# Temat projektu: Simple class register

1. Użyte technologie: asp.net core, jquery, bootstrap, entity framework core, asp.net core identity
2. Opis zasady działania projektu:

Projekt został zrealizowany na podstawie wzorca MVC z uwzględnieniem n-tier architecture, czyli dodana została dodatkowa warstwa serwisów w celu odciążenia kontrolerów. Nasz projekt zawiera główny folder „Services”, który dzieli się na 3 kolejne pod foldery. Każdy folder reprezentuje serwisy dla różnych wariantów. Folder „AccontServices” powiązany jest z funkcjonalnością rejestracji i logowania. Folder „StudentServices” powiązany jest z funkcjonalnościami takimi jak, wybieranie klasy przez studenta oraz przeglądanie wystawionych ocen. Natomiast folder o nazwie „TeacherServices” reprezentują odpowiednio funkcjonalności takie jak wybieranie przedmiotów przez nauczyciela, wybieranie klas i przedmiotu, którego dany nauczyciel naucza, oraz wystawianie ocen studentom. W dużym skrócie serwisy zawierają całą logikę związaną z zapisywaniem danych do bazy oraz odczytywanie odpowiednich danych. Idąc dalej tym tropem, w folderze controllers istnieją 4 kontrolery odpowiedzialne za przesyłaniem danych do widoku oraz odbieranie danych z widoku. 3 kontrolery mają swoje serwisy, które wstrzykiwane są za pomocą wbudowanego kontenera dependency injection. Niektóre akcje w kontrolerach korzystają z viewModeli, czyli kilku modeli, które przekazywane są do widoku za pomocą jednej klasy, dzięki temu możemy skorzystać ze wszystkich potrzebnych kolekcji oraz pojedynczych instancji. Dodatkowo każdy viewModel został nazwany w taki sposób, żeby skojarzyć dany viewModel z akcją w kontrolerze oraz serwisem. Oprócz asp.net cora oraz entity framworka , który służył nam do stworzenie relacyjnej bazy danych (patrz ERD), użyliśmy bootstrapa w celu zachowania odpowiedniej szaty graficznej oraz jquery z ajaxem w celu ułatwiania pracy z widokiem oraz żądaniami przekazywanymi do kontrolerów.

3. Z czego skorzystaliśmy:

- zasady czystego kodu,

- linq w celu odczytu danych z bazy,

- kolekcje,

- wyrażenia lambda,

- interfejsy,

- asynchroniczność,

- wzorzec dependency injection, n-tier architecture, mvc,

- dodatkowe frameworki,

- layout, view, partial view,

- jquery w celu wprowadzenia dynamiczności w projekcie

- bootstrap

4. Podział obowiązków względem funkcjonalności:

Konrad:

- Stworzenie oraz konfiguracja projektu,

- Stworzenie mechanizmu rejestracji i logowania,

- Stworzenie części związanej ze studentem oraz drobne poprawki dotyczące funkcjonalności związanej z nauczycielem.

Adrian:

- Projektowanie oraz nanoszenie poprawek dotyczących bazy danych,

- Stworzenie części związanej z nauczycielem,

- Aktualizacja struktury projektu.

W trakcie tworzenia projektu, oboje robiliśmy review kodu oraz wielokrotnie wprowadzaliśmy poprawki, aby struktura projektu była czytelniejsza.

5. Funkcjonalności Teacher:

Nauczyciel po pierwszym zalogowaniu, powinien przejść do zakładki „Subjects”, gdzie wybiera przedmioty, których chce nauczać. Następnie w zakładce „Classes” wybiera przedmiot oraz klasę, której chce zostać nauczycielem. W ostatniej zakładce o nazwie „Marks” nauczyciel, może przeglądać listę uczniów danej klasy, oraz wystawiać danemu uczniowi ocenę.

6. Funkcjonalności Student:

Student po pierwszym zalogowaniu podobnie jak nauczyciel wybiera klasę do której chce należeć (zakładka „Classes”). Natomiast w zakładce „Marks” może przeglądać oceny jakie dostał od nauczyciela z danego przedmiotu.

Uwagi:

Projekt został zrealizowany w VS 2017, w trakcie pierwszego uruchomienia aplikacji, baza danych zostanie dopiero stworzona oraz wypełniona wstępnymi danymi dotyczących klas oraz przedmiotów (Klas o nazwie „DbInitializer”).

Dnia : 08.08.2017 wstawimy film, opisujący kod krok po kroku. Wynika to niestety z sytuacji, która została przedstawiona prowadzącemu.