Dokumentacja projektu TI

Autorzy:
Tomasz Solarski
Aleksandra Kulpa
Sebastian Sukiennik

1. Cel projektu

Celem projektu jest stworzenie prostej aplikacji sklepu sportowego połączonej z lokalną bazą danych w technologii ASP.NET MVC, w której użytkownik może przeglądać przedmioty podzielone na działy, dodawać je do koszyka, a następnie składać zamówienia. Z poziomu administratora możemy zobaczyć listę wszystkich produktów i listę wszystkich działów w naszym sklepie, możemy je dodawać, usuwać i edytować. Możemy również zobaczyć wszystkie zamówienia, ich szczegóły, możemy je usuwać lub oznaczać jako wysłane.

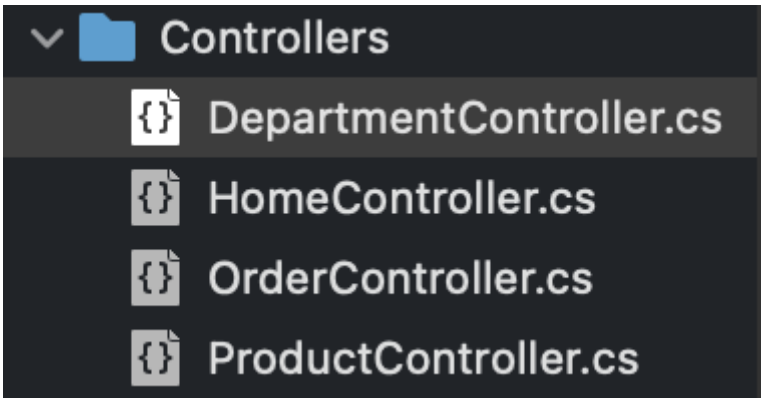
2. Zastosowanie biznesowe

Aplikacja jest jednym z prawdopodobnie najczęściej występujących typów stron w internecie. Prosty sklep, o ściśle określonej tematyce, z podkategoriami do filtrowania itemów (w naszym przypadku filtrowanie po działach). Rozwiązanie to przez swoją uniwersalność i szerokie spektrum zastosowań po niewielkich zmianach może znaleźć zastosowanie w innych tematykach. W stosunkowo łatwy sposób można by zmienić tematykę sklepu z sportowego na np. samochodowy, muzyczny, palarni kawy, etc. Całość funkcjonalność, tzn. wyświetlanie przedmiotów, zmianę kategorii, dodawanie produktów do koszyka itp. pozostałaby niezmieniona.

Aplikacja ta stwarza więc rozwiązanie problemu digitalizacji, gdzie lokalny przedsiębiorca z małym sklepem aby pozostać w biznesie musi posiadać własną domenę internetową. Nawet większość klientów pozostanie przy zakupach stacjonarnych, strona internetowa pozwala dotrzeć do szerszej grupy odbiorców, którzy mogą przeglądać ofertę sklepu. W naszym konkretnym przypadku strona ta byłaby idealna dla każdego małego sklepu zajmującego się sprzedażą detaliczną produktów sportowych z zakresu piłki nożnej i pływnia.

3. Dokumentacja kontrolerów

Kontrolery w projekcie odpowiedzialne są za przyjmowanie danych wejściowe od użytkownika i reagowanie na jego poczynania, zarządzają, aktualizują model oraz odświeżają widok. W naszym projekcie stworzyliśmy 4 kontrolery, jeden dla departamentów: DepartamentController, jeden domyślny dla strony początkowej: HomeController, dla strony zamówień OrderController oraz dla strony produktów ProductController.



Wszystkie kontrolery dziedziczą po klasie Controller. W poszczególnych kontrolerach znajdują się metody GET, POST dzięki którym obsługiwane są czynności wykonane przez użytkownika. Na przykład w kontrolerze ProductController.cs są m.in. metody:

- ViewAll() zwracająca widok z produktami należącymi do departamentu
- Create() zwracająca widok z nowym produktem
- Edit() możliwość edycji produktu
- Delete() usuwanie produktu

Podobne akcje obsługiwane są przez kontroler Order, gdzie obsługiwane są akcje dotyczące zamówienia, w tym:

- Checkout() - przejście do podsumania zamówienia
- ViewAll() - zwracający widok zamówień
- Shipped() - aktualizacja wartości czy wysłany dla danego produktu
- itp.

Kontroler Department ma metody podobne do kontrolera produktów, natomiast HomeController jest odpowiedzialny jedynie za zwrócenie widoku początkowego z odpowiednim ustalonym departamentem.

4. Dokumentacja modeli

W modelach zawarte są pliki odpowiedzialne za strukturę danych, tzn. klasy określające właściwości produktów, departamantów, zamówień itp.

Oprócz oczywistych klas takich jak Product, Order, Department, Cart, SessionCart mamy także klasy:

- SeedData - odpowiedzialna za zapełnienie aplikacji przykładowymi danymi
- StoreDbContext - połączenie z bazą danych, dziedzicząca po DbContext

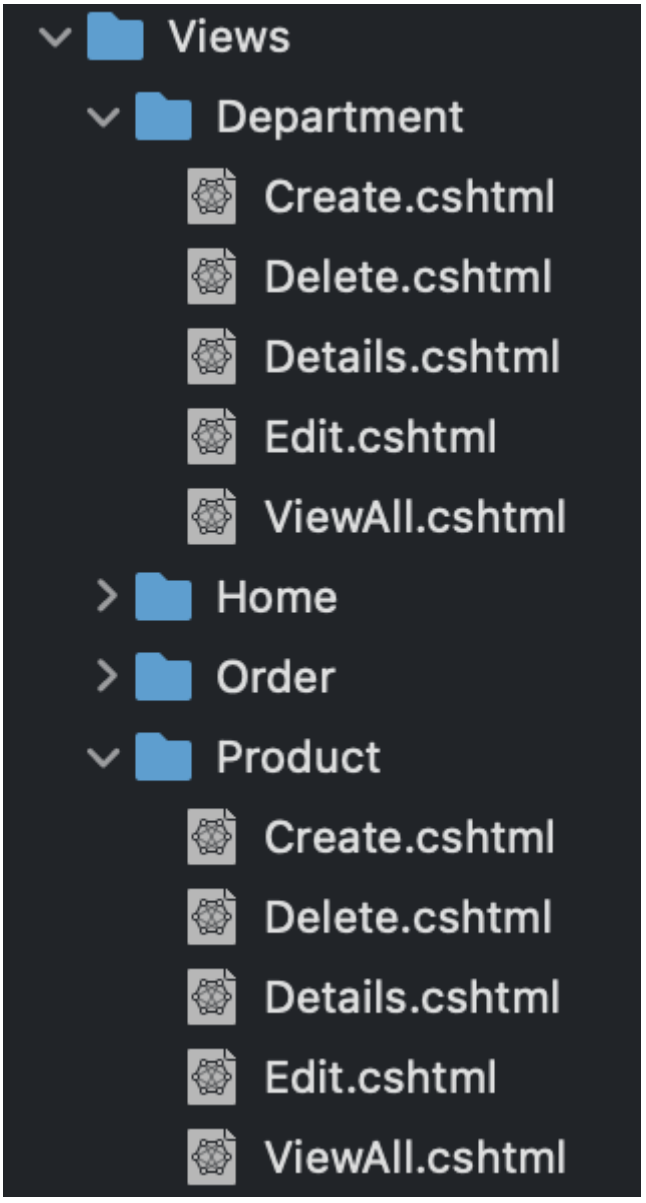
Klasy EFDepartmentRepository, EFOrderRepository, EFStoreRepository implementujące analogiczne interfejsy. Oznaczone skrótem EF (EntityFramework) odpowiedzialne są za bezpośrednie dodawanie, usuwanie, wybieranie i edytowanie rekordów w bazie danych (w odpowiadających tabelach). Przykładowy kod klasy

EFDepartmentRepository zaprezentowano poniżej, w pozostałych klasach wygląda on analogicznie.

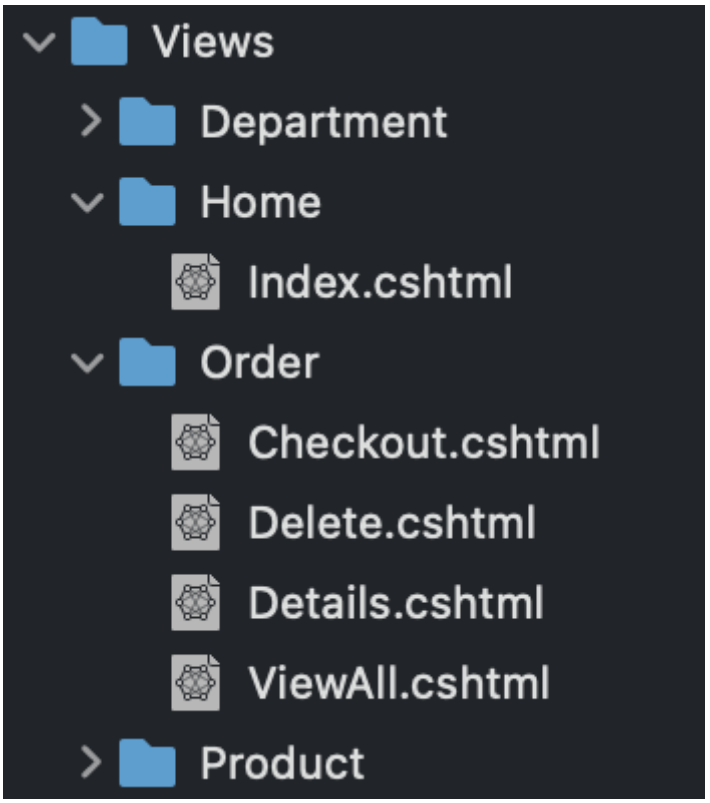
```
public class EFDepartmentRepository : IDepartmentRepository
{
    private StoreDbContext context;
    public EFDepartmentRepository(StoreDbContext ctx)
    {
        context = ctx;
    }
    public IQueryable<Department> Departments => context.Departments;

    public void Add(Department department)
    {
        context.Departments.Add(department);
        context.SaveChanges();
    }

    public void Delete(int id)
    {
        var result = GetById(id);
        if (result != null)
        {
            context.Departments.Remove(result);
            context.SaveChanges();
        }
    }

    public Department GetById(int id)
        => context.Departments.FirstOrDefault(x => x.DepartmentID == id);

    public void Update(Department department)
    {
        context.Entry(department).State = EntityState.Modified;
        context.SaveChanges();
    }
}
```

5. Dokumentacja widoków

Widoki są częścią modelu MVC odpowiedzialną za prezentowanie danych użytkownikowi. W widokach produktu są stworzone podobne widoki, do tworzenia, usuwania, szczegółów, edytowania oraz główny ViewAll.cshtml odpowiedzialny za wyświetlanie wszystkich departamentów/produktów.



Widok strony początkowej Home ma jedynie jeden plik jest nim Index.cshtml



W pliku Index.cshtml tworzony jest po prostu kontener w którym wypisywane są produkty z modelu Products oraz aktualnie zaznaczony deprtament.

```
@model ProductsListViewModel

@foreach (var p in Model?.Products ?? Enumerable.Empty<Product>())
{
    <partial name="ProductSummary" model="p" />
}
<div page-model="@Model?.PagingInfo" page-action="Index" page-classes-enabled="true"
    page-class="btn" page-class-normal="btn-outline-dark"
    page-class-selected="btn-primary" page-url-category="@Model?.CurrentDepartment!" class="btn-group pull-right m-1">
</div>
```

W widoku Order utworzone zostały pliki niezbędne do zaprezentowania użytkownikowi zamówienia. Obejmuje to stronę Checkout na której znajduje się podsumowanie składanego zamówienia, Delete gdzie można potwierdzić usunięcia zamówienia, Details informująca o szczegółach danego zamówienia oraz ViewAll.cshtml która ponownie wyświetla wszystkie obecne zamówienia.

6. Instrukcja użytkowania aplikacji

Gdy wchodzimy na aplikację jesteśmy na stronie głównej, na niej mamy listę wszystkich dostępnych w sklepie produktów. Jeżeli chcemy obejrzeć produkty z danego działu to wchodzimy w ten dział klikając w odpowiedni przycisk po lewej stronie. Aby dodać produkt do koszyka naciskamy przycisk „Add to Cart", jeżeli chcemy kontynuować zakupy to naciskamy „Continue shopping", jeśli jednak chcemy dokonać zakupu klikamy „Checkout", a następnie wypełniamy formularz i klikamy „Complete order". Jeśli chcemy zobaczyć co znajduje się w naszym koszyku – klikamy ikonę wózka po prawej stronie. Tam możemy również usunąć pozycje z koszyka.

Gdy wejdziemy na część przeznaczoną dla administratora również widzimy listę wszystkich produktów, jeżeli chcemy dodać jakiś nowy produkt klikamy „Add new product", jeżeli chcemy edytować istniejący już produkt – znajdujemy go na liście i klikamy „Edit". Podobnie robimy, gdy chcemy zobaczyć szczegóły produktu (klikamy „Details"), albo gdy chcemy usunąć produkt („Delete"). Kiedy wejdziemy w zakładkę „Orders" możemy zobaczyć zamówienia. Aby usunąć zamówienie klikamy „Delete", gdy chcemy oznaczyć je jako wysłane to klikamy „Shipped", a gdy chcemy zobaczyć szczegóły to klikamy „Details". Zakładka „Departments" działa dokładnie tak samo jak zakładka „Products" tylko dotyczy działów, a nie produktów.

7. Podział obowiązków

Tomasz Solarski - backend, modele, kontrolery, połączenie z bazą danych

Sebastian Sukiennik - walidacje formularzy, JavaScript, Widoki (frontend)

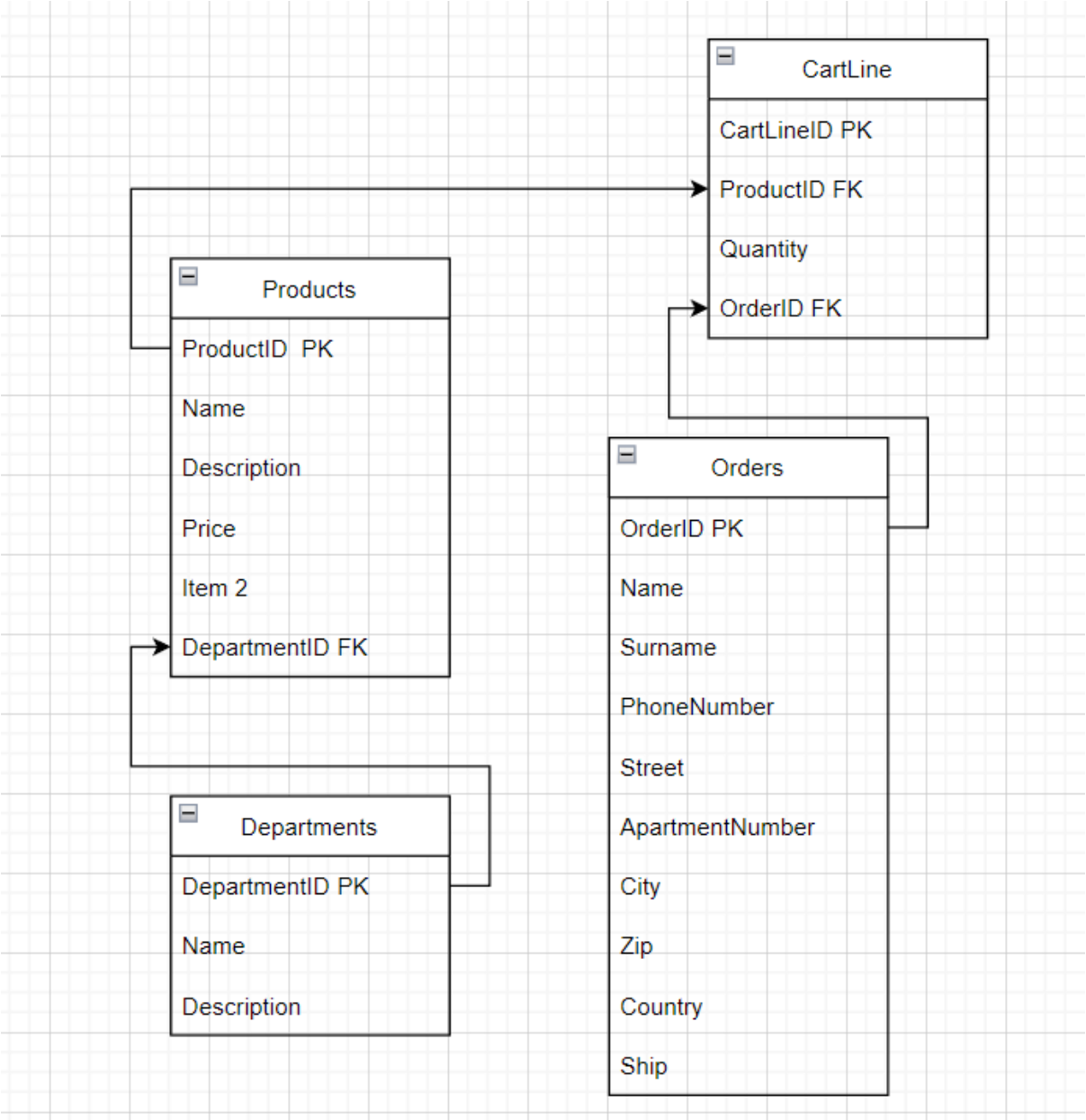
Alekdsandra Kulpa - dokumentacja, diagramy bazy danych, cel i zastosowanie projektu

8. Diagram bazy danych

W bazie danych stworzonej przy użyciu podejścia CodeFirst, czylie wygenerowanej po napisaniu uzyskaliśmy dokładnie 4 tabele:

- CartLine - zawierające produkty znajdujące się w koszyku
- Departments - departamenty sklepu i ich opisy
- Orders - zamówienia
- Products - produkty i podstawowe informacje o nich

Poniżej zaprezentowano diagram tabel w bazie danych:


CartLine
CartLineID PK
ProductID FK
Quantity
OrderID FK
Products
ProductID PK
Name
Description
Price
Item 2
DepartmentID FK
Orders
OrderID PK
Name
Surname
PhoneNumber
Street
ApartmentNumber
City
Zip
Country
Ship
Departments
DepartmentID PK
Name
Description