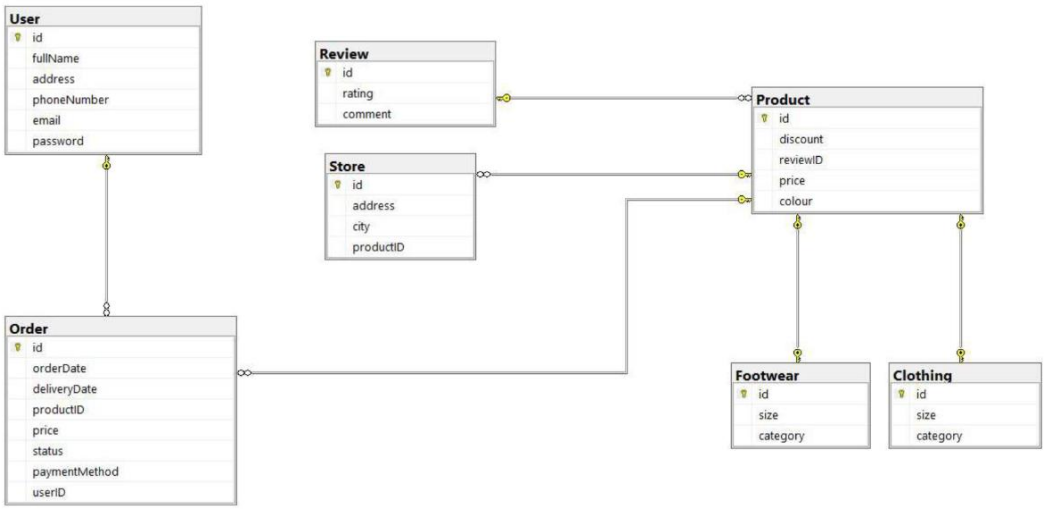
Refactoring

1. Prepravljanje ER dijagrama kroz kod

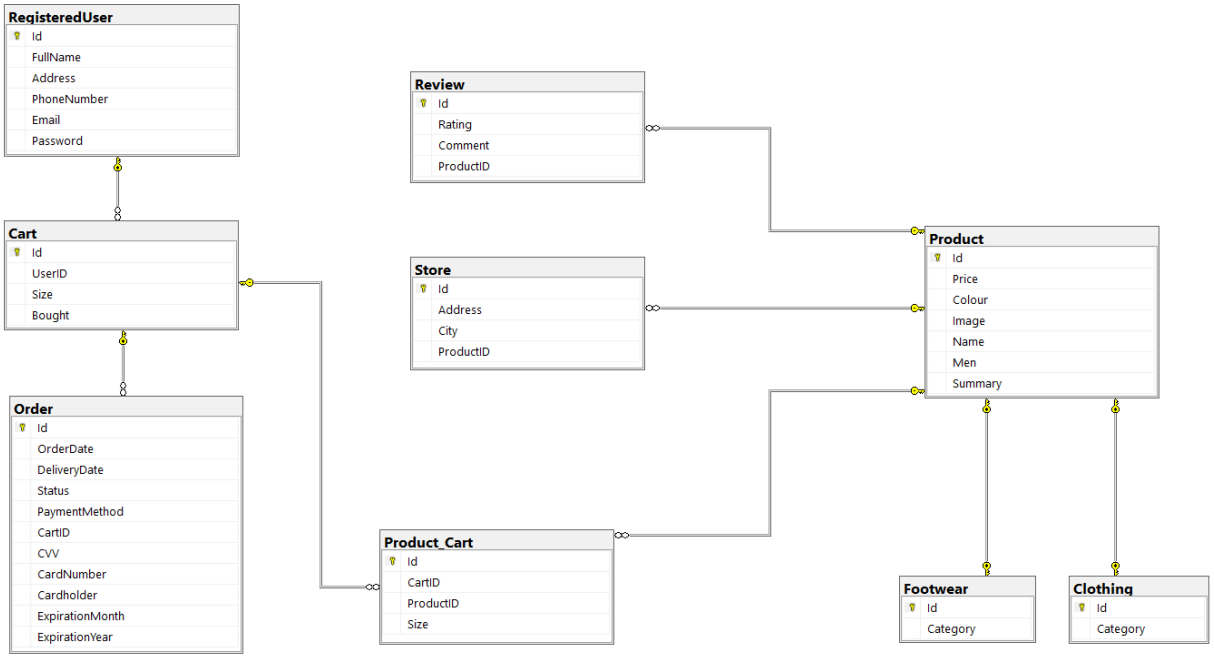
Nakon prve izvršene migracije, primjetili smo nekoliko propusta u početnoj fazi projektovanja naše web aplikacije. Tokom implementacije, postalo je očito da su određene stvari nedostajale a druge bile viška. Ustanovili smo da su neke tabele i neki atributi nedostajali u prvobitnom dizajnu baze podataka. To je dovelo do nemogućnosti pohranjivanja određenih informacija koje su ključne za funkcionalnost aplikacije. Veze između tabela su bile neispravno definisane, što je rezultiralo problemima u dohvaćanju i manipulaciji podacima te otežalo rad s bazom podataka.

ERD nakon prve izvršene migracije:



Ovi propusti su ukazali na potrebu za refaktoringom i poboljšanjem strukture baze podataka kako bi se osigurala pravilna i efikasna upotreba podataka u aplikaciji. U narednim fazama razvoja, fokusirali smo se na ispravljanje tih grešaka i implementiranje potrebnih promjena kako bismo osigurali pravilan rad aplikacije i zadovoljstvo korisnika.

ERD nakon izvršenih popravki:



Glavni propust koji smo primijetili nakon prve migracije baze podataka bio je nedostatak tabele "Cart" u kojoj bi se čuvali podaci o artiklima koji se nalaze u korisnikovoj korpi. S time da jedna korpa može sadržiti više proizvoda kao i što se jedan proizvod može nalaziti u više korpi, potrebno je bilo dodati međutabelu „Product_Cart“.
U tabeli „Product“ su nedostajali atributi Image, Name, Men i Summary te je prvobitni atribut Discount bio viška. Kao što smo naveli u specifikaciji svaki proizvod će imati svoju odgovarajuću sliku, opis kao i naziv te je to bilo potrebno i dodati. Potrebno je bilo da dodatno naznačimo da li je određeni artikal muški ili ženski, te smo uklonili Discount iz razloga što naša web aplikacija ne nudi popuste. Također smo vezu okrenuli tako da jedan artikal može imati više recenzija, dok u prvobitnom ERD-u je izgledalo kao da se jedna recenzija odnosi na više artikala. Promijenili smo ime tabele „User“ u „RegisteredUser“ iz razloga što je potrebno da vršimo evidenciju samo registrovanih korisnika. U tabeli „Order“ nismo bilježili podatke koje korisnik unosi prilikom naručivanje proizvoda u zavinosti od vrste plaćanja koju je odabrao.

Model „Product“ prije popravke

```
public abstract class Product
{
    [Key]
    26 references
    public int Id { get; set; }
    0 references
    public string discount { get; set; }
    [ForeignKey("Review")]
    0 references
    public int reviewID { get; set; }

    0 references
    public string price { get; set; }
    0 references
    public string colour { get; set; }

    0 references
    public Product() { }
}
}
```

Model „Product“ nakon popravke

```
public abstract class Product
{
    [Key]
    26 references
    public int Id { get; set; }
    21 references
    public string Price { get; set; }
    18 references
    public string Colour { get; set; }
    27 references
    public string Image { get; set; }
    22 references
    public string Name { get; set; }
    13 references
    public Boolean Men { get; set; }

    19 references
    public string Summary { get; set; }

    0 references
    public Product() { }
}
}
```

2. Refactoring na nivou podataka

Prilikom migracije, susreli smo se s problemom kada smo koristili tip varijable "DateTime" za određivanje datuma i vremena. Odlučili smo promijeniti tip varijable u „string“ što je omogućilo pravilnu migraciju. Promjenu smo izvršili u tabeli „Order“ gdje imamo varijablu „OrderDate“ i „DeliveryDate“.

```
public class Order
{


    [Key]
    8 references
    public int Id { get; set; }
    9 references
    public string OrderDate { get; set; }
    9 references
    public string DeliveryDate { get; set; }
    9 references
    public string Status { get; set; }
    9 references
    public PaymentMethod PaymentMethod { get; set; }
    [ForeignKey("Cart")]
    8 references
    public int CartID { get; set; }
    6 references
    public Cart Cart { get; set; }

    2 references
    public string? CardNumber { get; set; }
    2 references
    public int? ExpirationMonth { get; set; }
    2 references
    public int? ExpirationYear { get; set; }
    2 references
    public string? Cardholder { get; set; }
    2 references
    public int? CVV { get; set; }


    2 references
    public Order() { }
}
}
```

3. Promjena izgleda pogleda putem HTML i CSS

Željeli smo da naša aplikacija bude vizuelno privlačna, te se nismo zadovoljili sa defaultnim izgledom. Defaultni izgled pogleda je promijenjen putem HTML i CSS fajlova. Također smo koristili JavaScript kada je bio neophodan, uprkos nedostatku iskustva sa njim.

Prikaz slike artikla „Product/Details“ nakon njegovog odabira na prethodnom prozoru:

```
<div class="container" style="margin-top:25px">
    <div class="div4">
        <img src="@Url.Content(Model.Image)" alt="@Model.Name" style="width:400px; height: 550px; margin-left:90px">
    </div>
```

Promjena natpisa na buttonu u zavisnosti od odabrane opcije plaćanja korištenjem JavaScript:

```
<script>
    function selectPaymentMethod(methodId) {
        const paymentMethods = document.querySelectorAll('input[name="paymentMethod"]');
        let selectedMethod = '';
        paymentMethods.forEach(method => {
            if (method.id === methodId) {
                method.checked = true;
                selectedMethod = methodId;
            } else {
                method.checked = false;
            }
        });

        const continueButton = document.getElementById('continueButton');
        if (selectedMethod === 'cash-on-delivery') {
            continueButton.innerText = 'FINISH ORDER';
        } else if (selectedMethod === 'credit-card') {
            continueButton.innerText = 'PAY WITH CREDIT CARD';
        } else {
            continueButton.innerText = 'CONTINUE';
        }
    }
```