AKADEMIA GÓRNICZO HUTNICZA IM. STANISŁAWA STASZICA W KRAKOWIE

Wydział Zarządzania

SPRAWOZDANIE

Czerwiec 2022

Grupa 1

Autorzy: Monika Etrych, Anna Rubin, Marcin Kapłon

Przedmiot: Techniki Internetowe

Prowadzący: mgr Tomasz Wieroński

Spis treści:

1. Cel projektu i zastosowanie biznesowe.

Celem projektu było stworzenie aplikacji internetowej według wzoru MVC (Model-View-Controller) opartej na technologii ASP.NET. Aplikacja ma służyć właścicielowi firmy nieruchomościowej lub jego pracownikom. Dzięki jej wykorzystaniu można w przejrzysty sposób zarządzać firmą. W funkcje aplikacji wchodzi możliwość dodawania nowych danych do bazy oraz możliwość dostępu do informacji na ich temat, edycja oraz usuwanie. Dane skupiają się wokół pracowników, klientów, nieruchomości ich właścicieli oraz transakcji

2. Modele

Modele reprezentują zestaw klas opisujących dane. Opisują strukturę tych danych poprzez definiowanie elementów i ich atrybutów oraz powiązań między nimi. Wszystkie widoki są podobnie skonstruowane i napisane w języku C#. Modele zawierają całą logikę aplikacji.

W naszej aplikacji znajduje się 5 modeli: Client, Employee, Estate, Meeting i Owner, są to klasy C# na podstawie, których budowana jest baza danych.

3. Opis bazy danych wraz z jej diagramem.

Baza składa się pięciu tabel o nazwie Meetings, Clients, Employees, Estates i Owners. Tabele dotyczą przechowywanych danych. Wszystkie tabele posiadają swój unikalny klucz identyfikacyjny o nazwie Id. Wszystkie tabele zawierają relację jeden do wielu.

Dane tabeli Clients dotyczą id klienta, jego imienia i nazwiska, numeru telefonu oraz daty urodzin.

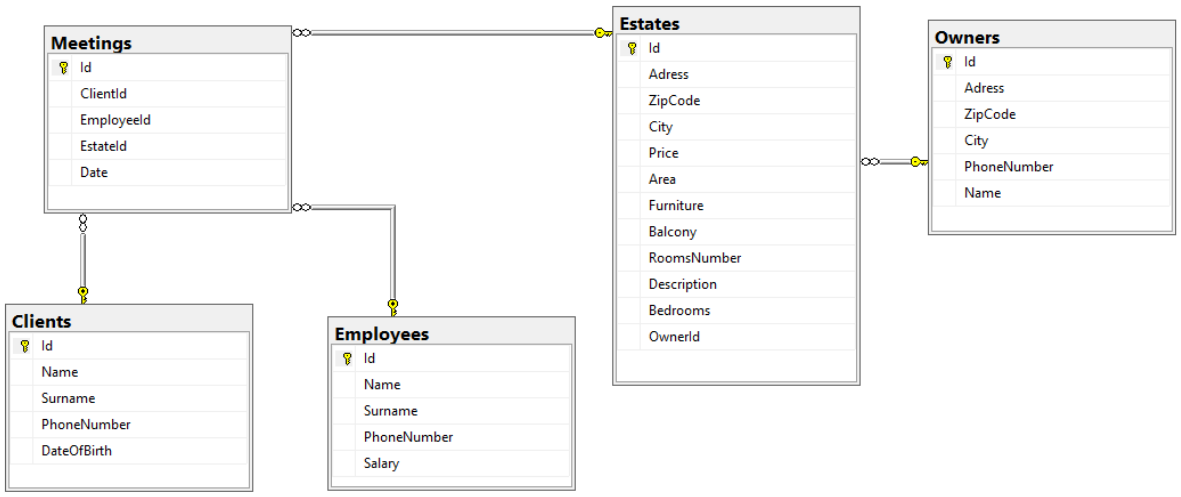
Tabela Employees złożona jest danych: id pracownika, jego imienia i nazwiska, numeru telefonu oraz kwoty wypłacanych należności.

Owners składa się z id właściciela posiadłości, jego nazwy firmy lub imienia i nazwiska, jego dokładnego adresu zamieszkania wraz z miastem, kodu pocztowego oraz numeru telefonu.

Estates dotycząca posiadłości dotyczy danych takich jak: id posiadłości, adres posiadłości i jego miasto, kodu pocztowego, ceny nieruchomości, opisu, powierzchni, id właściciela oraz danych dotyczących umeblowania, balkonu, liczby sypialń, ilości pokojów.

Tabela Meetings jest tabelą łączącą dane. Jej relacje wskazują na to, że klienci, pracownicy i przedmiot spotkań tzn. nieruchomości, mogą posiadać wiele spotkań. Tabela Meetings zawiera dane typu: id spotkania, id klienta, id pracownika, id posiadłości oraz data spotkania.

Relacja między tabelą Estates, a tabelą Owners wskazuję na to, że właściciel może posiadać wiele posiadłości, ale posiadłość może należeć tylko do jednego właściciela. Podobne relacje są pomiędzy resztą tabel.



4. Kontrolery

Kontrolery są odpowiedzialne za odpowiadanie na żądania dotyczące aplikacji internetowej. Każde żądanie przeglądarki jest mapowane na określony kontroler. Ponadto jest to jedyny element, za pośrednictwem którego użytkownik wchodzi w interakcję z systemem.

W naszej aplikacji znajduje się 7 kontrolerów. Znajdują się one w folderze Controllers są to5 głównych kontrolerów odpowiadających tableom w baize: ClientController, EmployeeController, EstateController, MeetingController, OwnerController. Ponadto dwa dodatkowe kontrolery: ErrorController i HomeController.

Ponadto główne kontrolery zawierają metody: Index(), Create(), ViewAll(), View(), Edit(), Delete(), które zwracają odpowiednie im widoki.

- Create – dodaje nowe dane do bazy danych
- ViewAll – wyświetla wszystkie dane z bazy danych dla danej tabeli
- View – wyświetla szczegóły o danym wierszu danych
- Edit – pozwala wyświetlić aktualny wiersz danych i umożliwia edytowanie pól
- Delete – usuwa dane pod wskazanym indeksem
- Index – który zwraca domyślny widok menu – Index

Dodatkowo kontroler Client zawiera metodę Error, która sprawdza czy działa error 500. Po wpisaniu w pasku przeglądarki: Client/Error wyskakuje odpowiedni błąd.

Kontroler Error zawiera metody Index() i NotFound(), które zwracają widok Index(error 500) bądź NotFound(error 404) z folderu Shared.

Kontroler Home zawiera metody Index (strona startowa) i About(miejsce, gdzie znajduje się dokumentacja), które zwracają odpowiadające im widoki.

5. ViewModele

ViewModele są warstwą pośrednią pomiędzy widokami, a modelami. W nich określa się walidację dla wprowadzanych danych, sposób wyświetlania danych, czy wypełnienie danego pola jest obowiązkowe oraz komunikaty zwrotne w przypadku błędów przy wprowadzaniu danych.

Zawarte w projekcie ViewModele odpowiadające modelom:

- ClientVM
- EmployeeVM
- EstateVM
- MeetingVM
- OwnerVM

Oraz jeden dodatkowy - ValidateDate, który służy do sprawdzania poprawności dat wprowadzanych przez użytkownika.

ViewModele zawierają metody ToDbModel, który mapuje ViewModel na DbModel oraz FromDbModel, która działa w drugą stronę.

6. Widoki

Widoki reprezentują dane do wyświetlania z viewmodeli, czyli to ta część aplikacji, która jest odpowiedzialna za prezentacje danych. Określają sposób wyświetlania interfejsu aplikacji.

Widoki odpowiadające modelom: Client, Employee, Estate, Owner i Meeting zawierają taką samą strukturę plików spełniającą tą samą funkcję - do obsługi CRUDa:

- Create
- View(Details)
- Edit
- Delete

Oraz:

- ViewAll(do wyświetlania wszystkich danych)
- Index(podstawowe menu każdego modelu)

Projekt zawiera również widoki:

- MenuBar (partial view)
- NotFound (widok błędu 404)
- Error (widok błędu 500)

Widok Index

Przy pomocy `ActionLink` możliwe jest wykonanie akcji przejścia do widoku ViewAll tzn. listy pracowników oraz przejścia do widoku Create tzn. widoku opisanego na początku, umożliwiającego stworzenie poprzez formularz nowego pracownika. Jest to widok początkowy wybranej pozycji z menu.

Widok Create

Następnie określany jest nagłówek przy pomocy znaczników html <h2>. Umieszczany on jest na stronie w celu poinformowania użytkownika, co może na tej stronie wykonać. Wyświetlany tekst to: „Add a new employee”.

Następnie pojawia się @using (Html.BeginForm()), co ułatwia pracę w tworzeniu formularzy. Metoda BeginForm służy do tworzenia takich właśnie formularzy w formacie html. W metodzie BeginForm u początku pojawia się @Html.AntiForgeryToken(), dzięki czemu jesteśmy chronieni przed ingerencją hakerską.

W dalszej części widoku znajdują się znaczniki służące wyświetlaniu opisów do wprowadzanych danych oraz miejsca do wprowadzania danych.

Możliwe dane do wpisania to w przypadku Employee: imię(string), nazwisko(string), numer telefonu(string z walidacją liczby cyfr), wartość wypłaty(decimal).

Ponadto w tym widoku znajduję się przycisk Create, który umożliwia przesłanie danych do bazy oraz odnośnik do widoku początkowego Index.

Widok ViewAll

W tym widoku wyświetlana jest tabela zawierająca wszystkie dane pracowników z tabeli Employees.

Nazwy zmiennych wyświetlane są przy pomocy DisplayNameFor, a ich rekordy dzięki wykorzystaniu intrukcji foreach i DisplayFor.

Dodatkowo, dzięki Url.ActionLink i ActionLink umieszczonym przed tabelą mamy możliwość przekierowania nas do widoku Create, umożliwiającego tworzenie nowego pracownika oraz dzięki umieszczonym w każdym rekordzie tabeli przyciskom mamy możliwość przejścia do widoków View, Edit, Delete.

Widok View

Ten widok wyświetlany jest po wybraniu opcji Details z listy pracowników (podobnie jak Edit i Delete). W tym widoku wyświetlane są dane wybranego rekordu – pracownika. Dane wyświetlane są analogicznie jak w przypadku widoku Delete, czyli Dalej przy pomocy DisplayNameFor oraz DisplayFor wyświetlane są kolejno dane wybranej pozycji – pracownika – dokładniej nazwa zmiennej i jej zawartość.

Widok Edit

Ten widok odpowiada za edycję zapisanych już rekordów w bazie danych. Poprzez wybranie opcji Edit przy danej pozycji pracownika jesteśmy przekierowywani do tego widoku. Wyświetla nam się analogiczny formularz jak w przypadku widoku Create, lecz pola służące do wypełniania danych są wypełnione istniejącymi danymi, dzięki czemu mamy możliwość edycji rekordów. Przycisk przedtem

służący do stworzenia pracownika w widoku Create, teraz służy do zapisania zmodyfikowanych danych pod nazwą Save.

Widok Delete

W tym widoku na początku również pojawia nam się tytuł strony „Delete" oraz pojawia się nagłówek z informacją dla użytkownika co może wykonać. W tym przypadku wyświetla się napis, czy użytkownik na pewno chce usunąć pozycję. Dalej przy pomocy `DisplayNameFor` oraz `DisplayFor` wyświetlane są kolejno dane wybranej pozycji – pracownika – dokładniej nazwa zmiennej i jej zawartość.

Metoda `Url.Action` umożliwiająca wykonanie akcji jaką jest powrót do listy wszystkich pracowników oraz przycisk umożliwiający usunięcie danej pozycji.

Widok MenuBar

Plik MenuBar.cshtml znajduje się lista wyświetlanego menu, czyli: Logo(Index), About, Client, Employee, Estate, Owner, Meeting. Dzięki `ActionLink` mamy możliwość przekierowania do innych widoków. Przykładowo pierwsze dwa odnoszą się do widoków Index, About z folderu widoków Home, a pozostałe odnoszą się do widoków Index wybranych pozycji.

Widoki Home

Widok Home zawiera pliki Index, About. Pierwszy służy wyświetlaniu głównej strony, a drugi wyświetla informacje.

Widoki Shared

Aplikacja ma spójny układ, który umożliwia użytkownikowi przechodzić od strony do strony w spójnym środowisku. Układ zawiera typowe elementy interfejsu użytkownika i jego zawartość znajduje się w widoku Shared. Układ zmniejsza ilość zduplikowanego kodu w widokach.

Plik _Layout.cshtml określa elementy <head>, <body> oraz <footer> strony html. W <head> umieszczone są takie elementy jak odczytywanie polskich znaków przez 'utf-8', metadane, tytuł wyświetlanej strony oraz odwołanie do stylów. Natomiast w <body> umieszczony jest odnośnik do partialview MenuBar.

Zdefiniowany jest również wygląd stopki strony, w której wyświetlany jest znak copywriter, aktualny rok oraz nazwa projektu.