- Opis problemu i opis funkcjonalności udostępnianej przez API
 - Celem API jest umożliwienie obsługi struktury organizacyjnej firmy będącej strukturą hierarchiczną. Elementami w hierarchii są pracownicy wraz z ich danymi personalnymi.
- Opis typów danych oraz metod (funkcji) udostępnionych w ramach API
 - Typy tabeli przechowującej dane:
 - Employee
 - Level hierarchyid
 - Name nvarchar(30)
 - Position nvarchar(30)
 - Salary Int
 - Udostępnione funkcje:
 - AddEmployee
 - RemoveEmployee
 - RemoveAllEmployees
 - GetEmployee
 - GetEmployeeWithChildren
 - GetAllEmployees
 - GetMaxSalary
 - GetAverageSalary
 - GetSumSalary
- Opis implementacji
 - Api udostępnia procedury wykonujące operacje na rekordach tabeli pracowników (Employees) w bazie danych. Jest ono zaimplementowane w postaci klasy w języku C# wykonującej zapytania SQL do bazy danych uruchamiających procedury wykonujące żądane operacje. W procedurach wykonywane są instrukcje SQL korzystające z dobrodziejstw użytych typów danych jak np. hierarchyid
 - Każda z tych funkcji jest dostępna z poziomu interaktywnej aplikacji konsolowej przyjmującej input od użytkownika na temat tego, co chce zrobić.
 - o Przykłady użycia:

C:\Users\admin\Source\Repos\MichalLeszczynski\CompanyApi\bin\Company\netcoreapp3.1\CompanyApi.exe

```
Welcome to company manager.
Available coomands:

1. Get all employees
2. Get employee by his position in hierarchy
3. Get employee with their subordinates by his position in hierarchy
4. Add new employee
5. Remove employee
6. Add mock data to the databse
7. Remove all employees
8. Get salary statistics
9. Exit the app
Your choice:
```

rys. 1. Ekran powitalny aplikacji

C:\Users\admin\Source\Repos\MichalLeszczynski\CompanyApi\bin\Company\netcoreapp3.1\CompanyApi.exe

```
Welcome to company manager.
Available coomands:
1. Get all employees

    Get employee by his position in hierarchy
    Get employee with their subordinates by his position in hierarchy

4. Add new employee
5. Remove employee
6. Add mock data to the databse

    Remove all employees
    Get salary statistics

9. Exit the app
Your choice:
Your command: 3
Getting employee with their subordinates
Pass the hierarchyid of the employee:
/3/
        Hilda Hall 12000 Vice President Marketing
/3/1/
                                   Sales Manager
        Selena Gomez
                         8000
                        7000
/3/2/
      David Diaz
                                   Advertising Manager
/3/2/1/ Michael Parker 3000
                                   Account Executive
Press any key to continue...
```

rys. 2. Pobranie pracownika z jego podwładnymi

```
Available statistics:

1. Maximum salary

2. Average salary

3. Sum salary

Your choice:

2

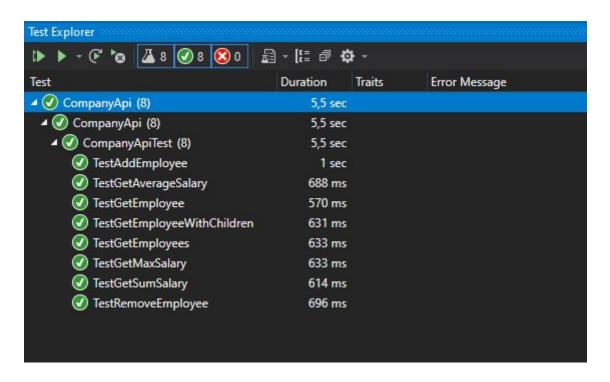
Your command: 2

Average salary: 9000

Press any key to continue...
```

rys. 3. Pobranie średniej wypłaty w firmie

- Prezentacja przeprowadzonych testów jednostkowych
 - TestGetEmployees testuje pobieranie wszystkich pracowników z bazy sprawdza, czy ich liczba jest poprawna
 - TestGetEmployee testuje pobieranie informacji o pracowniku na konkretnej pozycji - sprawdza, czy zostały pobrane poprawne dane
 - TestGetEmployeeWithChildren testuje pobieranie pracownika wraz ze wszystkimi jego podwładnymi - sprawdza, czy liczba pobranych pracowników jest poprawna
 - TestAddEmployee testuje dodawanie pracownika sprawdzenie, czy liczba pracowników się zwiększyła, oraz pobranie danych i sprawdzenie ich poprawności
 - TestRemoveEmployee testuje usuwanie pracownika sprawdzenie, czy liczba pracowników się zmniejszyła
 - TestGetMaxSalary testuje pobieranie informacji o maksymalnej wypłacie
 - TestGetAverageSalary testuje pobieranie informacji o średniej wypłacie
 - TestGetSumSalary testuje pobieranie informacji o sumie wypłat



rys. 4. Przykład uruchomienia testów testujących operacje korzystające z API

Podsumowanie, wnioski

Typ hierarchyid udostępnia wiele funkcji, które są przydatne i normalnie przy rozwiązywaniu tego typu problemów musiałyby być dodatkowo zaimplementowane, jednak wsparcie C# dla hierarchyid nie jest kompletne, i wynikło z tego wiele problemów podczas implementacji, które wymagały swoich walkaround'ów jak np. przesyłanie do bazy hierarchyid jako zmienna tekstowa, aby została przetworzona w bazie - na szczęście we wszystkich innych miejscach było możliwe użycie tego typu we właściwy sposób.

Literatura:

- https://www.plukasiewicz.net/Artykuly/ADO_NET
- https://docs.microsoft.com/pl-pl/aspnet/core/fundamentals/?view=aspnetcore-3.1&tabs=windows
- https://softwarehut.com/blog/tech/hierarchyid-entity-framework
- https://kariera.future-processing.pl/blog/hierarchy-in-the-entity-framework-6-wi th-the-hierarchyid-type/
- https://www.c-sharpcorner.com/article/a-basic-introduction-of-unit-test-for-beginners/
- Kod żródłowy wszystkich skryptów T-SQL wykorzystanych w projekcie (np. tworzenie testowej bazy danych) oraz aplikacji w C#.

- o W załączonych plikach .sql:
 - create_database.sql tworzenie bazy danych
 - create_procedures.sql tworzenie tabeli i procedur
 - delete_procedures.sql usuwanie tabeli i procedur