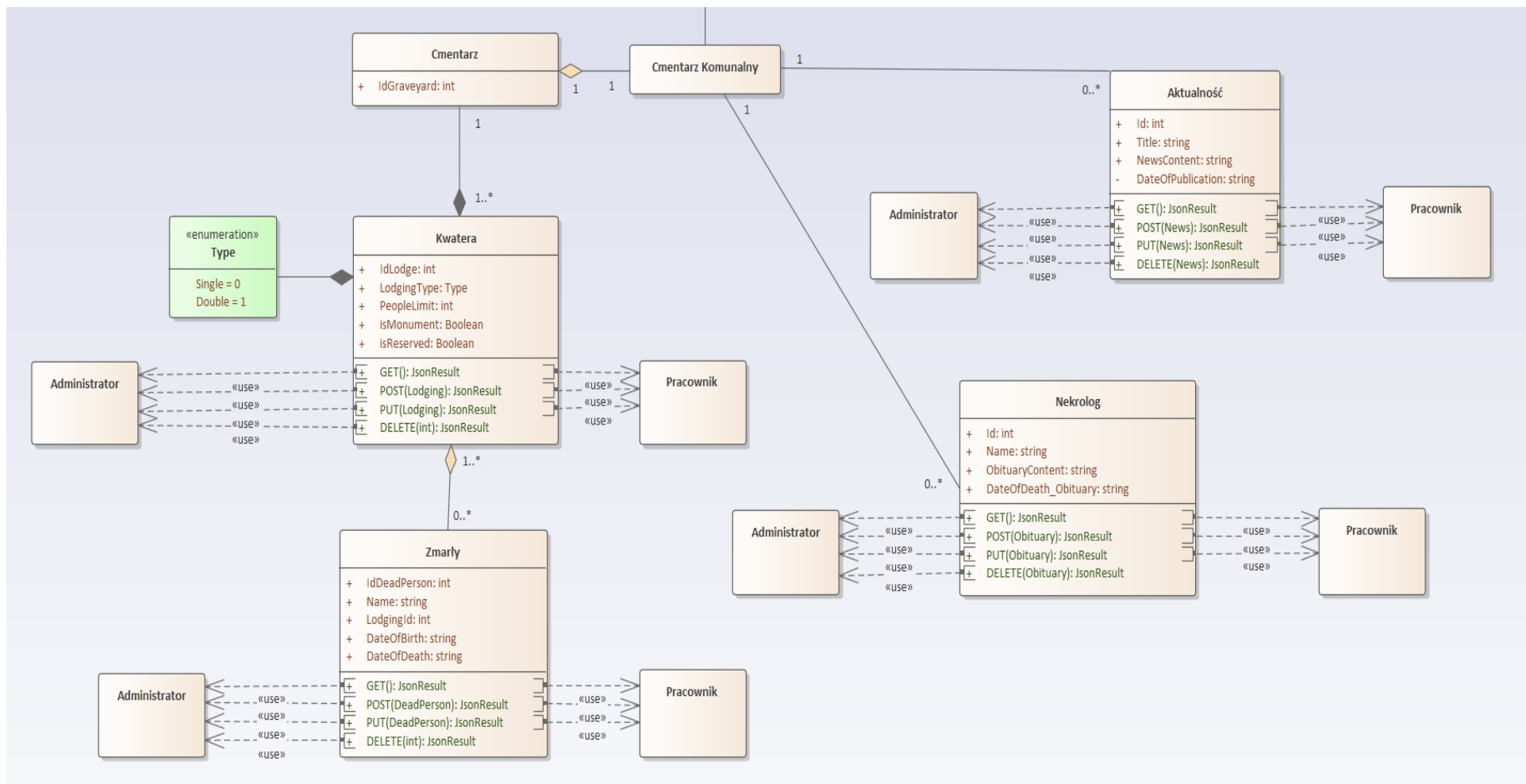


Użytkownik to domyślny aktor korzystający z aplikacji cmentarza. **Pracownik** po nim dziedziczy jego funkcjonalności. **Administrator** dziedziczy po pracowniku oraz po użytkowniku.

Diagram klas:



AspNetUserRoles połączone z CmentarzKomunalny, bo to jeden system.



Klasy stworzone przez ApplicationDbContext oraz system Identity w EF Core:

AspNetUsers:

Pola:

- [string] **Id:** identyfikator użytkownika
 - [string] **UserName:** nazwa użytkownika, jest równoważna w systemie z Email
 - [string] **Email:** email użytkownika, jest równoważny w systemie z UserName
- ... Reszta pól nie jest wymagana dla kontekstu diagramu klas.

AspNetRoles:

Pola:

- [string] **Id:** identyfikator roli
 - [string] **Name:** nazwa roli, np. Admin
 - [string] **NormalizedName:** nazwa znormalizowana danej roli, automatycznie, np. ADMIN
- ... Reszta pól nie jest wymagana dla kontekstu diagramu klas.

AspNetUserRoles:

Pola:

- [string] **UserId:** identyfikator użytkownika, połączony z Id AspNetUsers
- [string] **RoleId:** identyfikator roli, połączony z Id AspNetRoles

Klasy stworzone przez CmentarzContext:

Aktualność: [News]

Pola:

- [int] **Id:** identyfikator danej aktualności
- [string] **Title:** tytuł aktualności
- [string] **NewsContent:** zawartość aktualności
- [string] **DateOfPublication:** data zamieszczenia aktualności w aplikacji

Metody:

- [JsonResult] **GET():** pobranie danych dot. aktualności z bazy danych do wyświetlania w serwisie
- [JsonResult] **POST(News news):** przesłanie nowego rekordu do bazy danych typu Aktualność [Dostęp: Administrator / Pracownik]
- [JsonResult] **PUT(News news):** metoda służąca do aktualizowania danych typu Aktualność w bazie danych [Dostęp: Administrator / Pracownik]
- [JsonResult] **DELETE(int id):** metoda służąca usuwaniu rekordu z bazy danych typu Aktualność bazując na identyfikatorze [Dostęp: Administrator]

Cmentarz: [Graveyard]

Pola:

- [int] **IdGraveyard:** identyfikator cmentarza

Nekrolog: [Obituary]

Pola:

- [int] **Id:** identyfikator danego nekrologu
- [string] **Name:** imię i nazwisko zmarłej osoby w nekrologu
- [string] **ObituaryContent:** treść zamieszczonego nekrologu
- [string] **DateOfDeath_Obituary:** data śmierci zmarłej osoby zamieszczonej w nekrologu

Metody:

- [JsonResult] **GET():** pobranie danych dot. nekrologów z bazy danych do wyświetlania w serwisie
- [JsonResult] **POST(Obituary obituary):** przesłanie nowego rekordu do bazy danych typu Nekrolog [Dostęp: Administrator / Pracownik]
- [JsonResult] **PUT(Obituary obituary):** metoda służąca do aktualizowania danych typu Nekrolog w bazie danych [Dostęp: Administrator / Pracownik]
- [JsonResult] **DELETE(int id):** metoda służąca usuwaniu rekordu z bazy danych typu Nekrolog bazując na identyfikatorze [Dostęp: Administrator]

Kwatera: [Lodging]

Pola:

- [int] **IdLodge:** identyfikator kwatery, połączony kluczem z LodgingId w tabeli **Zmarły**
- [Type] **LodgingType:** typ kwatery, pojedyncza (*Single*) lub podwójna (*Double*), decyduje to o maksymalnej zmarłych ludzi których można w niej pochować (pojedyncza: 3, podwójna: 6), jeśli jest to kolumbarium lub inny typ, ustawić domyślnie na pojedynczą, a limit osób ustalić odgórnie w **PeopleLimit**
- [int] **PeopleLimit:** limit zmarłych osób które mogą znajdować się w danej kwaterze
- [bool] **IsMonument:** jeśli *true*, to jest grób murowany i może on być grobem podwójnym, jeśli *false*, to jest to grób zwyczajny, dzięki temu można po upływie 20 lat pochować następną osobę
- [bool] **IsReserved:** jeśli *true*, to jest to grób zarezerwowany, jeśli *false*, nie jest zarezerwowany przez nikogo

Metody:

- [JsonResult] **GET():** pobranie danych dot. kwater z bazy danych do wyświetlania w serwisie
- [JsonResult] **POST(Lodging lodging):** przesłanie nowego rekordu do bazy danych typu Kwatera [Dostęp: Administrator / Pracownik]
- [JsonResult] **PUT(Lodging lodging):** metoda służąca do aktualizowania danych typu Kwatera w bazie danych [Dostęp: Administrator / Pracownik]
- [JsonResult] **DELETE(int id):** metoda służąca usuwaniu rekordu z bazy danych typu Kwatera bazując na identyfikatorze [Dostęp: Administrator]

Zmarły: [DeadPerson]

Pola:

- [int] **IdDeadPerson:** identyfikator martwej osoby w systemie
- [string] **Name:** imię i nazwisko martwej osoby
- [int] **LodgingId:** identyfikator kwatery w której się znajduje martwa osoba, jest powiązana kluczem z tabelą **Kwatera [IdLodge]**.

- [string] **DateOfBirth**: data urodzenia zmarłej osoby, podanie jej nie jest wymagane jeśli nie jest znana
- [string] **DateOfDeath**: data śmierci zmarłej osoby

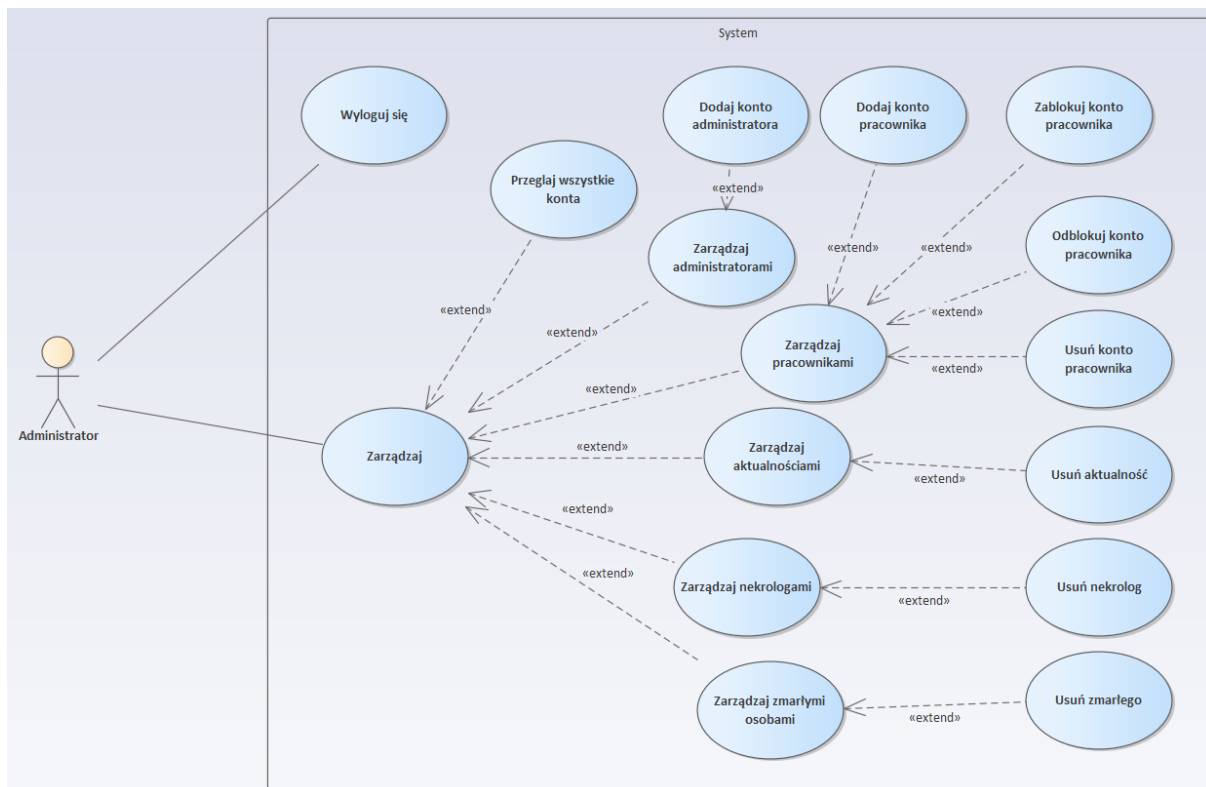
Metody:

- [JsonResult] **GET()**: pobranie danych dot. aktualności z bazy danych do wyświetlania w serwisie
- [JsonResult] **POST(DeadPerson deadperson)**: przesłanie nowego rekordu do bazy danych typu Zmarły [Dostęp: Administrator / Pracownik]
- [JsonResult] **PUT(DeadPerson deadperson)**: metoda służąca do aktualizowania danych typu Zmarły w bazie danych [Dostęp: Administrator / Pracownik]
- [JsonResult] **DELETE(int id)**: metoda służąca usuwaniu rekordu z bazy danych typu Zmarły bazując na identyfikatorze [Dostęp: Administrator]

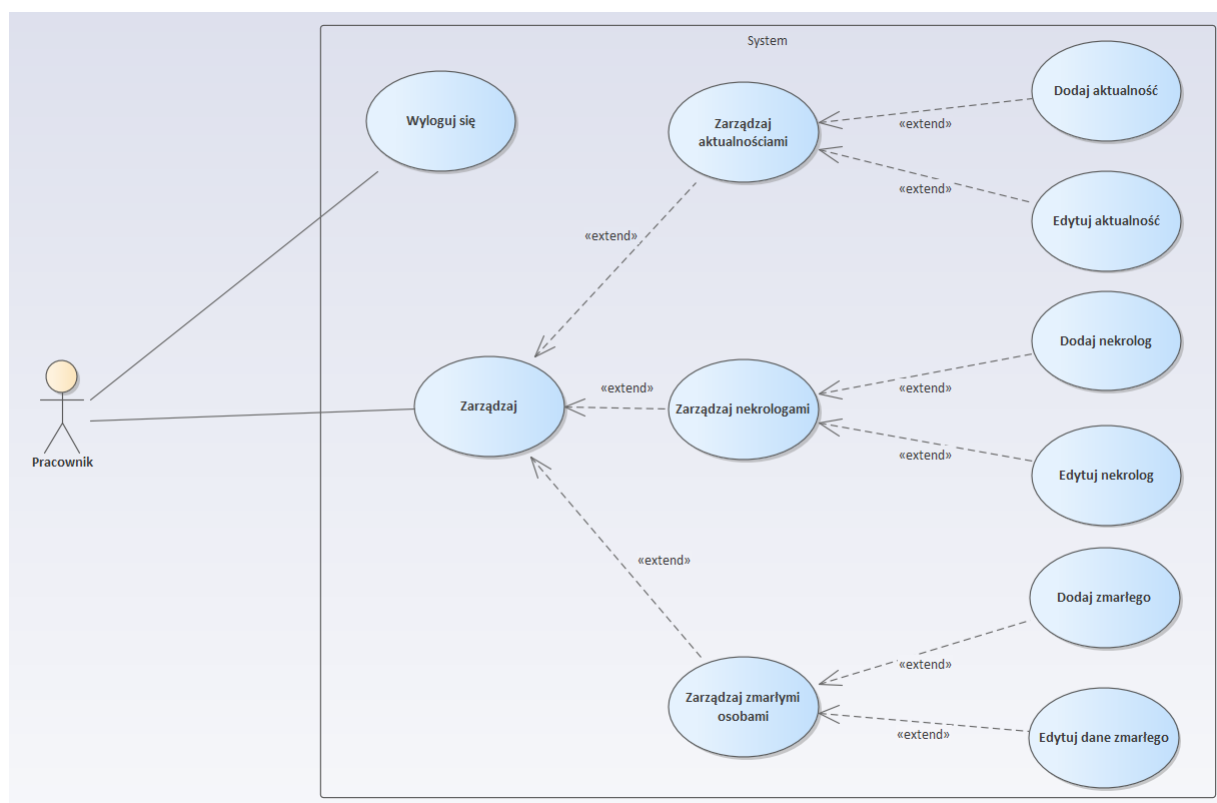
Diagramy przypadków użycia:

Opis poszczególnych procesów znajduje się w scenariuszach kluczowych czynności

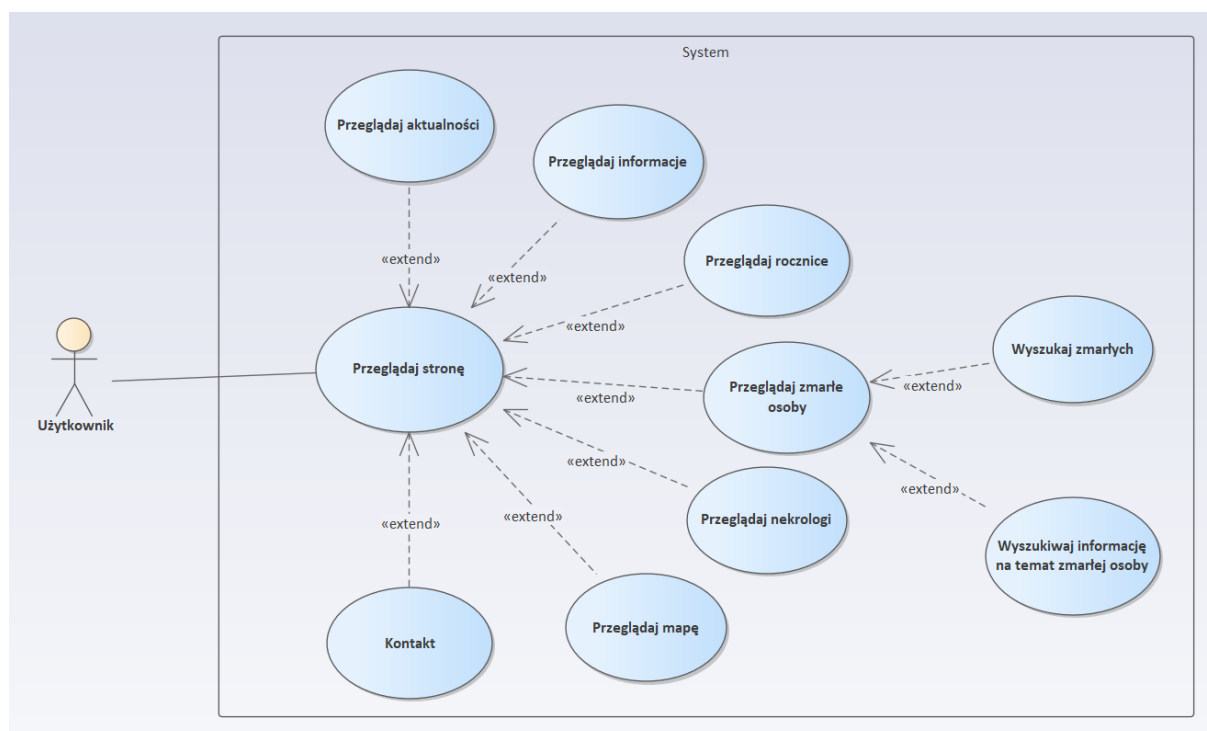
Administrator:



Pracownik:

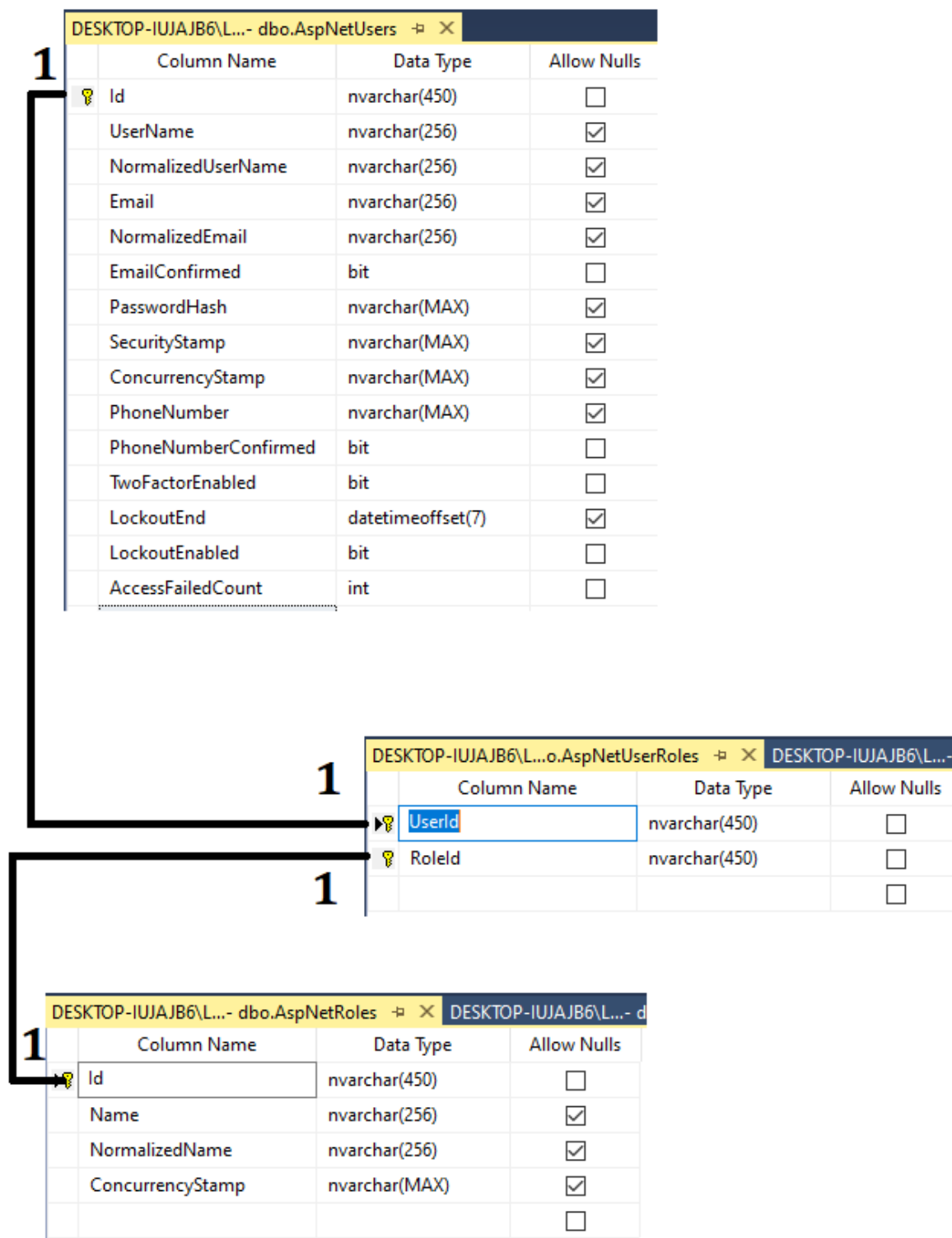


Użytkownik:



Struktura bazy danych

AspNetUsers / *AspNetRoles* / *AspNetUserRoles* – zarejestrowani użytkownicy w serwisie na podstawie **Microsoft.AspNetCore.Identity.EntityFrameworkCore** oraz **Microsoft.AspNetCore.Identity.UI**



dbo.DeadPeople oraz dbo.Lodgings [Tabela Zmarli oraz Kwatery]

DESKTOP-IUJAJB6\L... - dbo.DeadPeople -> X DESKTOP-IUJAJB6\L... - d			
	Column Name	Data Type	Allow Nulls
∞	IdDeadPerson	int	<input type="checkbox"/>
	Name	varchar(75)	<input checked="" type="checkbox"/>
	DateOfBirth	nvarchar(MAX)	<input checked="" type="checkbox"/>
	DateOfDeath	nvarchar(MAX)	<input type="checkbox"/>
	LodgingId	int	<input type="checkbox"/>
			<input type="checkbox"/>

DESKTOP-IUJAJB6\L...ST - dbo.Lodgings -> X DESKTOP-IUJAJB6\L... - d			
	Column Name	Data Type	Allow Nulls
1	IdLodge	int	<input type="checkbox"/>
	LodgingType	int	<input type="checkbox"/>
	isMonument	bit	<input type="checkbox"/>
	PeopleLimit	int	<input type="checkbox"/>
	isReserved	bit	<input type="checkbox"/>
			<input type="checkbox"/>

dbo.News [Tabela Aktualności]

DESKTOP-IUJAJB6\L...seTEST - dbo.News -> X DESKTOP-IUJAJB6\L...ST			
	Column Name	Data Type	Allow Nulls
▶	Id	int	<input type="checkbox"/>
	Title	nvarchar(MAX)	<input type="checkbox"/>
	DateOfPublication	nvarchar(MAX)	<input type="checkbox"/>
	NewsContent	nvarchar(MAX)	<input type="checkbox"/>
			<input type="checkbox"/>

dbo.Obituaries [Tabela Nekrologów]

DESKTOP-IUJAJB6\L...T - dbo.Obituaries -> X DESKTOP-IUJAJB6\L...seT			
	Column Name	Data Type	Allow Nulls
▶	Id	int	<input type="checkbox"/>
	Name	nvarchar(70)	<input type="checkbox"/>
	DateOfDeath_Obituary	nvarchar(MAX)	<input type="checkbox"/>
	ObituaryContent	nvarchar(MAX)	<input type="checkbox"/>

Zastosowane technologie

Angular 8

Angular 8 to otwarty framework oparty na języku JavaScript, wspierany i firmowany przez Google, wspomagający tworzenie i rozwój aplikacji internetowych na pojedynczej stronie. Zadaniem biblioteki jest wdrożenie wzorca MVC do aplikacji internetowych dla ułatwienia ich rozwoju i procesu testowania.

ASP.NET Core 3.1

ASP.NET Core 3.1 to wolne i otwarte oprogramowanie pozwalające tworzyć i uruchamiać wydajne aplikacje na platformach Windows, Linux, macOS. Framework ten umożliwia programowanie aplikacji przeznaczonych dla chmury obliczeniowej oraz IoT, a także back-endu aplikacji internetowych z użyciem wzorca MVC. Programy na *.NET Core* mogą być pisane przy pomocy języków C#, F#, oraz Visual Basic.

T-SQL

T-SQL oznacza transakcyjny SQL, czyli rozszerzenie języka SQL umożliwiające tworzenie konstrukcji takich jak pętle, instrukcje warunkowe oraz zmienne. Jest używany do tworzenia wyzwalaczy, procedur i funkcji składowanych w bazie. Został stworzony przez Sybase i wbudowany do serwerów SQL tej firmy, później prawa kupiła firma Microsoft i wykorzystuje ten język w kolejnych wersjach MS SQL Server.



Microsoft®
SQL Server®

Opis projektowanego systemu

Kontrolery:

AdminController

Konstruktor:

```
public SignInManager<IdentityUser> signInManager;
public UserManager<IdentityUser> userManager;
public RoleManager<IdentityRole> roleManager;
public IConfiguration configuration;
private readonly ILogger logger;
private readonly ApplicationDbContext context;
Odwolania: 0
public AdminController( ApplicationDbContext _context, IConfiguration _configuration,
                        SignInManager<IdentityUser> _signInManager,
                        UserManager<IdentityUser> _userManager,
                        RoleManager<IdentityRole> _roleManager, ILogger<AdminController> _logger)
{
    context = _context;
    configuration = _configuration;
    signInManager = _signInManager;
    userManager = _userManager;
    roleManager = _roleManager;
    logger = _logger;
}
```

SignInManager – menadżer logowania się w aplikacji

UserManager - menadżer użytkowników w serwisie

RoleManager – menadżer ról użytkowników w serwisie

IConfiguration – konfiguracja projektu przez Startup, z konfiguracji otrzymuje adres bazy danych

ApplicationDbContext – kontekst na podstawie którego tworzeni są użytkownicy oraz modyfikacje

Najważniejsze metody kontrolerów:

BlockEmployee

```
[HttpPost]
Odwolania: 0
public async Task<IActionResult> BlockEmployee(List<RegisterViewModel> model)
{
    string blockedEmployee = "BLOCKEDEMPLOYEE";
    string employee = "EMPLOYEE";

    for (int i = 0; i < model.Count; i++)
    {
        model[i].Role = employee;
        var users = await userManager.GetUsersInRoleAsync(employee);
        var user = await userManager.FindByEmailAsync(model[i].Email);
        //var user = await userManager.FindByIdAsync(model[i].UserId);
        IdentityResult result = null;
        //
        if (model[i].IsSelected && !(await userManager.IsInRoleAsync(user, blockedEmployee)))
        {
            await userManager.RemoveFromRoleAsync(user, employee);
            result = await userManager.AddToRoleAsync(user, blockedEmployee);
        }
        else if (model[i].IsSelected && model[i].Role == null)
        {
            await userManager.AddToRoleAsync(user, blockedEmployee);
        }
    }
    else
    {
        continue;
    }
    if (result.Succeeded)
    {
        if (i < (model.Count - 1))
            continue;
        else
            return RedirectToAction("BlockEmployee", "Admin");
    }
    foreach (var error in result.Errors)
    {
        ModelState.AddModelError("", error.Description);
    }
}

return View(model);
}
```


UnblockEmployee

```
[HttpPost]
Odwolania: 0
public async Task<IActionResult> UnblockEmployee(List<RegisterViewModel> model)
{
    string blockedEmployee = "BLOCKEDEMPLOYEE";
    string employee = "EMPLOYEE";

    for (int i = 0; i < model.Count; i++)
    {
        model[i].Role = blockedEmployee;
        var users = await userManager.GetUsersInRoleAsync(blockedEmployee);
        var user = await userManager.FindByEmailAsync(model[i].Email);
        //var user = await userManager.FindByIdAsync(model[i].UserId);
        IdentityResult result = null;
        //
        if (model[i].IsSelected && !(await userManager.IsInRoleAsync(user, employee)))
        {
            await userManager.RemoveFromRoleAsync(user, blockedEmployee);
            result = await userManager.AddToRoleAsync(user, employee);
        }
        else if (model[i].IsSelected && model[i].Role == null)
        {
            await userManager.AddToRoleAsync(user, employee);
        }
        else
        {
            continue;
        }
        if (result.Succeeded)
        {
            if (i < (model.Count - 1))
                continue;
            else
                return RedirectToAction("UnblockEmployee", "Admin");
        }
        foreach (var error in result.Errors)
        {
            ModelState.AddModelError("", error.Description);
        }
    }

    return View(model);
}
```

Widok:

 Cmentarz Komunalny

Strona główna admina

Nowy admin

Nowy pracownik

Zablokuj pracownika

Odblokuj pracownika

Spis użytkowników

Odblokuj pracownika

Odblokuj

Powrót

© Cmentarz Komunalny
Projekt zespołowy

RegisterEmployee

```
[HttpPost]
Odwolania: 0
public async Task<IActionResult> RegisterEmployee(RegisterViewModel model)
{
    if (ModelState.IsValid)
    {
        var user = new IdentityUser { UserName = model.Email, Email = model.Email };
        var result = await userManager.CreateAsync(user, model.Password);

        if (result.Succeeded)
        {
            // await signInManager.SignInAsync(user, isPersistent: false);
            await userManager.AddToRoleAsync(user, "EMPLOYEE");
            return RedirectToAction("Index", "Admin");
        }
        foreach (var error in result.Errors)
        {
            ModelState.AddModelError("", error.Description);
        }
    }
    return View(model);
}
```

Widok:

⇒ [Cmentarz Komunalny](#)

[Strona główna admina](#)

[Nowy admin](#)

[Nowy pracownik](#)

[Zablokuj pracownika](#)

[Odblokuj pracownika](#)

[Spis użytkowników](#)

Stwórz nowego pracownika w serwisie Cmentarza

Stwórz nowego użytkownika.

Email

Hasło

Potwierdź hasło

[Stwórz pracownika](#)

RegisterAdmin

```
[HttpPost]
Odwolania: 0
public async Task<IActionResult> RegisterAdmin(RegisterViewModel model, string roleId)
{
    if (ModelState.IsValid)
    {
        var roles = roleManager.Roles;
        var role = await this.roleManager.FindByIdAsync(roleId);

        var user = new IdentityUser { UserName = model.Email, Email = model.Email, /* Role = model.Role*/ };
        var result = await userManager.CreateAsync(user, model.Password);

        if (result.Succeeded)
        {
            await signInManager.SignInAsync(user, isPersistent: false);
            await userManager.AddToRoleAsync(user, "ADMINISTRATOR");
            return RedirectToAction("Index", "Admin");
        }
        foreach (var error in result.Errors)
        {
            ModelState.AddModelError("", error.Description);
        }
    }
    ViewBag.RoleUzytkownikow = new SelectList(roleManager.Roles, "Name", "Name");
    return View(model);
}
```

[DeadPerson/News/Lodging/Obituary]Controller

W tej sekcji opisane kontrolery będą opisane na podstawie DeadPerson. Dlaczego? Ponieważ wszystkie kontrolery działają na podobnej zasadzie, powielanie zrzutów ekranu nie ma większego konieczne.

Konstruktor:

```
private readonly MockDeadPeopleRepo _mockRepo = new MockDeadPeopleRepo();
private readonly IConfiguration _configuration;
private readonly IDeadPeopleRepo _repository; // dependency injection
private readonly IMapper _mapper;

Odwolania: 0
public DeadPersonController(IConfiguration configuration,
                             IDeadPeopleRepo repository,
                             IMapper mapper)
{
    _configuration = configuration;
    _repository = repository;
    _mapper = mapper;
}
```

IDeadPeopleRepo – interfejs repozytorium przechowującego dane nt. zmarłych osób, jednakże wraz z rozwojem aplikacji i wykorzystaniu bezpośredniego kontaktu z bazą przy pomocy JSON, nie jest on wykorzystywany. Tak samo ma się rzecz z pozostałymi kontrolerami.

IMapper – interfejs AutoMapper’a, tak samo nie jest wykorzystywany ze względów jak wyżej.

IConfiguration – konfiguracja projektu poprzez Startup, z konfiguracji otrzymuje adres bazy danych

Najważniejsze metody kontrolerów:

GET

```
[HttpGet]
Odwolania: 0
public JsonResult Get()
{
    string query = @"
        select IdDeadPerson, LodgingId, Name, DateOfBirth, DateOfDeath from dbo.DeadPeople";
    DataTable table = new DataTable();
    string sqlDataSource = _configuration.GetConnectionString("CmentarzConnectionTEST");
    SqlDataReader myReader;
    using (SqlConnection myCon = new SqlConnection(sqlDataSource))
    {
        myCon.Open();
        using (SqlCommand myCommand = new SqlCommand(query, myCon))
        {
            myReader = myCommand.ExecuteReader();
            table.Load(myReader); ;

            myReader.Close();
            myCon.Close();
        }
    }
    return new JsonResult(table);
}
```

POST

```
[Authorize(Policy = "RequireAdministratorRole")]
[Authorize(Policy = "RequireEmployeeRole")]
[HttpPost]
Odwolania: 0
public JsonResult Post(DeadPerson deadp)
{
    string query = @"
        insert into dbo.DeadPeople values
        (N'" + deadp.Name + @"', N'" + deadp.DateOfBirth + @"',
        N'" + deadp.DateOfDeath + @"', '"" + deadp.LodgingId + @"'");

    DataTable table = new DataTable();
    string sqlDataSource = _configuration.GetConnectionString("CmentarzConnectionTEST");
    SqlDataReader myReader;
    using (SqlConnection myCon = new SqlConnection(sqlDataSource))
    {
        myCon.Open();
        using (SqlCommand myCommand = new SqlCommand(query, myCon))
        {
            myReader = myCommand.ExecuteReader();
            table.Load(myReader); ;

            myReader.Close();
            myCon.Close();
        }
        return new JsonResult("Dodano pomyslnie");
    }
}
```

PUT

```
[Authorize(Policy = "RequireAdministratorRole")]
[Authorize(Policy = "RequireEmployeeRole")]
[HttpPut]
Odwolania: 0
public JsonResult Put(DeadPerson deadp)
{
    string query = @"
        update dbo.DeadPeople set
        Name = N'" + deadp.Name + @"',
        DateOfBirth = '"" + deadp.DateOfBirth + @"',
        DateOfDeath = '"" + deadp.DateOfDeath + @"',
        LodgingId = '"" + deadp.LodgingId + @"'
        where IdDeadPerson = '"" + deadp.IdDeadPerson + @"'";

    DataTable table = new DataTable();
    string sqlDataSource = _configuration.GetConnectionString("CmentarzConnectionTEST");
    SqlDataReader myReader;
    using (SqlConnection myCon = new SqlConnection(sqlDataSource))
    {
        myCon.Open();
        using (SqlCommand myCommand = new SqlCommand(query, myCon))
        {
            myReader = myCommand.ExecuteReader();
            table.Load(myReader); ;

            myReader.Close();
            myCon.Close();
        }
        return new JsonResult("Zaktualizowano pomyslnie");
    }
}
```

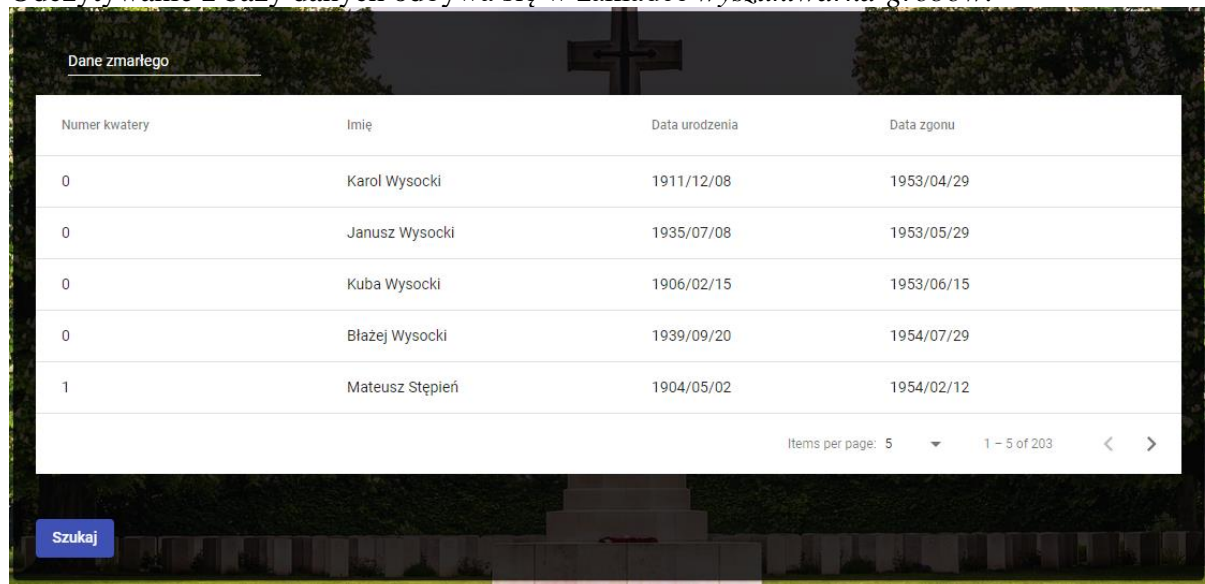
DELETE

```
[Authorize(Policy = "RequireAdministratorRole")]
[Authorize(Policy = "RequireEmployeeRole")]
[HttpDelete("{id}")]
Odwolania: 0
public JsonResult Delete(int id)
{
    string query = @"
        delete from dbo.DeadPeople
        where IdDeadPerson = " + id + @"";

    DataTable table = new DataTable();
    string sqlDataSource = _configuration.GetConnectionString("CmentarzConnectionTEST");
    SqlDataReader myReader;
    using (SqlConnection myCon = new SqlConnection(sqlDataSource))
    {
        myCon.Open();
        using (SqlCommand myCommand = new SqlCommand(query, myCon))
        {
            myReader = myCommand.ExecuteReader();
            table.Load(myReader); ;

            myReader.Close();
            myCon.Close();
        }
        return new JsonResult("Zaktualizowano pomyślnie");
    }
}
```

Odczytywanie z bazy danych odbywa się w zakładce *wyszukiwarka-grobow*:



Dane zmarłego

Numer kwatery	Imię	Data urodzenia	Data zgonu
0	Karol Wysocki	1911/12/08	1953/04/29
0	Janusz Wysocki	1935/07/08	1953/05/29
0	Kuba Wysocki	1906/02/15	1953/06/15
0	Błażej Wysocki	1939/09/20	1954/07/29
1	Mateusz Stępień	1904/05/02	1954/02/12

Items per page: 5 1 - 5 of 203

Szukaj

Widok aktualności:

Cmentarz Komunalny Rocznice Aktualności Nekrologi Informacje Wyszukiwarka grobów Mapa Kontakt

Renowacja ogrodzenia

Upzejmie informujemy, że w okresie czasu od 1.09.2020r. do 20.09.2020r. odbędzie się renowacja uszkodzonych elementów ogrodzenia cmentarza. Osoby odwiedzające groby prosimy o zachowanie ostrożności w okolicach prac.

Data wpisu: 2020/08/10

Akcja "Kupujemy kwiaty"

Z racji pandemii która obecnie opanowała cały świat, w dniu Wszystkich Świętych cmentarze zostaną zamknięte. Jednakże nie zapominajmy o osobach które z tego powodu poniosą duże koszty finansowe. wesprzyjmy kwiaciarzy i odkupmy od nich kwiaty, by te małe biznesy które co roku zapewniają nam piękne bukiety na groby naszych zmarłych mogły przetrwać ten trudny okres.

Data wpisu: 2020/11/09

Zamknięcie cmentarza.

W dniu 7 listopada rząd w trosce o nasze zdrowie podjął trudną decyzję o zamknięciu cmentarzy na czas bliżej nie określony. Spowodowane jest to panującą na świecie pandemią i wzrostem zachorowań. Jest nam niezmiernie przykro ale w tym roku nie będzie możliwe odwiedzenia grobów w okresie Wszystkich Świętych. Życzymy zdrowia.

Data wpisu: 2020/11/07

Sprzątanie świata.

Akcja sprzątanie świata zawitała również na nasz cmentarz. Chętnych do pomocy oczyszczenia cmentarza jak i okolic cmentarnych ze zużytych zniczy, zwiędłych kwiatów oraz ogólnie pojętych śmieci zapraszamy 18.09. o godzinie 10:00. Rękawiczki ochronne oraz worki na śmieci zapewni cmentarz.

Data wpisu: 2020/09/01

Widok nekrologi:

Dnia 2020/04/02 zmarł/a

Ś.P. Łukasz Kaźmierczak

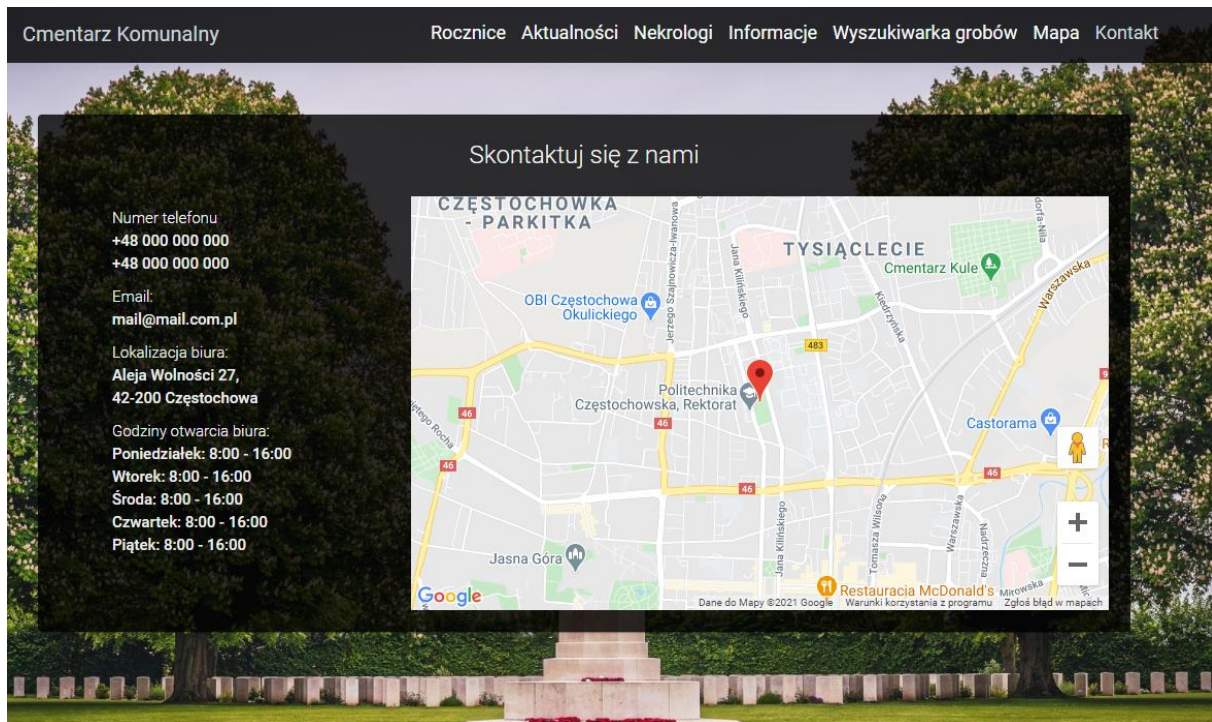
W dniu 31 kwietnia pożegnaliśmy z naszego grona Pana Łukasza. Ukochanego męża i przyjaciela. Panie, świeć oraz miej go w swej opiece. Nabożeństwo żałobne odbędzie się 4 kwietnia o godzinie 12:00. Po czym nastąpi odprowadzenie drogiego nam Łukasza na cmentarz.

Dnia 2020/01/15 zmarł/a

Ś.P. Igor Wiśniewski

W dniu 31 stycznia pożegnaliśmy z naszego grona Pana Igora. Ukochanego męża i przyjaciela. Panie, świeć oraz miej go w swej opiece. Nabożeństwo żałobne odbędzie się 19 stycznia o godzinie 12:00. Po czym nastąpi odprowadzenie drogiego nam Igora na cmentarz.

Widok kontakt:



Odczytywanie danych z bazy w pliku shared.service.ts:

```
getZmarliList(): Observable<any[]> {
  return this.http.get<any>(this.APIUrl + '/deadperson');
}

addDeadPerson(val: any) {
  return this.http.post(this.APIUrl + '/deadperson/', val);
}

putDeadPerson(val: any) {
  return this.http.put(this.APIUrl + '/deadperson/', val);
}

deleteDeadPerson(val: any) {
  return this.http.delete(this.APIUrl + '/deadperson/' + val);
}
```

```
getAktualnosciList(): Observable<any[]> {
  return this.http.get<any>(this.APIUrl + '/news');
}
/* METODA DODAJĄCA DANE DO BAZY */
addAktualnosci(val: any) {
  return this.http.post(this.APIUrl + '/news', val);
}
/* METODA AKTUALIZUJĄCA DANE DO BAZY */
putAktualnosci(val: any) {
  return this.http.put(this.APIUrl + '/news', val);
}
/* METODA AKTUALIZUJĄCA DANE DO BAZY (usuwanie po ID aktualnosci) */
deleteAktualnosci(val: any) {
  return this.http.delete(this.APIUrl + '/news/' + val);
}

/* METODA KONSOLIDUJĄCA DANE 2 API */
getNekrologiList(): Observable<any[]> {
  return this.http.get<any>(this.APIUrl + '/obituary');
}
/* METODA DODAJĄCA DANE DO BAZY */
addNekrologi(val: any) {
  return this.http.post(this.APIUrl + '/obituary', val);
}
/* METODA AKTUALIZUJĄCA DANE DO BAZY */
putNekrologi(val: any) {
  return this.http.put(this.APIUrl + '/obituary', val);
}
/* METODA AKTUALIZUJĄCA DANE DO BAZY (usuwanie po ID aktualnosci) */
deleteNekrologi(val: any) {
  return this.http.delete(this.APIUrl + '/obituary/' + val);
}
```

Odczytywanie danych przekazane do panelów administratora oraz pracownika, wraz z metodami wysyłającymi / aktualizującymi / usuwającymi dane z bazy:

Przykład edytowania aktualności:

```
183 //Edytuj "Aktualności"
184 GetDataFromEdytujAktualnosc() {
185     //pobierz tytuł
186     this.inputEdytujAktualnoscTytul = (document.getElementById("edytujAktualnoscTytul") as HTMLInputElement).value;
187     //pobierz treść wiadomości
188     this.inputEdytujAktualnoscTresc = (document.getElementById("edytujAktualnoscTresc") as HTMLTextAreaElement).value;
189     //pobierz tytuł
190     this.inputEdytujAktualnoscData = (document.getElementById("edytujAktualnoscData") as HTMLInputElement).value;
191
192     //Wczytanie wszystkich danych do tablicy
193     this.aktualnosciEdit = this.selectionAktualnosci.selected.map(x => x.Id).join("");
194
195     if (this.inputEdytujAktualnoscTytul == "") {
196         this.inputEdytujAktualnoscTytul = this.selectionAktualnosci.selected.map(x => x.Title).join("");
197     }
198
199     if (this.inputEdytujAktualnoscTresc == "") {
200         this.inputEdytujAktualnoscTresc = this.selectionAktualnosci.selected.map(x => x.NewsContent).join("");
201     }
202
203     if (this.inputEdytujAktualnoscData == "") {
204         this.inputEdytujAktualnoscData = this.selectionAktualnosci.selected.map(x => x.DateOfPublication).join("");
205     }
206
207     const aktualnoscjson = {
208         Id: this.aktualnosciEdit,
209         Title: this.inputEdytujAktualnoscTytul,
210         NewsContent: this.inputEdytujAktualnoscTresc,
211         DateOfPublication: this.inputEdytujAktualnoscData,
212     };
213     if (this.aktualnosciEdit == "") { alert("Nie wybrano żadnej aktualności, spróbuj ponownie") }
214     else if (confirm) {
215         alert("Czy na pewno?")
216         this.service.putAktualnosci(aktualnoscjson).subscribe();
217
218         this.refreshAktualnosciList();
219         //Sprawdzenie
220         console.log(this.inputEdytujAktualnoscTytul);
221         console.log(this.inputEdytujAktualnoscTresc);
222         console.log(this.inputEdytujAktualnoscData);
223     }
224     this.refreshAktualnosciList();
225 }
226
```

Przykład dodawania aktualności:

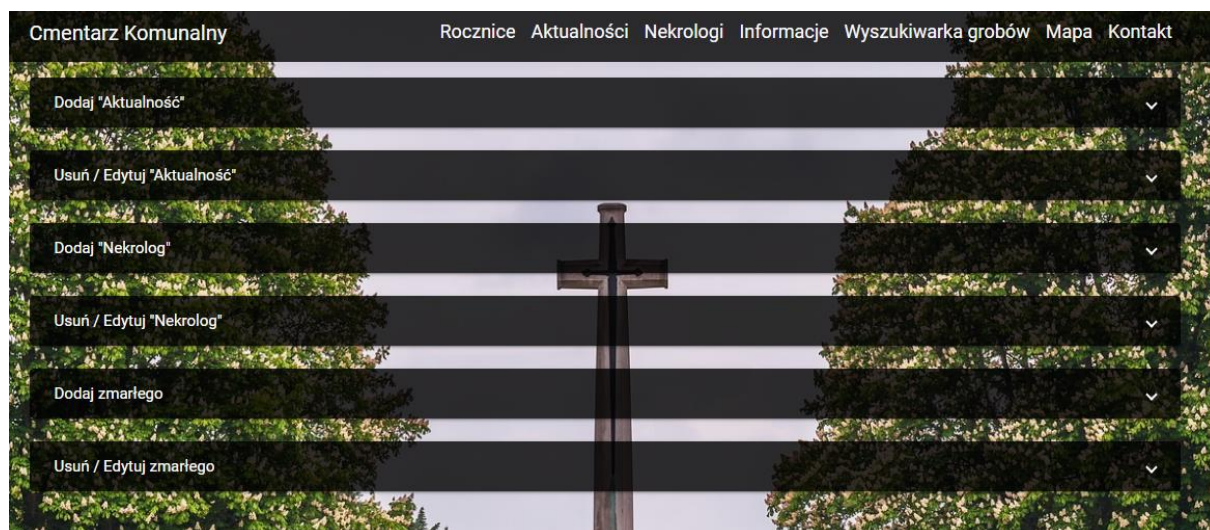
```
133 //Dodaj "Aktualności"
134 GetInputFromDodajAktualnosc() {
135     //pobierz tytuł
136     let inputDodajAktualnoscTytul = (document.getElementById("dodajAktualnoscTytul") as HTMLInputElement).value;
137     //pobierz treść wiadomości
138     let inputDodajAktualnoscTresc = (document.getElementById("dodajAktualnoscTresc") as HTMLTextAreaElement).value;
139     //pobierz tytuł
140     let inputDodajAktualnoscData = (document.getElementById("dodajAktualnoscData") as HTMLInputElement).value;
141
142     const aktualnoscjson = {
143         Title: inputDodajAktualnoscTytul,
144         DateOfPublication: inputDodajAktualnoscData,
145         NewsContent: inputDodajAktualnoscTresc
146     };
147
148     this.service.addAktualnosci(aktualnoscjson).subscribe(akt => this.service.addAktualnosci(aktualnoscjson));
149     alert("Dodano aktualność");
150     this.refreshAktualnosciList();
151
```

Przykład usuwania aktualności:

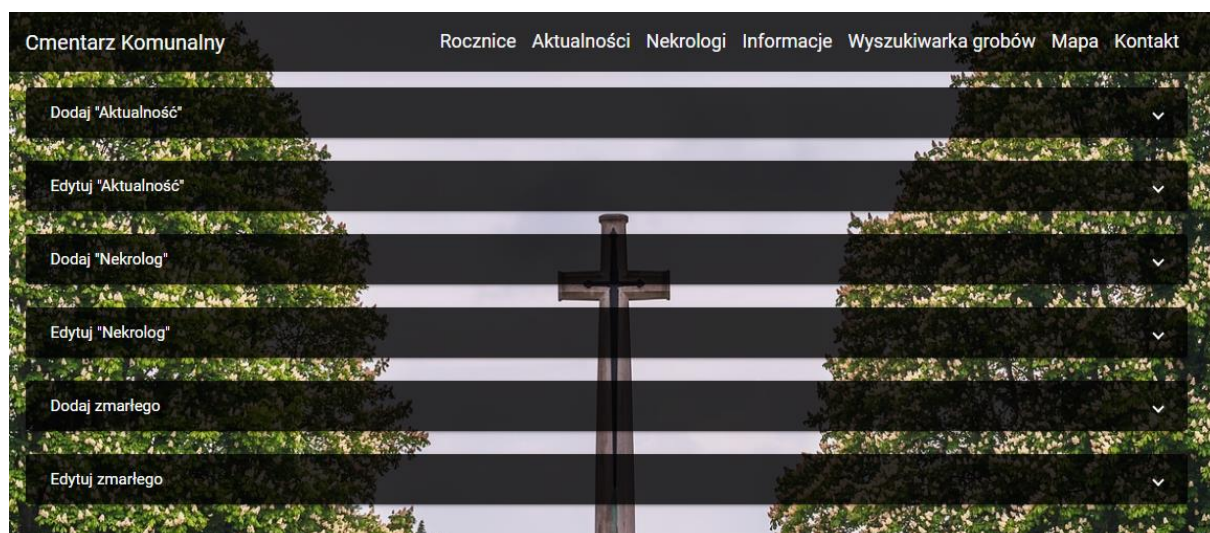
```
158 //Usun "Aktualnosci"
159 GetCheckedUsunAktualnosci() {
160     //Wczytanie wszystkich danych do tablicy
161
162     this.aktualnosciDelete = this.selectionAktualnosci.selected.map(x => x.Id);
163
164     if (confirm("Czy jesteś pewny?"))
165     {
166         if (this.aktualnosciDelete == "") { alert("Nie zaznaczono aktualności, spróbuj ponownie") }
167         else {
168             this.service.deleteAktualnosci(this.aktualnosciDelete).subscribe();
169
170             alert("Usunięto aktualność")
171             //Sprawdzenie
172             console.log(this.aktualnosciDelete);
173             console.log(this.service.deleteAktualnosci(this.aktualnosciDelete).subscribe());
174         }
175         this.refreshAktualnosciList();
176     }
177 }
```

W bardzo podobnym stylu zrealizowane są wszystkie pozostałe metody.

Widok *paneladministratora*:



Widok *panelpracownika*:



Wymagania sprzętowe oraz programowe niezbędne do pełnej implementacji

Rodzaj aplikacji: Aplikacja internetowa
Platformy docelowe: Przeglądarki internetowe na komputerze / telefonie

Opis sposobu implementacji systemu w środowisku produkcyjnym

Zainstalowane pakiety NuGet:

- **Microsoft.EntityFrameworkCore** **v3.1.3**
Entity Framework (EF) Core to lekkie, rozszerzalne i wieloplatformowe wersje popularnej technologii Entity Framework dostępu do danych. W przypadku EF Core dostęp do danych odbywa się przy użyciu modelu, składa się on z klas jednostki i obiektu kontekstu, który reprezentuje sesję z bazą danych. Obiekt Context umożliwia wykonywanie zapytań oraz zapis danych.
- **Microsoft.EntityFrameworkCore.Relational** **v3.1.3**
Udostępnione komponenty dla relacyjnych baz danych.
- **Microsoft.EntityFrameworkCore.Tools** **v3.1.1**
Narzędzia pomagające w design-time rozwijające aplikacje. Głównie używane do zarządzania migracjami i tworzenia układu szkieletowego na podstawie DbContext.
- **Microsoft.EntityFrameworkCore.Design** **v3.1.3**
Design-time komponenty udostępnione dla EF Core.
- **Microsoft.EntityFrameworkCore.SqlServer** **v3.1.3**
Dostawca bazodanowy Microsoft SQL Server dla EF Core.
- **Microsoft.AspNetCore.Services.Extensions** **v3.1.1**
Wspomagacze (Helpers) w budowaniu pojedynczych stron w ASP.NET MVC.
- **Microsoft.AspNetCore.Diagnostics.EntityFrameworkCore** **v3.1.1**
Zestaw diagnostyczny w przypadkach gdy na stronach dochodzi do błędów.
- **Microsoft.AspNetCore.Identity.UI** **v3.1.1**
Domyślne strony z UI na bazie Razor.
- **Microsoft.AspNetCore.Newtonsoft.Json** **v3.1.3**
Funkcjonalności MVC które używają Newtonsoft.Json, zawiera formatowanie input oraz output w formacie JSON oraz JSON PATCH.
- **Microsoft.AspNetCore.Identity.EntityFrameworkCore** **v3.1.3**
Dostawca Identity który używa EF Core.
- **System.Data.SqlClient** **v4.8.2**
Dostarcza dostęp do bazy danych w SQL Server. Te klasy posiadają dostęp do wersji SQL Server oraz protokołów dostępnych dla danej bazy danych.
- **System.Net.Http** **v4.3.4**
Dostarcza interfejs programistyczny dla nowoczesnych aplikacji HTTP, zawiera komponenty klienta które pozwalają aplikacjom na otrzymywanie serwisów.
- **AutoMapper** **v8.1.0**
Rozszerzenie do ASP.Net Core, automatyczne mapowanie pomiędzy obiektami a ich widokami. Użyte przede wszystkim do testowania aplikacji.

Wymagane programy:

Microsoft Visual Studio 2019

Cel: Kompilacja oraz tworzenie aplikacji

Microsoft SQL Server Management Studio 18

Cel: Zarządzanie bazą danych oraz migracjami

<https://docs.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-ver15>

node.js 14.15.4 LTS

Cel: Uruchomienie obsługi oraz instalacji Angular

<https://nodejs.org/en/>

AngularJS CLI (min. wersja 8.0)

Cel: Front-end aplikacji, wizerunek

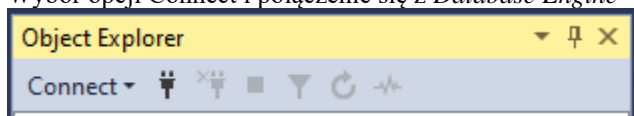
Instalacja poprzez wiersz poleceń

Proces instalacji aplikacji:

Wszystkie pakiety NuGet są zainstalowane wraz z przesłanym projektem, nie ma konieczności ich ponownej instalacji.

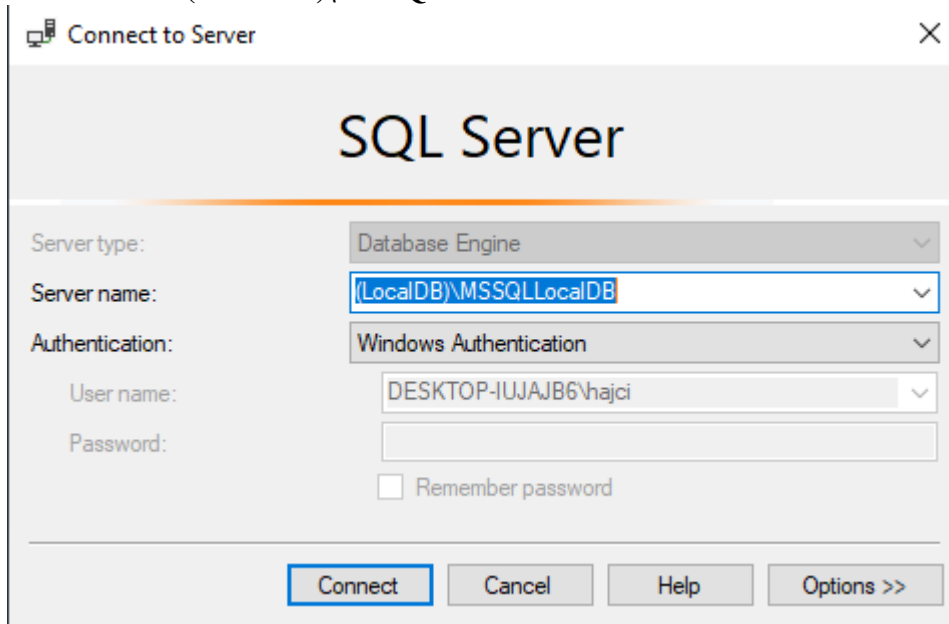
Konfiguracja SQL Server Management Studio 18:

Wybór opcji Connect i połączenie się z *Database Engine*



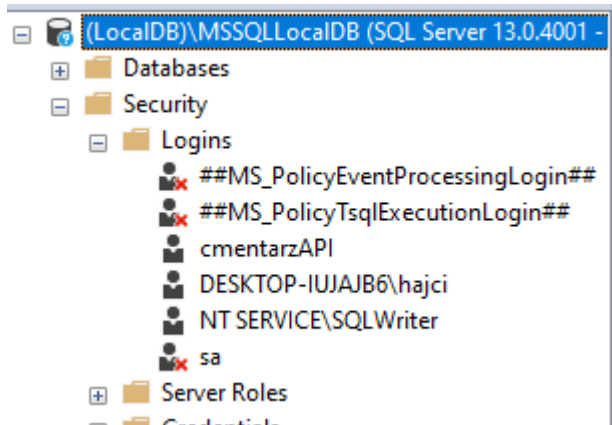
Wpisanie nazwy serwera

Server name: **(LocalDB)\MSSQLLocalDB**



Połączenie z bazą można zrobić na dwa sposoby autentykacji. Wszystko można zrobić bez potrzeby robienia autentykacji SQL Server, wystarczy Windows Authentication. Jednakże przedstawię proces w jaki to osiągnąć, jeśli by nie działało.

Logujemy się za pomocą Windows Authentication i klikamy *Connect*:



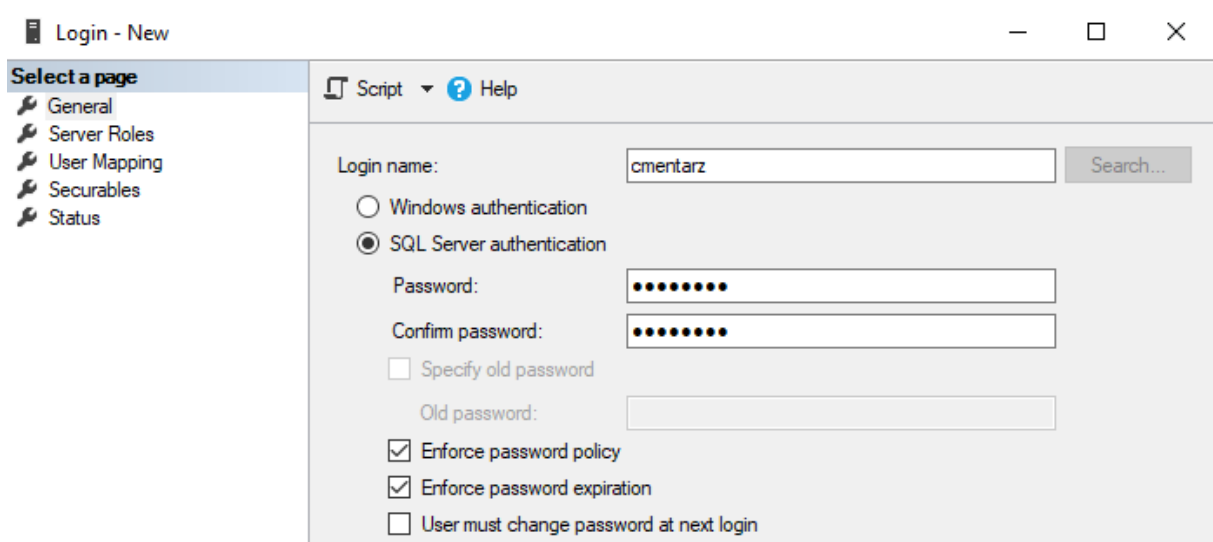
Przechodzimy do *Logins* i prawym przyciskiem myszy wybieramy *New Login*, uzupełniamy dane w następujący sposób:

Uwaga: Trzeba koniecznie zaznaczyć SQL Server authentication.

Dane mogą być dowolne, przykład:

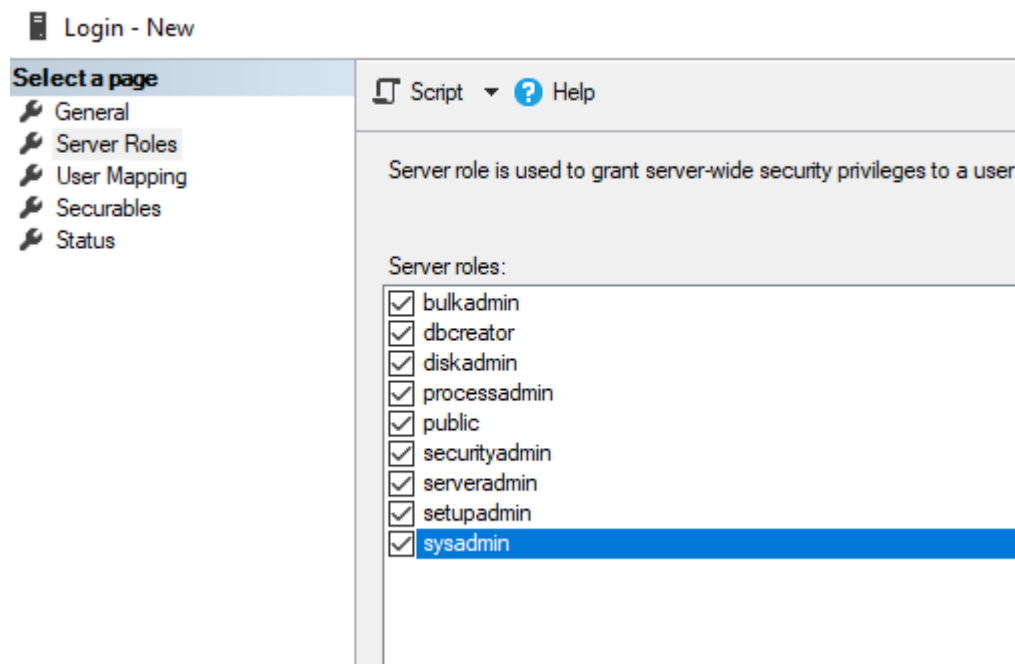
Login name: cmentarz

Password: cmentarz

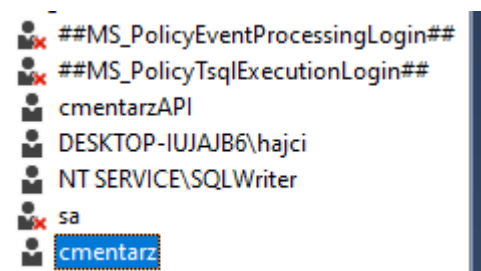



Uwaga: Proszę pamiętać o odhaczeniu „User must change password at next login”.

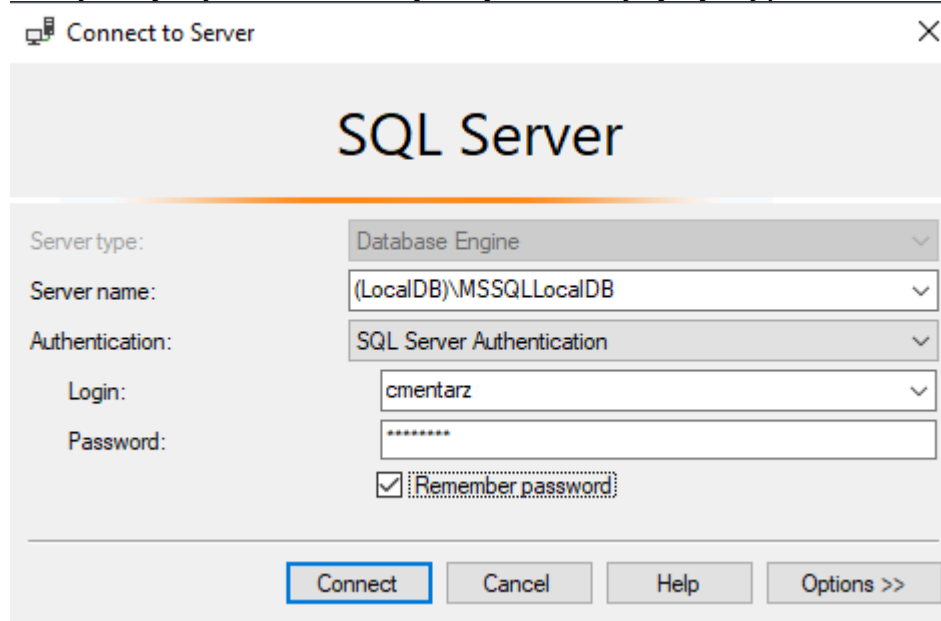
Przechodzimy do *Server Roles* i zaznaczamy wszystkie role by mieć 100% pewność, że wszystko będzie działało poprawnie:



Po wszystkim klikamy OK. Login został utworzony:



Rozłączamy się z silnikiem bazy danych  i łączymy się ponownie:



Najwyższa pora stworzyć bazę danych w programie Visual Studio:
Opcje mogą pozostać wprowadzone przez zespół, jednakże jeśli wymagana jest zmiana to:

UWAGA: W ramach bezpieczeństwa tworzenia systemu oraz uniknięcia występowania powielania się migracji, poradnik pracuje na CmentarzConnectionTEST2. Oficjalna implementacja jest zagwarantowana na CmentarzConnection.

Plik appsettings.json:

Trzeba wprowadzić *connection string* dla bazy danych na taki sposób:

```
{
  "ConnectionStrings": {
    // "CommanderConnection": "Server=(LocalDB)\\MSSQLLocalDB;Initial Catalog=CommanderDB; User ID=CmentarzAPI;Password=pass;",
    "CmentarzConnection": "Data Source=(LocalDB)\\MSSQLLocalDB; Initial Catalog=CmentarzDatabase; Integrated Security=true; Trusted_Connection=True",
    "CmentarzConnectionTEST": "Data Source=(LocalDB)\\MSSQLLocalDB; Initial Catalog=CmentarzDatabaseTEST; Integrated Security=true; Trusted_Connection=True",
    "CmentarzConnectionTEST2": "Data Source=(LocalDB)\\MSSQLLocalDB; Initial Catalog=CmentarzDatabaseTEST2; Integrated Security=true; Trusted_Connection=True"
    // Commander is for testing purposes
  }
}
```

Wybieramy CmentarzConnection który utworzy w SQL Server bazę CmentarzDatabase.

"CmentarzConnection": "Data Source=(LocalDB)\\MSSQLLocalDB; Initial Catalog=CmentarzDatabase; Integrated Security=true; Trusted_Connection=True"

Plik Startup.cs:

W tym pliku uruchamiana jest cała aplikacja we współpracy z *Program.cs* oraz z innymi wszystkimi plikami.

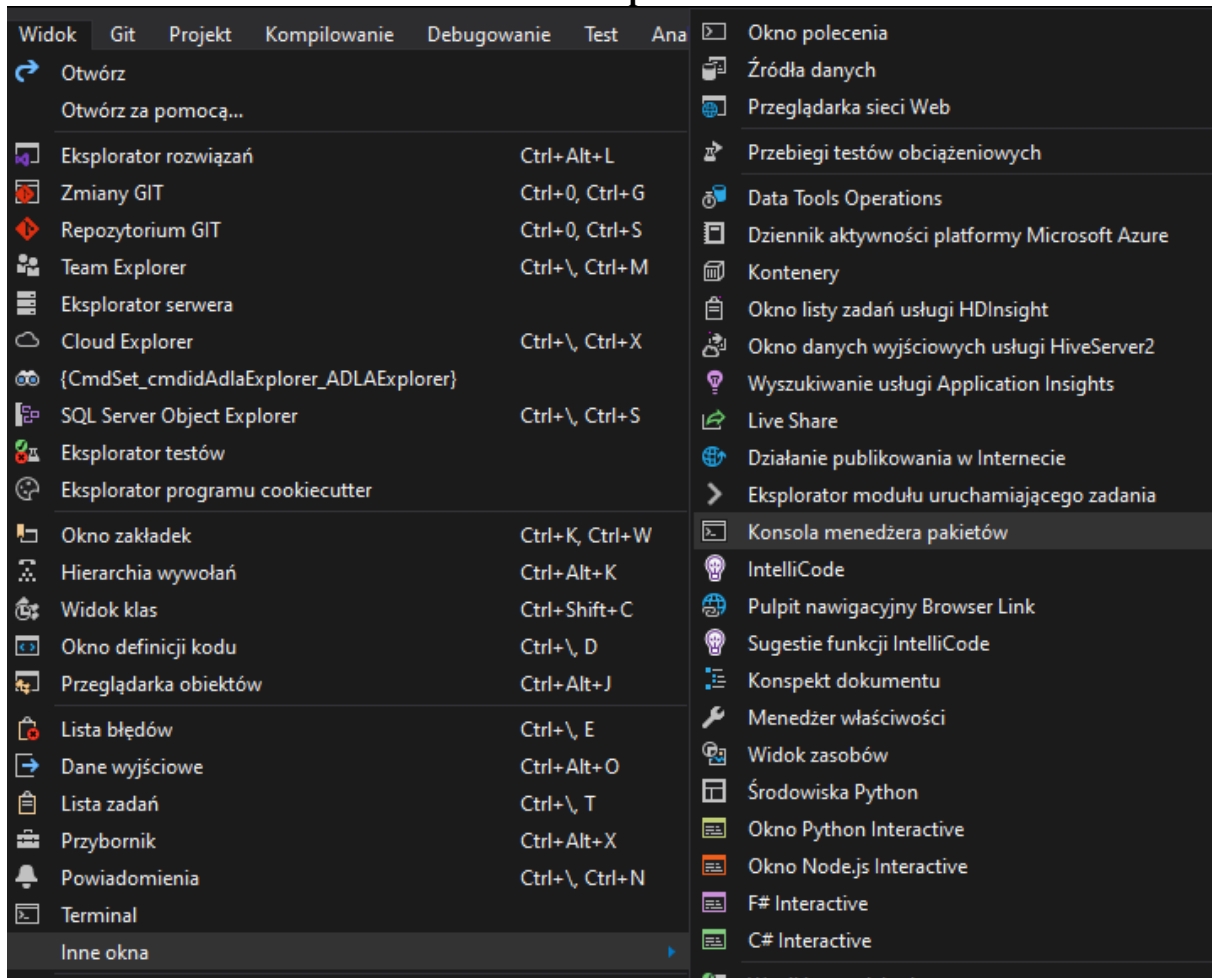
```
40 // we're adding context over here !!!!!!!!!!!
41 // switch it up for CmentarzContext when it will be ready to go
42 services.AddDbContext<CmentarzContext>(options =>
43     options.UseSqlServer(
44         Configuration.GetConnectionString("CmentarzConnection")));
45 services.AddDbContext<ApplicationDbContext>(options =>
46     options.UseSqlServer(
47         Configuration.GetConnectionString("CmentarzConnection")));
48
```

W liniach 44 oraz 47 trzeba wprowadzić naszego connection stringa, jest wprowadzony domyślnie odpowiedni do uruchomienia.

W tej chwili jesteśmy gotowi do stworzenia migracji oraz zaktualizowania naszej bazy danych.

Musimy otworzyć konsolę menadżera pakietów:

Widok => Inne Okna => Konsola menedżera pakietów



Jako, że nasza aplikacja wykorzystuje dwa konteksty (jeden dotyczy cmentarza, drugi użytkowników na bazie *Identity*), to musimy wykonać następujące operacje w konsoli menadżera pakietów:

```
Konsola menedżera pakietów
Źródło pakietów: Wszystkie
Projekt domyślny: CmentarzKomunalny.Web
PM> Add-Migration InitialTest2 -Context ApplicationDbContext -verbose
Using project 'CmentarzKomunalny.Web'.
Using startup project 'CmentarzKomunalny.Web'.
Build started...
Build succeeded.
```

[Na zrzucie ekranu są migracje tworzone na CmentarzConnectionTEST2]

Add-Migration InitialOfficial -Context ApplicationDbContext -verbose

Dodanie migracji o nazwie InitialOfficial bazując na kontekście ApplicationDbContext, opcja verbose wyświetla cały proces co się wykonuje. Dodaje ona system Identity do bazy danych.

Add-Migration InitialOfficial -Context CmentarzContext -verbose

Dodanie migracji bazującej na kontekście CmentarzContext, dodaje ona wszystkie modele oraz wymagane rzeczy potrzebne do funkcjonowania cmentarza.

W tej chwili możemy przejść do zaktualizowania bazy danych:

```
PM> Update-Database -Context ApplicationDbContext
Build started...
Build succeeded.
The EF Core tools version '3.1.1' is older than the
Done.
PM> Update-Database -Context CmentarzContext
Build started...
Build succeeded.
The EF Core tools version '3.1.1' is older than the
Done.
```

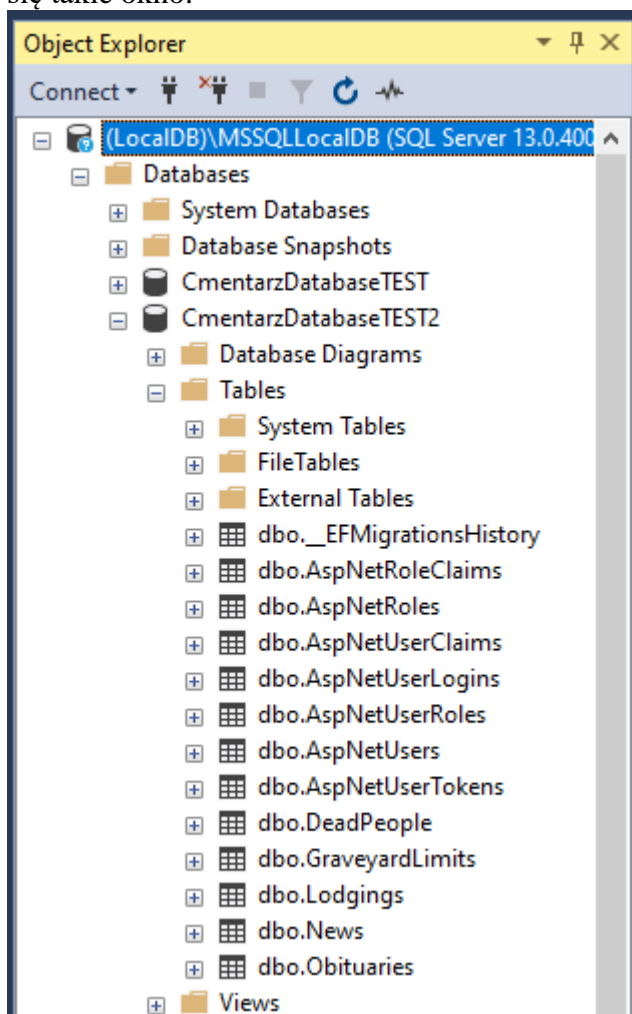
Update-Database -Context ApplicationDbContext -verbose

Update-Database -Context CmentarzContext -verbose

Verbose jest opcjonalne. Aktualizujemy nasz serwer naszymi migracjami. W bazie pojawi się również tabela dotycząca historii migracji.

W tej chwili udało nam się zaktualizować serwer z naszą bazą danych.

Wracamy do SQL Server Management Studio 18, przy zalogowaniu powinno nam pokazać się takie okno:

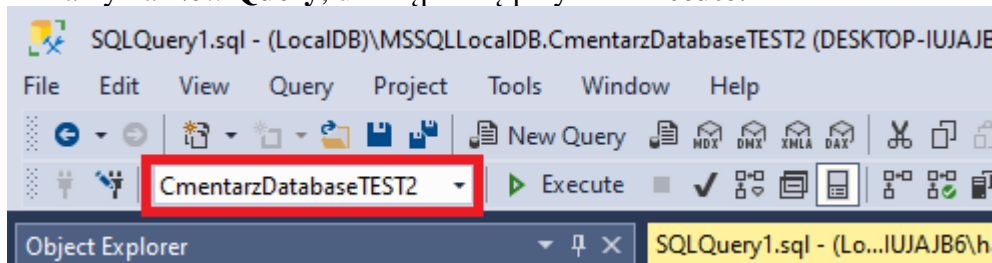


Jeśli tak jest to oznacza to, że wszystko przebiegło pomyślnie!

Na chwilę obecną nasza baza danych jest pusta. Trzeba ją zapęłnić informacjami przy pomocy pliku *transactSQLdatabase.sql*. Możemy go uruchomić bezpośrednio lub stworzyć Query.

UWAGA: Proszę się upewnić, czy jest wybrany **CmentarzDatabase** w czerwonym prostokącie.

Klikamy na **New Query**, udostępni się przycisk **Execute**:



Uruchamiamy plik *transactSQLdatabase.sql*, powinien się uruchomić automatycznie w programie SQL Server Management Studio 18. Jeśli tak nie jest, proszę skopiować treść i wkleić do *SQLQuery1.sql*:

```
transactSQLdatabas...IUJAJB6\hajci (55)  SQLQuery1.sql - (Lo...IUJAJB6\hajci (52)  DESKTOP-IUJAJB6\...- dbo.AspNetUsers
1  DELETE FROM [dbo].[DeadPeople]
2  DELETE FROM [dbo].[Obituaries]
3  DELETE FROM [dbo].[Lodgings]
4  DELETE FROM [dbo].[News]
5
6
7  SET IDENTITY_INSERT [dbo].[DeadPeople] ON
8  INSERT INTO [dbo].[DeadPeople] ([IdDeadPerson], [Name], [LodgingId], [DateOfBirth], [DateOfDeath]) VALUES
9  (0, N'Karol Wysocki', 0, '1911/12/08', '1953/04/29'),
10 (1, N'Janusz Wysocki', 0, '1935/07/08', '1953/05/29'),
11 (2, N'Kuba Wysocki', 0, '1906/02/15', '1953/06/15'),
12 (3, N'Błażej Wysocki', 0, '1939/09/20', '1954/07/29'),
13 (4, N'Mateusz Stępień', 1, '1904/05/02', '1954/02/12'),
14 (5, N'Oktawian Stępień', 1, '', '1955/01/31'),
15 (6, N'Dariusz Stępień', 1, '1904/04/05', '1955/01/02'),
16 (7, N'Krystian Zieliński', 2, '1911/03/15', '1956/08/31'),
17 (8, N'Dawid Zieliński', 2, '1916/01/16', '1956/09/31'),
18 (9, N'Wiesław Zieliński', 2, '', '1957/11/31'),
19 (10, N'Anastazy Czarnecki', 3, '1901/02/09', '1957/12/09'),
20 (11, N'Kryspin Czarnecki', 3, '1901/02/09', '1957/01/09'),
21 (12, N'Joanna Czarnecka', 3, '1901/02/09', '1958/10/09'),
22 (13, N'Kacper Czarnecki', 3, '', '1958/03/24'),
23 (14, N'Amadeusz Wiśniewski', 4, '1903/01/31', '1959/04/31'),
24 (15, N'Jędrzej Wiśniewski', 4, '1929/11/06', '1959/07/28'),
25 (16, N'Bożydar Wiśniewski', 4, '1930/10/12', '1961/03/28'),
26 (17, N'Ignacy Ziółkowska', 5, '1935/10/21', '1961/02/15'),
27 (18, N'Przemysław Ziółkowski', 5, '1911/12/08', '1962/05/26'),
28 (19, N'Krystian Ziółkowska', 5, '1950/02/15', '1962/07/26'),
29 (20, N'Emanuel Kubiak', 5, '1911/12/08', '1965/08/31'),
30 (21, N'Czesław Sikora', 6, '1912/12/08', '1965/09/26'),
31 (22, N'Kazimierz Sikora', 6, '1950/02/15', '1964/02/15'),
32 (23, N'Bruno Sikora', 6, '1939/02/15', '1964/08/31'),
33 (24, N'Michał Kaczmarczyk', 7, '1900/08/31', '1969/08/31'),
34 (25, N'Bartłomiej Kaczmarczyk', 7, '1950/05/01', '1969/08/31'),
35 (26, N'Rafał Kaczmarczyk', 7, '1920/04/05', '1969/08/21'),
36 (27, N'Julia Kaczmarczyk', 7, '1911/03/15', '1969/08/24'),
37 (28, N'Joachim Kwiatkowski', 8, '1916/01/16', '1961/08/24'),
38 (29, N'Piotr Kwiatkowski', 8, '1920/02/12', '1961/08/31'),
39 (30, N'Czesław Kwiatkowski', 8, '', '1961/05/14'),
40 (31, N'Marcel Jasiński', 9, '1907/12/09', '1961/08/01'),
41 (32, N'Borys Jasiński', 9, '1909/08/09', '1961/12/09'),
42 (33, N'...', 9, '1909/08/09', '1961/12/09'),
43 (34, N'...', 9, '1909/08/09', '1961/12/09'),
44 (35, N'...', 9, '1909/08/09', '1961/12/09'),
45 (36, N'...', 9, '1909/08/09', '1961/12/09'),
46 (37, N'...', 9, '1909/08/09', '1961/12/09'),
47 (38, N'...', 9, '1909/08/09', '1961/12/09'),
48 (39, N'...', 9, '1909/08/09', '1961/12/09'),
49 (40, N'...', 9, '1909/08/09', '1961/12/09'),
50 (41, N'...', 9, '1909/08/09', '1961/12/09'),
51 (42, N'...', 9, '1909/08/09', '1961/12/09'),
52 (43, N'...', 9, '1909/08/09', '1961/12/09'),
53 (44, N'...', 9, '1909/08/09', '1961/12/09'),
54 (45, N'...', 9, '1909/08/09', '1961/12/09'),
55 (46, N'...', 9, '1909/08/09', '1961/12/09'),
56 (47, N'...', 9, '1909/08/09', '1961/12/09'),
57 (48, N'...', 9, '1909/08/09', '1961/12/09'),
58 (49, N'...', 9, '1909/08/09', '1961/12/09'),
59 (50, N'...', 9, '1909/08/09', '1961/12/09'),
60 (51, N'...', 9, '1909/08/09', '1961/12/09'),
61 (52, N'...', 9, '1909/08/09', '1961/12/09'),
62 (53, N'...', 9, '1909/08/09', '1961/12/09'),
63 (54, N'...', 9, '1909/08/09', '1961/12/09'),
64 (55, N'...', 9, '1909/08/09', '1961/12/09'),
65 (56, N'...', 9, '1909/08/09', '1961/12/09'),
66 (57, N'...', 9, '1909/08/09', '1961/12/09'),
67 (58, N'...', 9, '1909/08/09', '1961/12/09'),
68 (59, N'...', 9, '1909/08/09', '1961/12/09'),
69 (60, N'...', 9, '1909/08/09', '1961/12/09'),
70 (61, N'...', 9, '1909/08/09', '1961/12/09'),
71 (62, N'...', 9, '1909/08/09', '1961/12/09'),
72 (63, N'...', 9, '1909/08/09', '1961/12/09'),
73 (64, N'...', 9, '1909/08/09', '1961/12/09'),
74 (65, N'...', 9, '1909/08/09', '1961/12/09'),
75 (66, N'...', 9, '1909/08/09', '1961/12/09'),
76 (67, N'...', 9, '1909/08/09', '1961/12/09'),
77 (68, N'...', 9, '1909/08/09', '1961/12/09'),
78 (69, N'...', 9, '1909/08/09', '1961/12/09'),
79 (70, N'...', 9, '1909/08/09', '1961/12/09'),
80 (71, N'...', 9, '1909/08/09', '1961/12/09'),
81 (72, N'...', 9, '1909/08/09', '1961/12/09'),
82 (73, N'...', 9, '1909/08/09', '1961/12/09'),
83 (74, N'...', 9, '1909/08/09', '1961/12/09'),
84 (75, N'...', 9, '1909/08/09', '1961/12/09'),
85 (76, N'...', 9, '1909/08/09', '1961/12/09'),
86 (77, N'...', 9, '1909/08/09', '1961/12/09'),
87 (78, N'...', 9, '1909/08/09', '1961/12/09'),
88 (79, N'...', 9, '1909/08/09', '1961/12/09'),
89 (80, N'...', 9, '1909/08/09', '1961/12/09'),
90 (81, N'...', 9, '1909/08/09', '1961/12/09'),
91 (82, N'...', 9, '1909/08/09', '1961/12/09'),
92 (83, N'...', 9, '1909/08/09', '1961/12/09'),
93 (84, N'...', 9, '1909/08/09', '1961/12/09'),
94 (85, N'...', 9, '1909/08/09', '1961/12/09'),
95 (86, N'...', 9, '1909/08/09', '1961/12/09'),
96 (87, N'...', 9, '1909/08/09', '1961/12/09'),
97 (88, N'...', 9, '1909/08/09', '1961/12/09'),
98 (89, N'...', 9, '1909/08/09', '1961/12/09'),
99 (90, N'...', 9, '1909/08/09', '1961/12/09'),
100 (91, N'...', 9, '1909/08/09', '1961/12/09'),
101 (92, N'...', 9, '1909/08/09', '1961/12/09'),
102 (93, N'...', 9, '1909/08/09', '1961/12/09'),
103 (94, N'...', 9, '1909/08/09', '1961/12/09'),
104 (95, N'...', 9, '1909/08/09', '1961/12/09'),
105 (96, N'...', 9, '1909/08/09', '1961/12/09'),
106 (97, N'...', 9, '1909/08/09', '1961/12/09'),
107 (98, N'...', 9, '1909/08/09', '1961/12/09'),
108 (99, N'...', 9, '1909/08/09', '1961/12/09'),
109 (100, N'...', 9, '1909/08/09', '1961/12/09'),
110 (101, N'...', 9, '1909/08/09', '1961/12/09'),
111 (102, N'...', 9, '1909/08/09', '1961/12/09'),
112 (103, N'...', 9, '1909/08/09', '1961/12/09'),
113 (104, N'...', 9, '1909/08/09', '1961/12/09'),
114 (105, N'...', 9, '1909/08/09', '1961/12/09'),
115 (106, N'...', 9, '1909/08/09', '1961/12/09'),
116 (107, N'...', 9, '1909/08/09', '1961/12/09'),
117 (108, N'...', 9, '1909/08/09', '1961/12/09'),
118 (109, N'...', 9, '1909/08/09', '1961/12/09'),
119 (110, N'...', 9, '1909/08/09', '1961/12/09'),
120 (111, N'...', 9, '1909/08/09', '1961/12/09'),
121 (112, N'...', 9, '1909/08/09', '1961/12/09'),
122 (113, N'...', 9, '1909/08/09', '1961/12/09'),
123 (114, N'...', 9, '1909/08/09', '1961/12/09'),
124 (115, N'...', 9, '1909/08/09', '1961/12/09'),
125 (116, N'...', 9, '1909/08/09', '1961/12/09'),
126 (117, N'...', 9, '1909/08/09', '1961/12/09'),
127 (118, N'...', 9, '1909/08/09', '1961/12/09'),
128 (119, N'...', 9, '1909/08/09', '1961/12/09'),
129 (120, N'...', 9, '1909/08/09', '1961/12/09'),
130 (121, N'...', 9, '1909/08/09', '1961/12/09'),
131 (122, N'...', 9, '1909/08/09', '1961/12/09'),
132 (123, N'...', 9, '1909/08/09', '1961/12/09'),
133 (124, N'...', 9, '1909/08/09', '1961/12/09'),
134 (125, N'...', 9, '1909/08/09', '1961/12/09'),
135 (126, N'...', 9, '1909/08/09', '1961/12/09'),
136 (127, N'...', 9, '1909/08/09', '1961/12/09'),
137 (128, N'...', 9, '1909/08/09', '1961/12/09'),
138 (129, N'...', 9, '1909/08/09', '1961/12/09'),
139 (130, N'...', 9, '1909/08/09', '1961/12/09'),
140 (131, N'...', 9, '1909/08/09', '1961/12/09'),
141 (132, N'...', 9, '1909/08/09', '1961/12/09'),
142 (133, N'...', 9, '1909/08/09', '1961/12/09'),
143 (134, N'...', 9, '1909/08/09', '1961/12/09'),
144 (135, N'...', 9, '1909/08/09', '1961/12/09'),
145 (136, N'...', 9, '1909/08/09', '1961/12/09'),
146 (137, N'...', 9, '1909/08/09', '1961/12/09'),
147 (138, N'...', 9, '1909/08/09', '1961/12/09'),
148 (139, N'...', 9, '1909/08/09', '1961/12/09'),
149 (140, N'...', 9, '1909/08/09', '1961/12/09'),
150 (141, N'...', 9, '1909/08/09', '1961/12/09'),
151 (142, N'...', 9, '1909/08/09', '1961/12/09'),
152 (143, N'...', 9, '1909/08/09', '1961/12/09'),
153 (144, N'...', 9, '1909/08/09', '1961/12/09'),
154 (145, N'...', 9, '1909/08/09', '1961/12/09'),
155 (146, N'...', 9, '1909/08/09', '1961/12/09'),
156 (147, N'...', 9, '1909/08/09', '1961/12/09'),
157 (148, N'...', 9, '1909/08/09', '1961/12/09'),
158 (149, N'...', 9, '1909/08/09', '1961/12/09'),
159 (150, N'...', 9, '1909/08/09', '1961/12/09'),
160 (151, N'...', 9, '1909/08/09', '1961/12/09'),
161 (152, N'...', 9, '1909/08/09', '1961/12/09'),
162 (153, N'...', 9, '1909/08/09', '1961/12/09'),
163 (154, N'...', 9, '1909/08/09', '1961/12/09'),
164 (155, N'...', 9, '1909/08/09', '1961/12/09'),
165 (156, N'...', 9, '1909/08/09', '1961/12/09'),
166 (157, N'...', 9, '1909/08/09', '1961/12/09'),
167 (158, N'...', 9, '1909/08/09', '1961/12/09'),
168 (159, N'...', 9, '1909/08/09', '1961/12/09'),
169 (160, N'...', 9, '1909/08/09', '1961/12/09'),
170 (161, N'...', 9, '1909/08/09', '1961/12/09'),
171 (162, N'...', 9, '1909/08/09', '1961/12/09'),
172 (163, N'...', 9, '1909/08/09', '1961/12/09'),
173 (164, N'...', 9, '1909/08/09', '1961/12/09'),
174 (165, N'...', 9, '1909/08/09', '1961/12/09'),
175 (166, N'...', 9, '1909/08/09', '1961/12/09'),
176 (167, N'...', 9, '1909/08/09', '1961/12/09'),
177 (168, N'...', 9, '1909/08/09', '1961/12/09'),
178 (169, N'...', 9, '1909/08/09', '1961/12/09'),
179 (170, N'...', 9, '1909/08/09', '1961/12/09'),
180 (171, N'...', 9, '1909/08/09', '1961/12/09'),
181 (172, N'...', 9, '1909/08/09', '1961/12/09'),
182 (173, N'...', 9, '1909/08/09', '1961/12/09'),
183 (174, N'...', 9, '1909/08/09', '1961/12/09'),
184 (175, N'...', 9, '1909/08/09', '1961/12/09'),
185 (176, N'...', 9, '1909/08/09', '1961/12/09'),
186 (177, N'...', 9, '1909/08/09', '1961/12/09'),
187 (178, N'...', 9, '1909/08/09', '1961/12/09'),
188 (179, N'...', 9, '1909/08/09', '1961/12/09'),
189 (180, N'...', 9, '1909/08/09', '1961/12/09'),
190 (181, N'...', 9, '1909/08/09', '1961/12/09'),
191 (182, N'...', 9, '1909/08/09', '1961/12/09'),
192 (183, N'...', 9, '1909/08/09', '1961/12/09'),
193 (184, N'...', 9, '1909/08/09', '1961/12/09'),
194 (185, N'...', 9, '1909/08/09', '1961/12/09'),
195 (186, N'...', 9, '1909/08/09', '1961/12/09'),
196 (187, N'...', 9, '1909/08/09', '1961/12/09'),
197 (188, N'...', 9, '1909/08/09', '1961/12/09'),
198 (189, N'...', 9, '1909/08/09', '1961/12/09'),
199 (190, N'...', 9, '1909/08/09', '1961/12/09'),
200 (191, N'...', 9, '1909/08/09', '1961/12/09'),
201 (192, N'...', 9, '1909/08/09', '1961/12/09'),
202 (193, N'...', 9, '1909/08/09', '1961/12/09'),
203 (194, N'...', 9, '1909/08/09', '1961/12/09'),
204 (195, N'...', 9, '1909/08/09', '1961/12/09'),
205 (196, N'...', 9, '1909/08/09', '1961/12/09'),
206 (197, N'...', 9, '1909/08/09', '1961/12/09'),
207 (198, N'...', 9, '1909/08/09', '1961/12/09'),
208 (199, N'...', 9, '1909/08/09', '1961/12/09'),
209 (200, N'...', 9, '1909/08/09', '1961/12/09'),
210 (201, N'...', 9, '1909/08/09', '1961/12/09'),
211 (202, N'...', 9, '1909/08/09', '1961/12/09'),
212 (203, N'...', 9, '1909/08/09', '1961/12/09'),
213 (204, N'...', 9, '1909/08/09', '1961/12/09'),
214 (205, N'...', 9, '1909/08/09', '1961/12/09'),
215 (206, N'...', 9, '1909/08/09', '1961/12/09'),
216 (207, N'...', 9, '1909/08/09', '1961/12/09'),
217 (208, N'...', 9, '1909/08/09', '1961/12/09'),
218 (209, N'...', 9, '1909/08/09', '1961/12/09'),
219 (210, N'...', 9, '1909/08/09', '1961/12/09'),
220 (211, N'...', 9, '1909/08/09', '1961/12/09'),
221 (212, N'...', 9, '1909/08/09', '1961/12/09'),
222 (213, N'...', 9, '1909/08/09', '1961/12/09'),
223 (214, N'...', 9, '1909/08/09', '1961/12/09'),
224 (215, N'...', 9, '1909/08/09', '1961/12/09'),
225 (216, N'...', 9, '1909/08/09', '1961/12/09'),
226 (217, N'...', 9, '1909/08/09', '1961/12/09'),
227 (218, N'...', 9, '1909/08/09', '1961/12/09'),
228 (219, N'...', 9, '1909/08/09', '1961/12/09'),
229 (220, N'...', 9, '1909/08/09', '1961/12/09'),
230 (221, N'...', 9, '1909/08/09', '1961/12/09'),
231 (222, N'...', 9, '1909/08/09', '1961/12/09'),
232 (223, N'...', 9, '1909/08/09', '1961/12/09'),
233 (224, N'...', 9, '1909/08/09', '1961/12/09'),
234 (225, N'...', 9, '1909/08/09', '1961/12/09'),
235 (226, N'...', 9, '1909/08/09', '1961/12/09'),
236 (227, N'...', 9, '1909/08/09', '1961/12/09'),
237 (228, N'...', 9, '1909/08/09', '1961/12/09'),
238 (229, N'...', 9, '1909/08/09', '1961/12/09'),
239 (230, N'...', 9, '1909/08/09', '1961/12/09'),
240 (231, N'...', 9, '1909/08/09', '1961/12/09'),
241 (232, N'...', 9, '1909/08/09', '1961/12/09'),
242 (233, N'...', 9, '1909/08/09', '1961/12/09'),
243 (234, N'...', 9, '1909/08/09', '1961/12/09'),
244 (235, N'...', 9, '1909/08/09', '1961/12/09'),
245 (236, N'...', 9, '1909/08/09', '1961/12/09'),
246 (237, N'...', 9, '1909/08/09', '1961/12/09'),
247 (238, N'...', 9, '1909/08/09', '1961/12/09'),
248 (239, N'...', 9, '1909/08/09', '1961/12/09'),
249 (240, N'...', 9, '1909/08/09', '1961/12/09'),
250 (241, N'...', 9, '1909/08/09', '1961/12/09'),
251 (242, N'...', 9, '1909/08/09', '1961/12/09'),
252 (243, N'...', 9, '1909/08/09', '1961/12/09'),
253 (244, N'...', 9, '1909/08/09', '1961/12/09'),
254 (245, N'...', 9, '1909/08/09', '1961/12/09'),
255 (246, N'...', 9, '1909/08/09', '1961/12/09'),
256 (247, N'...', 9, '1909/08/09', '1961/12/09'),
257 (248, N'...', 9, '1909/08/09', '1961/12/09'),
258 (249, N'...', 9, '1909/08/09', '1961/12/09'),
259 (250, N'...', 9, '1909/08/09', '1961/12/09'),
260 (251, N'...', 9, '1909/08/09', '1961/12/09'),
261 (252, N'...', 9, '1909/08/09', '1961/12/09'),
262 (253, N'...', 9, '1909/08/09', '1961/12/09'),
263 (254, N'...', 9, '1909/08/09', '1961/12/09'),
264 (255, N'...', 9, '1909/08/09', '1961/12/09'),
265 (256, N'...', 9, '1909/08/09', '1961/12/09'),
266 (257, N'...', 9, '1909/08/09', '1961/12/09'),
267 (258, N'...', 9, '1909/08/09', '1961/12/09'),
268 (259, N'...', 9, '1909/08/09', '1961/12/09'),
269 (260, N'...', 9, '1909/08/09', '1961/12/09'),
270 (261, N'...', 9, '1909/08/09', '1961/12/09'),
271 (262, N'...', 9, '1909/08/09', '1961/12/09'),
272 (263, N'...', 9, '1909/08/09', '1961/12/09'),
273 (264, N'...', 9, '1909/08/09', '1961/12/09'),
274 (265, N'...', 9, '1909/08/09', '1961/12/09'),
275 (266, N'...', 9, '1909/08/09', '1961/12/09'),
276 (267, N'...', 9, '1909/08/09', '1961/12/09'),
277 (268, N'...', 9, '1909/08/09', '1961/12/09'),
278 (269, N'...', 9, '1909/08/09', '1961/12/09'),
279 (270, N'...', 9, '1909/08/09', '1961/12/09'),
280 (271, N'...', 9, '1909/08/09', '1961/12/09'),
281 (272, N'...', 9, '1909/08/09', '1961/12/09'),
282 (273, N'...', 9, '1909/08/09', '1961/12/09'),
283 (274, N'...', 9, '1909/08/09', '1961/12/09'),
284 (275, N'...', 9, '1909/08/09', '1961/12/09'),
285 (276, N'...', 9, '1909/08/09', '1961/12/09'),
286 (277, N'...', 9, '1909/08/09', '1961/12/09'),
287 (278, N'...', 9, '1909/08/09', '1961/12/09'),
288 (279, N'...', 9, '1909/08/09', '1961/12/09'),
289 (280, N'...', 9, '1909/08/09', '1961/12/09'),
290 (281, N'...', 9, '1909/08/09', '1961/12/09'),
291 (282, N'...', 9, '1909/08/09', '1961/12/09'),
292 (283, N'...', 9, '1909/08/09', '1961/12/09'),
293 (284, N'...', 9, '1909/08/09', '1961/12/09'),
294 (285, N'...', 9, '1909/08/09', '1961/12/09'),
295 (286, N'...', 9, '1909/08/09', '1961/12/09'),
296 (287, N'...', 9, '1909/08/09', '1961/12/09'),
297 (288, N'...', 9, '1909/08/09', '1961/12/09'),
298 (289, N'...', 9, '1909/08/09', '1961/12/09'),
299 (290, N'...', 9, '1909/08/09', '1961/12/09'),
300 (291, N'...', 9, '1909/08/09', '1961/12/09'),
301 (292, N'...', 9, '1909/08/09', '1961/12/09'),
302 (293, N'...', 9, '1909/08/09', '1961/12/09'),
303 (294, N'...', 9, '1909/08/09', '1961/12/09'),
304 (295, N'...', 9, '1909/08/09', '1961/12/09'),
305 (296, N'...', 9, '1909/08/09', '19
```

Gdy to już zostanie zrealizowane, można kliknąć przycisk **Execute** i wprowadzić dane do bazy, uzyskamy taką odpowiedź zwrotną:

```

Messages

(0 rows affected)

(0 rows affected)

(0 rows affected)

(0 rows affected)

(201 rows affected)

(10 rows affected)

(100 rows affected)

(5 rows affected)

Completion time: 2021-01-13T15:49:26.2067390+01:00

```

Gdy sprawdzimy nasze tabele, zobaczymy, że zostały zapełnione danymi, przykład:

DESKTOP-IUJAJB6\L...- dbo.DeadPeople X transactSQLdatabas...IUJAJB6\hajci (55) SQLQ					
	IdDeadPerson	Name	DateOfBirth	DateOfDeath	LodgingId
▶	0	Karol Wysocki	1911/12/08	1953/04/29	0
	1	Janusz Wysocki	1935/07/08	1953/05/29	0
	2	Kuba Wysocki	1906/02/15	1953/06/15	0
	3	Błażej Wysocki	1939/09/20	1954/07/29	0
	4	Mateusz Stępień	1904/05/02	1954/02/12	1
	5	Oktawian Stępień		1955/01/31	1
	6	Dariusz Stępień	1904/04/05	1955/01/02	1
	7	Krystian Zieliński	1911/03/15	1956/08/31	2
	8	Dawid Zieliński	1916/01/16	1956/09/31	2
	9	Włodzisław Zieliński		1957/11/31	2

UWAGA: Następnie należy tymczasowo wyłączyć **Visual Studio**.

Proszę zainstalować node.js 14.15.4 LTS na komputerze, link zamieszczony wyżej.



Sprawdzamy w wierszu poleceń czy instalacja się powiodła komendą sprawdzającą wersję:

npm -v

```

C:\Users\hajci>npm -v
6.14.9

```

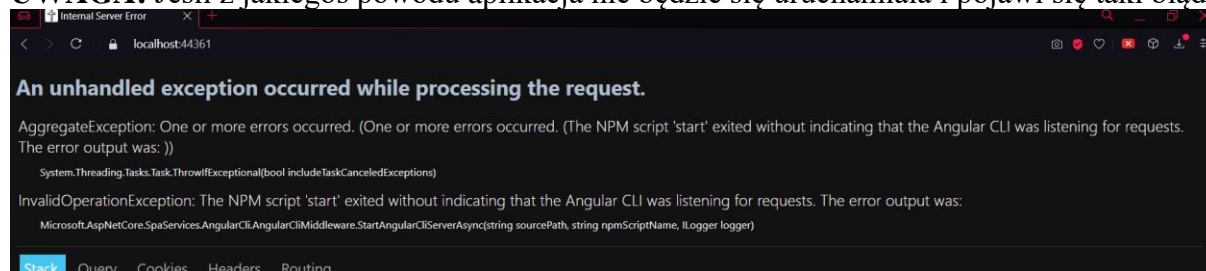
Po zainstalowaniu node.js pora na instalację **Angular CLI**. Minimalna wersja której wymaga projekt to Angular 8, jednakże wystarczy zainstalować domyślnie przez wiersz polecenia.

Proszę wpisać do wiersza poleceń:

npm install -g @angular/cli

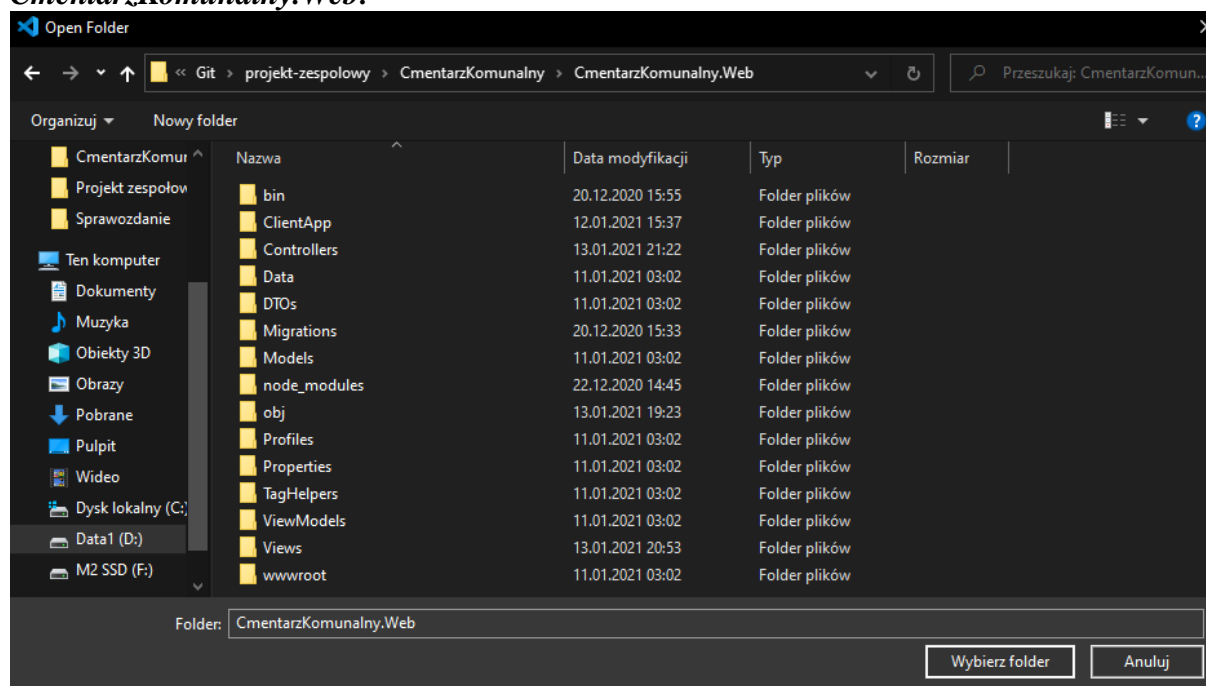
Po uczynieniu tej procedury Angular CLI powinno zainstalować się na komputerze.

UWAGA: Jeśli z jakiegoś powodu aplikacja nie będzie się uruchamiała i pojawi się taki błąd:

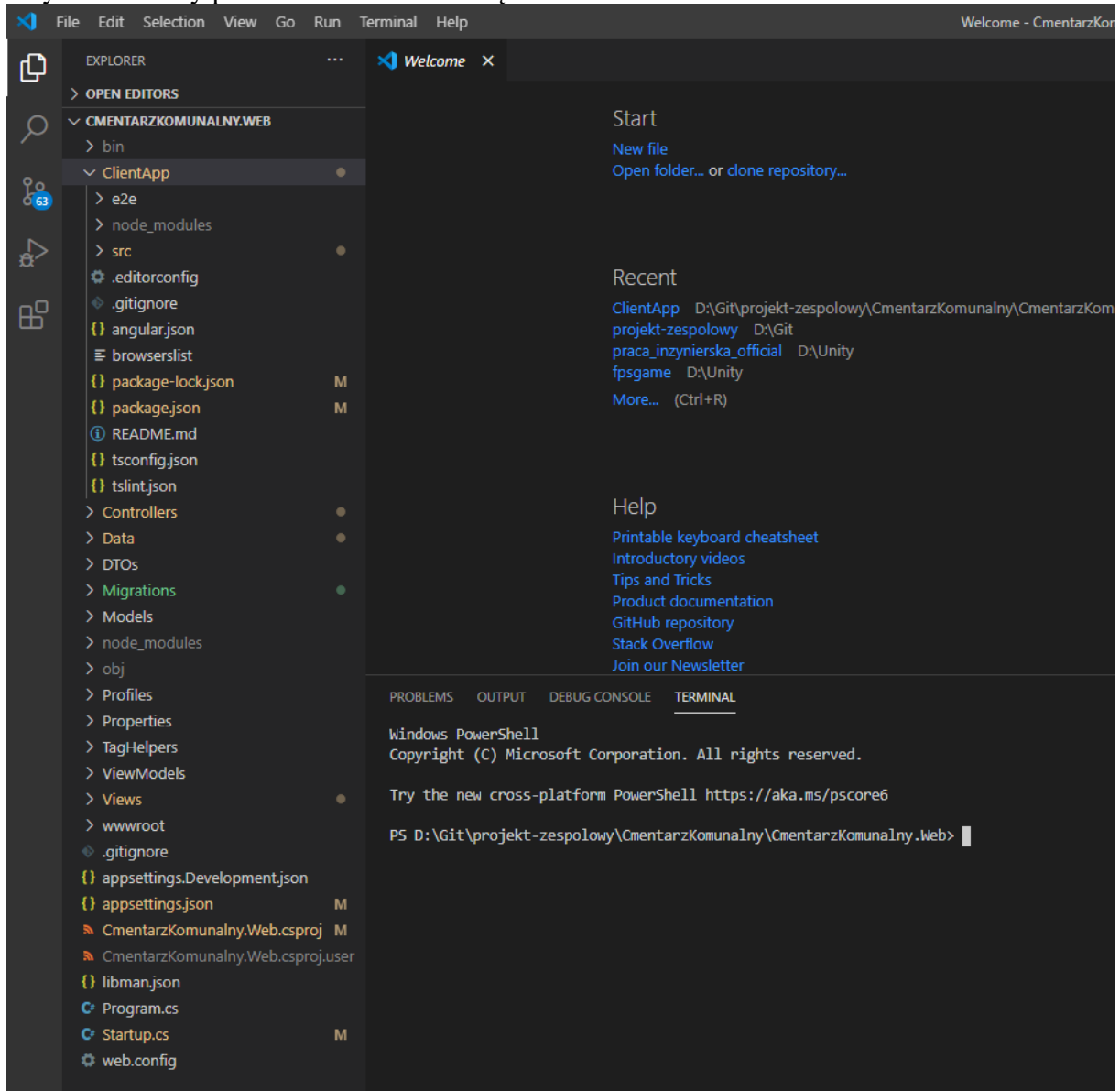


To inną metodą instalacji jest bezpośrednio w folderze Angulara aplikacji. Jeśli dojdzie do takiej sytuacji to najłatwiej uruchomić projekt przez program **Visual Studio CODE**. Choć taka sytuacja nie powinna mieć miejsca bo Angulara instalujemy bezpośrednio na naszym komputerze.

Otwieramy zakładkę **File => Open Folder**. Wybieramy ścieżkę tak by otworzyć **CmentarzKomunalny.Web**:



Gdy tak zrobimy powinien ukazać nam się taki widok:

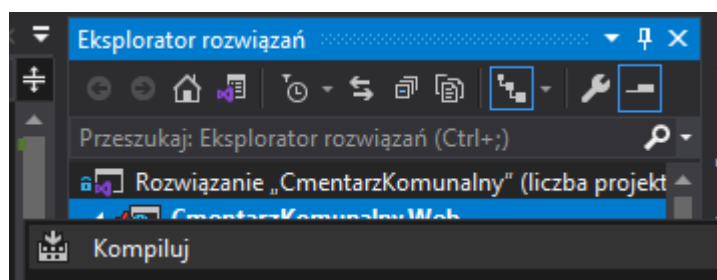


Otwieramy zakładkę **Terminal** i uruchamiamy nowy terminal.

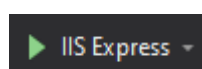
Ukaże nam się okno Terminal widoczne na poprzednim obrazku. Jedyne co trzeba jeszcze zrobić to przejść do folderu ClientApp komendą **cd ClientApp**. W tej chwili pozostaje tylko wpisanie komendy do instalacji Angulara: **npm -g install @angular/cli**

W tym momencie wszystko powinno przejść pomyślnie.

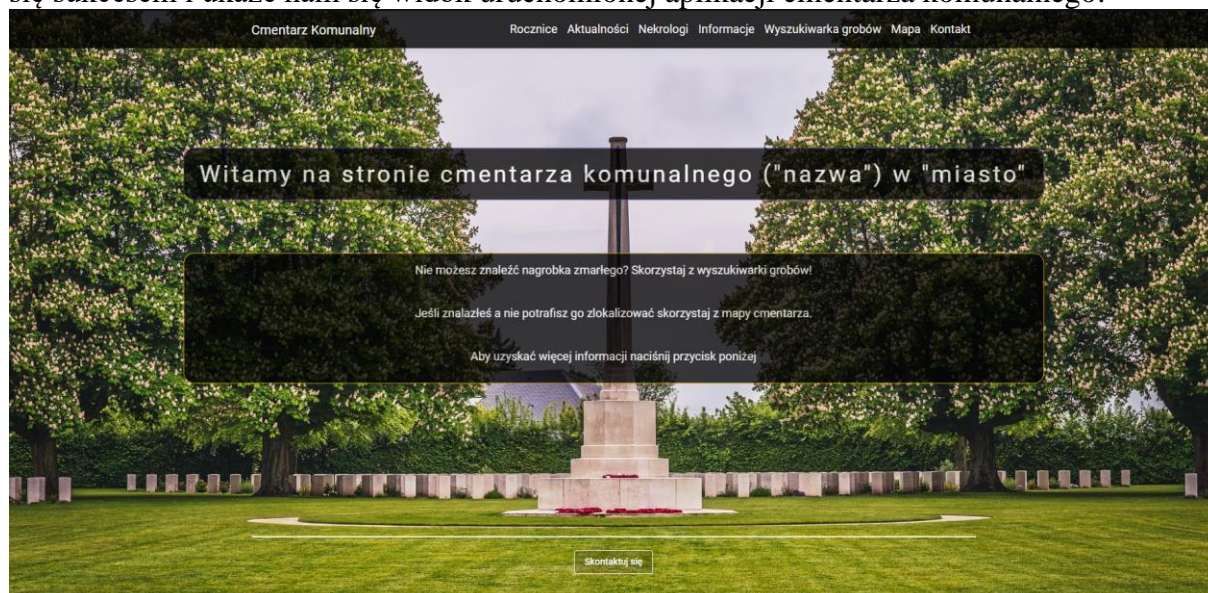
W tej chwili możemy skompilować aplikację oraz uruchomić ją przez **IIS Express**:
Klikamy prawym przyciskiem myszy na projekt **CmentarzKomunalny.Web** lub korzystamy ze skrótu **CTRL + SHIFT + B** i kompilujemy:



Po kompilacji powinniśmy móc uruchomić naszą aplikację przez IIS Express:



Jeśli wszystkie kroki zostały przeprowadzone to uruchomienie aplikacji powinno zakończyć się sukcesem i ukaże nam się widok uruchomionej aplikacji cmentarza komunalnego:



Jeśli aplikacja jest uruchamiana pierwszy raz to może pokazać się okno logowania Identity. Jest ono kwestią rozwojową aplikacji i zabezpieczającą w trakcie pierwszego uruchamiania:

CmentarzKomunalny.Web

Log in

Use a local account to log in.

Email

Password

☐ Remember me?

Log in

Use another service to log in.

There are no external authentication services configured. See [this article](#) for details on setting up this ASP.NET application to support logging in via external services.

Dane do logowania się w aplikacji przekazane w pliku SQL.

ADRESY WIDOKÓW LOGOWANIA SIĘ:

~/Account/Login

~/Identity/Account/Login

ADMINISTRATOR:

Login: administrator@administrator.pl

Hasło: administrator1

PRACOWNIK:

Login: pracownik1@pracownik.pl / pracownik2@pracownik.pl

Hasło: pracownik1

ZABLOKOWANY PRACOWNIK:

Login: pracownik3@pracownik.pl

Hasło: pracownik1

Podsumowanie

Aplikacja „Cmentarz Komunalny” organizuje informacje oraz dane wybranego cmentarza w mieście wybranym przez klienta. Ma ona służyć do przejrzenia rzeczy takich jak: *nekrologi, aktualności, wyszukać zmarłych i miejsca ich pochówku na mapie*.

Aplikacja pozwala na wyszukiwanie zmarłych po konkretnych informacjach, wyświetla rocznice bazujące na danym dniu, wyświetla aktualności oraz nekrologi mieszczące się w bazie danych. Nekrologi są osobnym bytem od zmarłych, głównie dlatego, że nie wszyscy korzystający z takich aplikacji a posiadający zmarłych na danym cmentarzu chcieliby, by ukazane były nekrologi z ich bliskimi.

Aplikacja została stworzona z myślą o użytkownikach w różnym wieku. Starano się utrzymać jednolitość wizerunkową oraz funkcjonalną w taki sposób, aby korzystanie przechodziło gładko oraz zrozumiale.

Kwestia przyszłości rozwoju aplikacji:

W przyszłości i rozwinięciu aplikacji do pełnej implementacji – ma ona za zadanie umożliwić klientom rezerwację grobu dla zmarłej bliskiej mu osoby, aplikacja będzie posiadała możliwość przeprowadzenia użytkownika „za rękę” z całym procesem, od wypełnienia formularza dotyczącego szczegółów pochówku, po zapłacenie za wszystkie aspekty.

Będzie udostępniona możliwość komunikacji z domem pogrzebowym który zajmie się wszystkim na miejscu, możliwość wyszukiwania grobu danej zmarłej osoby bezpośrednio na stronie.