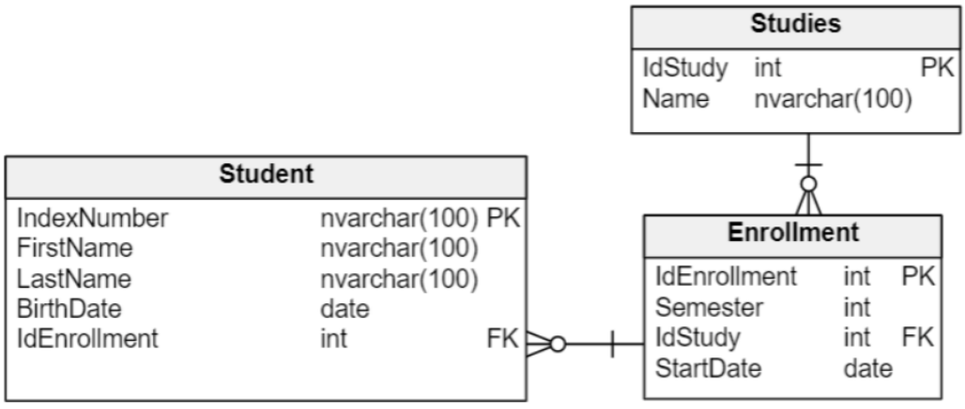
# APBD - Ćwiczenie 6

pgago@pja.edu.pl

6 kwietnia 2020

## 1 Wstęp

W niniejszych ćwiczeniach usprawnimy nieco nasze API. Dodamy dokumentację i dwa middleware'y. Pracujemy na kodzie z poprzednich zajęć.



# 1. Dodanie middleware - sprawdzanie indeksu

W tym zadaniu musimy dodać własny middleware. Pamiętamy, że dodawania middleware i podpinaniu pod tzw. "pipelineźwiązany z przetwarzaniem żądań HTTP odbywa się w klasie Startup.cs i metodzie Configure.
Naszym zadaniem jest dodanie middleware'u, który sprawdzi czy wszystkie żądania przychodzące do naszego API pochodzą od studentów. Z tego powodu oczekujemy, że każde żądania będzie posiadało numer studenta w nagłówku żądania zapisanego pod kluczem "Index".
Ponadto nasz middleware powinien sprawdzić czy taki student faktycznie istnieje w bazie danych. Poniżej mamy fragment kodu do uzupełnienia. Pamiętaj, że kolejność dodawanie middleware do pipeline'a jest ważna. Uwierzytelnienie powinna zachodzić na początku.
W przypadku kiedy nagłówek Index nie znajduje się w żądaniu - zwracamy błąd 401.

```
app.Use(async (context, next) =>
{
    //Custom code
    await next();
});
```

## 2. Dodanie middleware - logowanie żądań do pliku

W tym wypadku chcielibyśmy stworzyć dodatkowy middleware, który będzie logować podstawowe informacje związane ze wszystkimi przychodzącymi do nas żądaniami. Tym razem stworzymy middleware w osobnej klasie. W tym celu warto dodać folder o nazwie Middlewares. Następnie wewnątrz niego umieścić klasę podobno do poniższej. W metodzie InvokeAsync umieszczamy nasz kod. Chcemy zapisywać do pliku o nazwie requestsLog.txt następujące informacje:

1. Metodę HTTP (GET, POST, itd.)
2. Ścieżkę na którą zostało wysłane żądanie (/api/students)
3. Dodanie logowania ciała żądania HTTP (np. wysłany JSON)
4. Zapisywanie informacji z Query string (?name=Kowalski)

```
public class LoggingMiddleware
{
    private readonly RequestDelegate _next;

    public LoggingMiddleware(RequestDelegate next)
    {
        _next = next;
    }

    public async Task InvokeAsync(HttpContext httpContext)
    {
        //Our code
        await _next(httpContext);
    }
}
```

Następnie w klasie Startup.cs dodajemy uruchomienie metody UseMiddleware. W celu zarejestrowania middleware dodajemy w klasie Startup.cs -> COnfigure

```
app.UseMiddleware<LoggingMiddleware>();
```