

Porucivanje pita

CS322 – C#

Profesor: Student:

Aleksandar Jovanović Andrijana Jovanović 3918

**Sadržaj**

1. Uvod ………………………………………………………………………… 3

2. Funkcionalni zahtevi ………………………..…………………………….…. 4

3. Use case ……………………………………………………………………... 4

4. Scenariji koriscenja…………………………………………………………... 5

5. Database diagram…………………………………………………………….. 8

6.Implementacija ………………………………………………………………...9

7. Zakljucak …………………………………………………………………… 27

8.Literatura …………………………………………………………………….. 27

**1. Uvod**

Tema ovog projektnog zadatka jeste izrada i dokumentovanje sistema za poslasticarnicu. Program je dizajniran tako da olakša i ubrza rad jedne poslasticarnice tako što će njegovim radnicima i korisnicima omogućiti alate kojima oni komuniciraju sa celim sistemom.

Ovaj sistem omogućava administratoru, zapošljenom u poslasticarnici, da pregleda, doda, izmeni i izbriše pitu, pregled svih porudzbina. Takodje ima mogucnost pregleda svih korisnika, izmenu i brisanje.

Korisnik ima mogućnost pregleda svih pita, dodavanje u korpu, pregled korpe.

Projekat je zasnovan na ASP.NET-u, **ASP.NET MVC** omogućava programerima softvera da naprave web aplikaciju kao sastav od tri uloge: Model, View i Controller. MVC model definiše web aplikacije sa 3 logička sloja:

* Model (poslovni sloj)
* Prikaz (prikaži sloj)
* Kontroler (kontrola ulaza)

Model predstavlja stanje određenog aspekta aplikacije. Kontroler upravlja interakcijama i ažurira model kako bi odražavao promenu stanja aplikacije, a zatim prenosi informacije u prikaz. Pogled prihvata potrebne informacije od kontrolera i prikazuje korisnički interfejs za prikaz tih informacija.

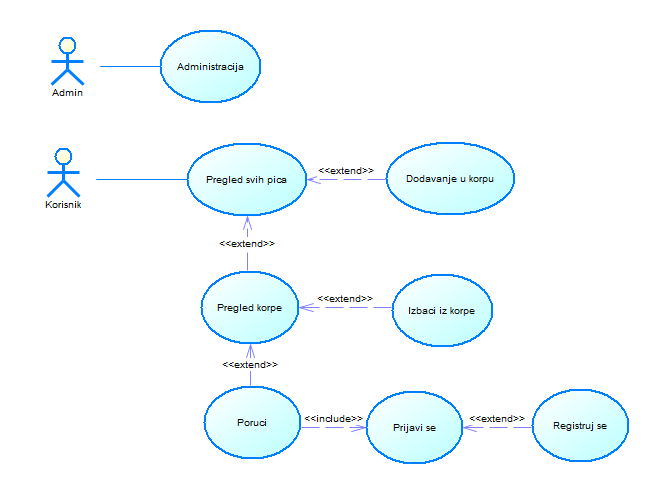
Koristila sam **SQL Server Management Studio (SSMS).** je softverska aplikacija koja je prvi put pokrenuta sa Microsoft SQL Server 2005 koja se koristi za konfigurisanje, upravljanje i administriranje svih komponenti unutar Microsoft SQL Servera. SSMS je jedan od alata za upravljanje SQL Serverom, bez obzira na vašu lokaciju, koji se koristi za dizajniranje upita i upravljanje bazama podataka i skladištima podataka putem personalnog računara.

**2.Funkcionalni zahtevi**

****

**3.Use Case diagram**

Na sledecoj slici je prikazan dijagram slucajeva koriscenja za korisnika i administratora Sistema.



Korisniku se prikazuju sve pite i ima mogucnost da doda koju zeli pitu u korpu. Takodje korisnik ima mogucnost pregleda svoje korpe gde ce se prikazati sve pite koje zeli da poruci, mogucnost porucivanja svega sto je u korpi ukoliko je prijavljen na sistem.

Administrator nakon uspesne prijave na sistem ima mogucnost upravaljanja svim pitama (sto podrazumeva dodavanje novih i modifikovanje postojecih pita), pregleda svih narudzbina i korisnika.

**4.Scenariji koriscenja:**

|  |  |
| --- | --- |
| **Slučaj korišćenja:** | **Prijavi se** |
| **Opis** | Korisnik ima mogucnost prijave na sistem |
| **Cilj** | Uspesna prijava na sistem |
| **Aktori** | Korisnik |
| **Preduslovi** | Aprikacija ima konekciju sa Internetom |
| **Okidač** | Korisnik klikom na dugme “Prijavi se” vrsi prijavu na sistem |
| **Normalni tok** | 1. Korisnik unosi svoje korisnicko ime  2. Korisnik unosi svoju sifru  3. Korisnik klikom na dugme “Prijavi se” vrsi prijavu na sistem |
| **Izuzeci** | 1. Korisnik je uneo nevalidne podatke  2. Korisnik se prikazuje error poruka  3. Korisnik može pokušati ponovo da se uloguje |

|  |  |
| --- | --- |
| **Slučaj korišćenja:** | **Registruj se** |
| **Opis** | Korisnik ima mogucnost registracije na sistem |
| **Cilj** | Uspesna registracija |
| **Aktori** | Korisnik |
| **Preduslovi** | Aplikacija ima konekciju sa Internetom |
| **Okidač** | Korisnik klikom na dugme “Registruj se” vrsi registraciju na sistem |
| **Normalni tok** | 1. Korisnik unosi korisnicko ime  2. Korisnik unosi korisicnku lozinku  3. Korisnik potvrdjuje unetu lozinku  4. Korisnik klikom na dugme “Registruj se” vrsi registraciju na sistem |
| **Izuzeci** | 1. Korisnik je uneo nevalidne podatke  2. Korisnik je uneo korisnicko ime koje vec postoji  3. Korisnik se prikazuje error poruka  4. Korisnik može pokušati ponovo da se registruje |

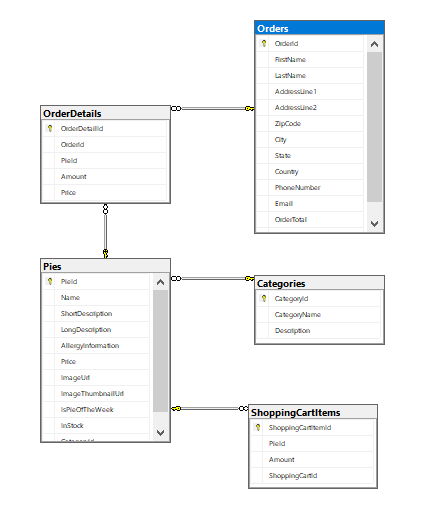
|  |  |
| --- | --- |
| **Slučaj korišćenja:** | **Administracija svim pitama** |
| **Opis** | Administrator ima mogućnost upravljanja svim pitama |
| **Cilj** | Administracija pitama |
| **Aktori** | Administrator |
| **Preduslovi** | Administrator je ulogovan na sistem |
| **Okidač** | Administrator iz liste pita bira jedanu pitu koji želi da obriše ili izmeni, klikom na dugme ***Obriši ili Izmeni***  Klikom na dugme ***Dodaj***  vrši dodavanje neke pite |
| **Normalni tok** | Administrator ima mogucnost izbora opcije:  Administratoru se prikazuje lista svih prethodno kreiranih pita   * ***Opcija:*** Dodaj novu pitu  1. Administrator unosi sliku, sastoje(ili opis), cenu pice 2. Administrator čuva te podatke klikom na dugme ***Sačuvaj* [EXC 1]**  * ***Opcija:*** Izmeni izabranu pitu  1. Iz liste pita bira jedanu pitu koji zeli da ažurira 2. Klikom na dugme ***Izmeni*** ima mogućnost izmene pojedinačnih podataka 3. Koordinator klikom na dugme ***Sačuvaj,*** čuva izmenjeni aranžman **[EXC 2]**  * ***Opcija***:Obriši izabranu pitu  1. Iz liste aranžmana bira jedan pitu koji želi da obriše 2. Klikom na dugme ***Obriši*** vrši se brisanje |
| **Izuzeci** | **[EXC 1]** Ukoliko prilikom dodavanja nove pitue ona vec postoji u listi, izlazi obaveštenje o njegovom postojanju  **[EXC 2]** Ukoliko su podaci prilikom ažuriranja pita netačni ili već postoje izbacuje se obaveštenje |

|  |  |
| --- | --- |
| **Slučaj korišćenja** | **Pregled svih pita i dodavanje u korpu** |
| **Opis** | Korisnik ima mogućnost pregleda svih pita |
| **Cilj** | Korisnik moze da pregleda sve pite te poslasticarnice |
| **Aktori** | Korisnik |
| **Preduslovi** | Korisnik mora biti registrovan na sistem |
| **Okidač** | Korisnik klikom na dugme “Dodaj u korpu” vrsi dodavanje izabrane pite u korpu |
| **Normalni tok** | * Sistem prikazuje korisniku sve pite * Korisnik klikom na dugme “Dodaj u korpu” vrsi dodavanje izbrane pite * Korisnik klikom na dugme “Pregled korpe” ima mogucnost da vidi sve pite koje je dodao u korpu |

|  |  |
| --- | --- |
| **Slučaj korišćenja** | **Pregled korpe** |
| **Opis** | Korisnik ima mogućnost pregleda svoje korpe u koju je dodao pite koje zeli da poruci i mogucnost izbacivanje istog |
| **Aktori** | Korisnik |
| **Preduslovi** | Korisnik mora biti registrovan na sistem |
| **Okidač** | Korisnik klikom na dugme “Pregled korpe” ima mogucnost pregleda svih pita koje je dodao u korpu |
| **Normalni tok** | * Sistem prikazuje korisniku sve pite iz korpe * Korisnik klikom na dugme “Izbaci iz korpe” vrsi izbacivanje izabrane pite * Korisnik klikom na dugme “Poruci” ima mogucnost da vidi sve pite koje je dodao u korpu |

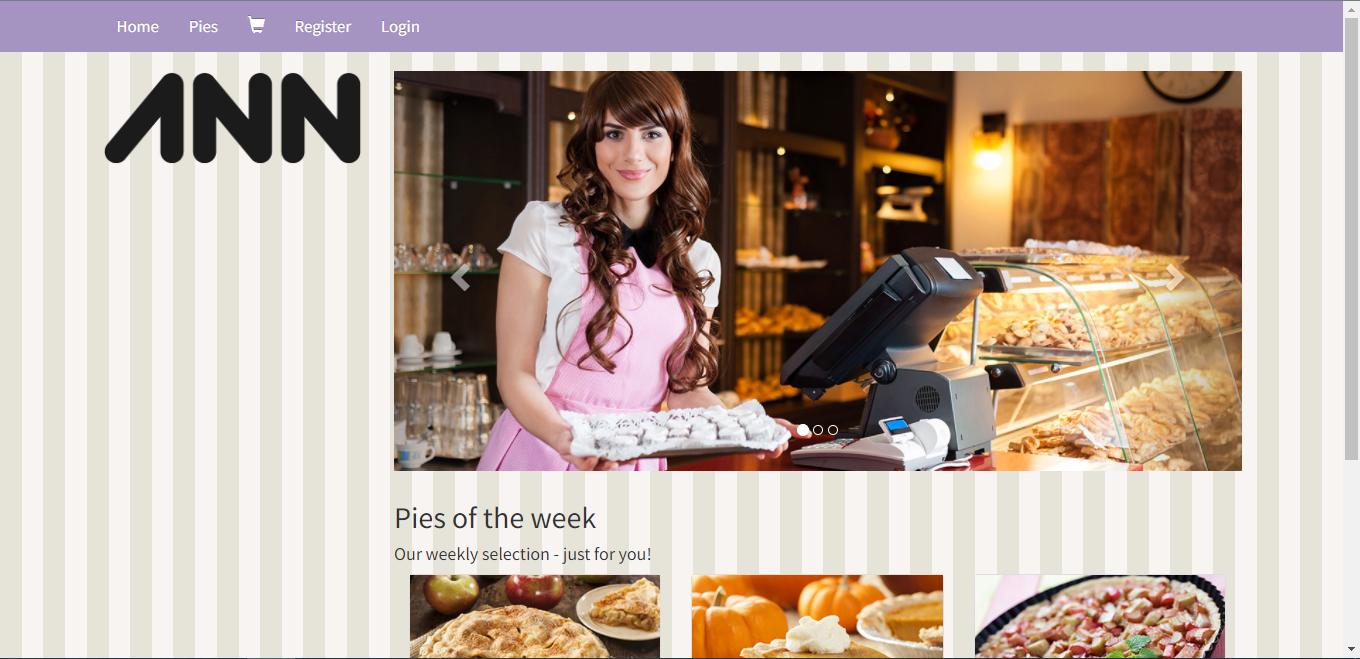
|  |  |
| --- | --- |
| **Slučaj korišćenja** | **Poruci** |
| **Opis** | Korisnik ima mogućnost porucivanja pite |
| **Aktori** | Korisnik |
| **Preduslovi** | Korisnik treba da bude prijavljen na sistem |
| **Okidač** | Korisnik klikom ***“Poruci”*** vrsi porucivanjesvih pita koje je dodao u korpu |
| **Normalni tok** | * Sistem prikazuje sve pite koje se nalaze u korpi * Korisnik vrsi porucivanje klikom na dugme “Poruci” * Sistem prikazuje korisniku formu(prikazuje adresu i grad) * Korisnik popunjava zadatu formu. Unosi adresu i grad gde zeli da mu stigne pita. * Klikom na dugme “Potvrdi” vrsi porucivanje. |

**5.Database diagram:**

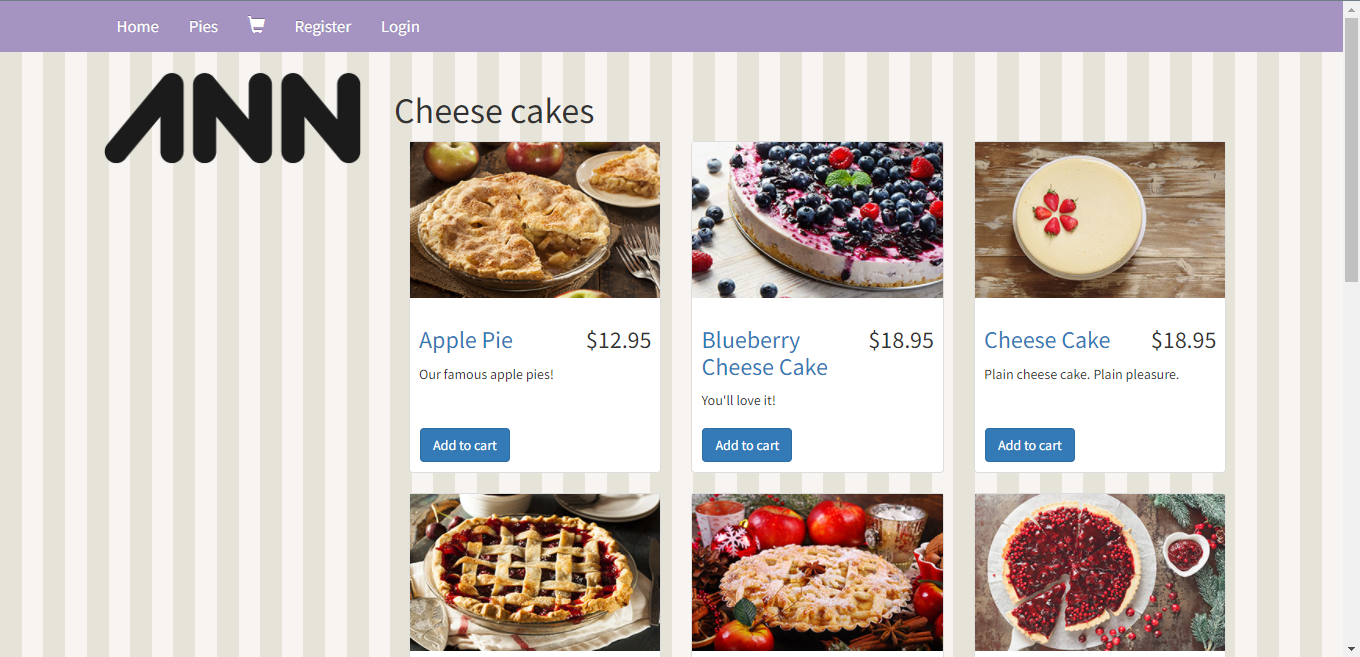


**5. Implementacija**

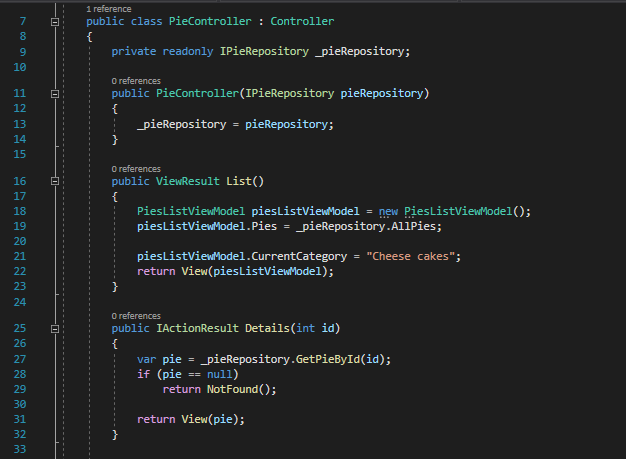
Kada se pokrene projekat na pocetnoj strani vidimo meni(Home, Pies, Korpu, Registraciju I Login).



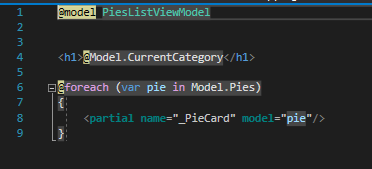
Na slici ispod se vide sve ponudjene pite koje korisnik moze da poruci.



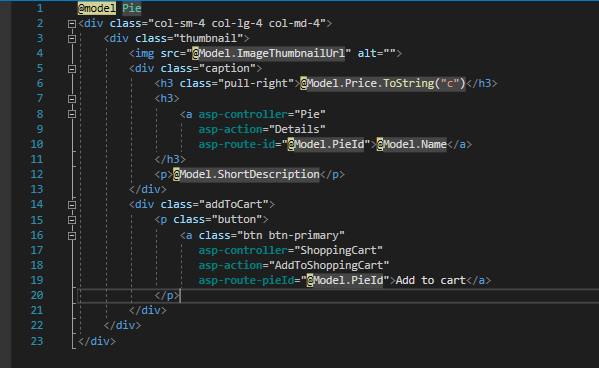
U kodu to izgleda bas kao na slici ispod. Nas kontroler ce kreirati I popuniti PiesListViewModel. Iz PiesListViewModel uzima listu I popunjava je preko metode koja se nalazi u IPieRepodisitory a koja je implementirana u PieRepository(gde iz baze vraca sve Pie proizvode u listu).



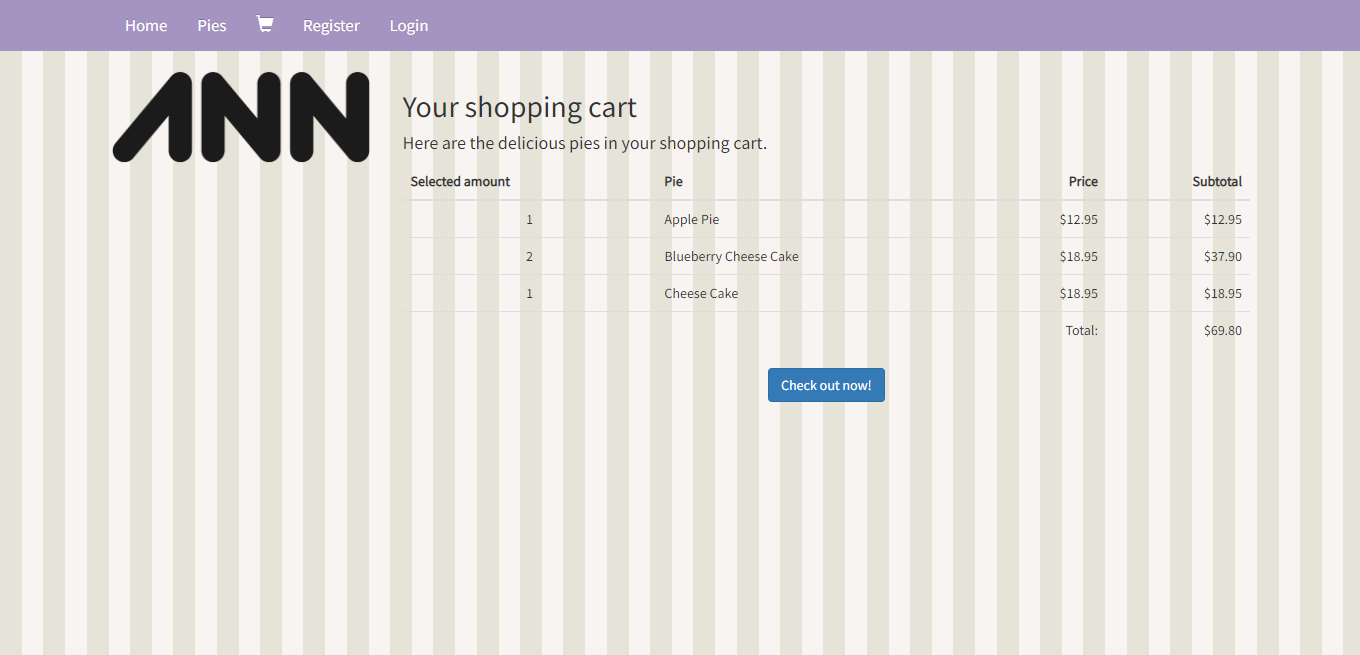
Nakon sto se popuni lista iz PiesListViewModel taj objekat se prosledjuje u List.cshtml kao sto vidimo na slici. Zatim se prolazi pomocu foreach petlje prolazi kroz listu iz PIewListViewModel i ubacuje se ovaj partial \_PieCard .



\_PieCard nam omogucava prikaz pojedinacno svakog Pie iz liste. Kada korisnik klikne da doda neki proizvod pokrece nam se kontroler ShoppingCart I metoda u njemu AddToShopingCart() I prosledjuje se Id tog Pie-a koji korisnik zeli da doda.

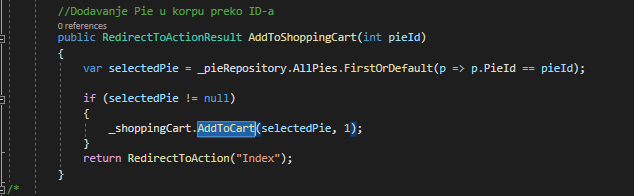


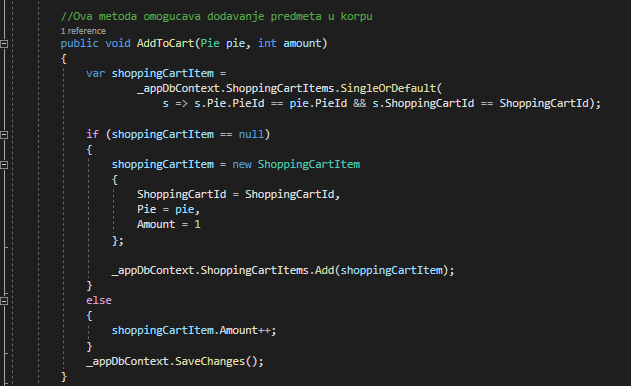
Kada korisnik klikne na korpu program mu vraca sve pite koje je korisnik dodao. Ispod ove slike se nalazi kako te izgleda u kodu.



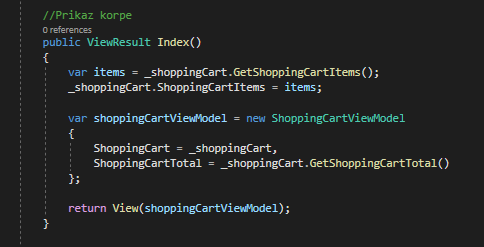
Metoda AddToshoppingCart() prima pieId. Preko metode AllPies koja je implementirana u PieRepository vracamo prvi Pie na koji naidjemo sa tim ID-om.

Ukoliko selektovani pie nije null onda ce se pozvati metoda AddToCart() kojoj prosledjujemom Pie koji smo nasli I kolicinu 1. Obasnjenje metode AddToCart() je ispod ove slike.

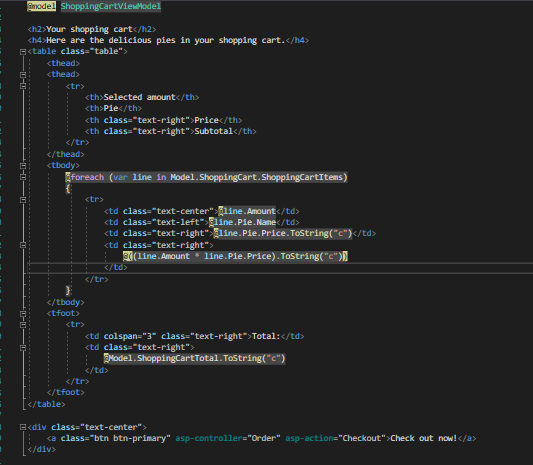


Metoda AddToCart() gde prosledjujemo pie I kolicinu. Ova metoda proverava u bazi u tabeli ShoppingCartItems da li u korpi vec postoji neki Pie sa tim ID-om I ShoppingCartID. Ukoliko nema ni jedan shoppingCartItem onda kreira novi ShoppingCartItem I dodeljuje mu ShoppingCartId, Pie koji smo prosledili I kolicinu 1 I to dodaje u bazi u tabeli ShoppingCartItems. U else delu ukoliko postoji vec ShoppingCartItem koji ima isti pieID I ShoppingCartId onda se kolicina samo povecava za 1. Sve promene u bazi sacuvamo. 

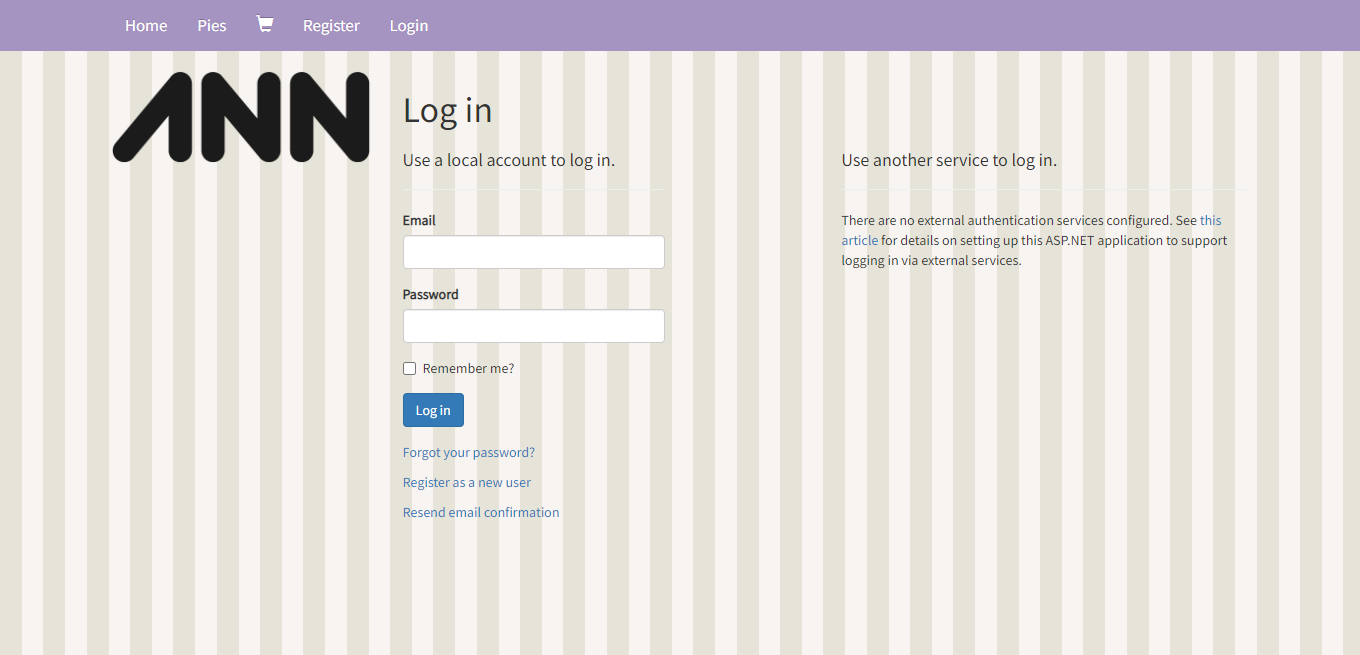
Index u ShoppingCartController-u nam omogucava da vratimo sve proizvode koji su dodati u korpu preko metode GetShoppingCart koja vraca listu svih proizvoda dodatih u korpu iz baze. Kreiramo ShoppingCartViewModel koji sadrzi nasu ShoppingCart i ShoppingCartTotal koji vrca ukupan iznos iz korpe preko metode GetShoppingCartTotal. Taj ShoppingCartViewModel prosledjujemo u Index stranici koja se nalazi u folderu ShoppingCart.



Model ShoppingCartViewModel nam omogucava prikaz pojedinacnog proizvoda iz korpe. Prolazimo kroz listu ShoppingCartItems I prikazujemo svaki proizvod koji je dodat.(Prikazujemo kolicinu, ime pite, cenu, i totalnu cenu).



Klikom na dugme “Check out now” korisniku se prikazuje login ukoliko nije prijavljen na aplikaciju.



Login je kreiran tako sto ASP.NET Identity(moci cemo da potvrdimo identitet I autorizujemo korisnike) je ugradjen u ASP.NET Core.

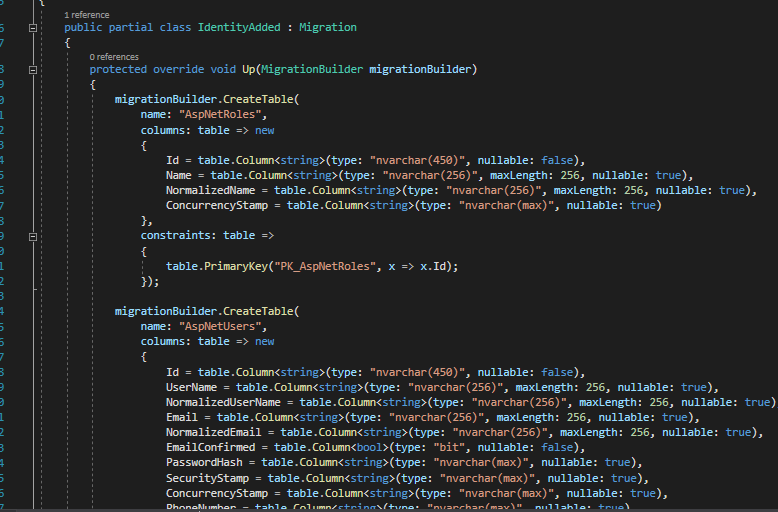
Potrebnu sam nam 2 biblioteke.



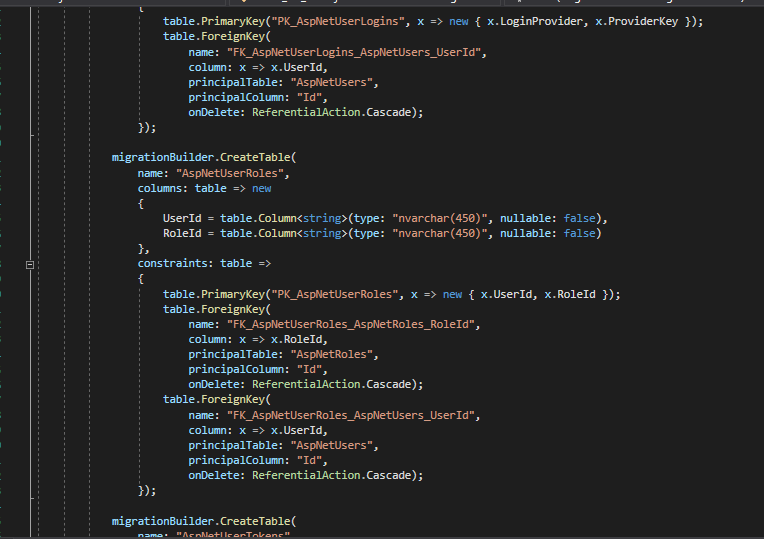


IdentityDbCOntext je osovna klasa za kontekst Entity Framework Core koja se koristi s Identitetom. A IdentityUser je ugradjena klasa koja se moze koristiti za predstavljanje korisnika u identity-u.

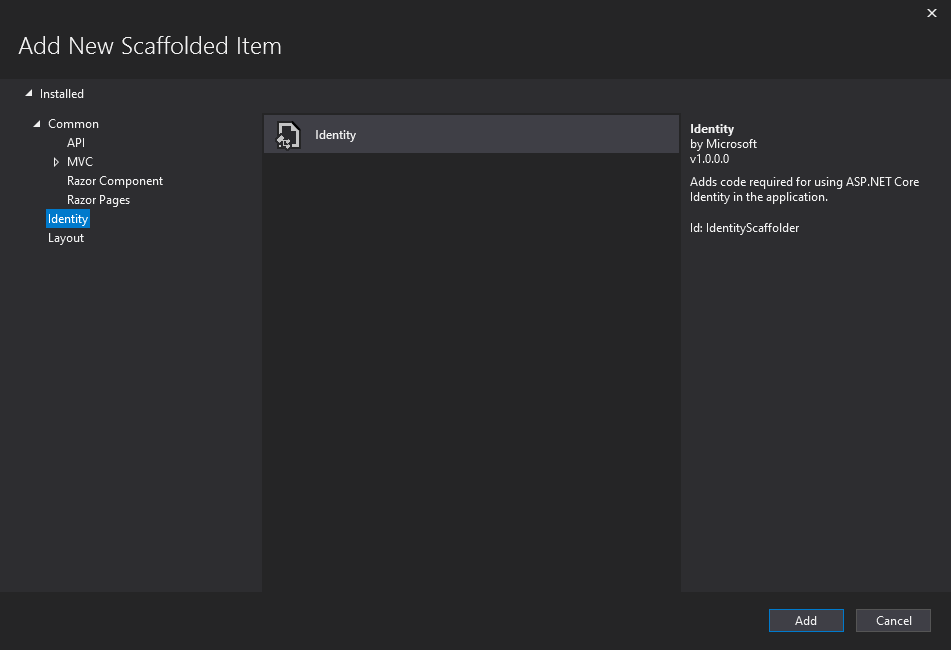
Preko migration kreiramo sve tabele u bazi potrebne za login tako sto u Package Manager Console pisemom add-migration IdentityAdded I kreirace nam sve potrebne tabele.

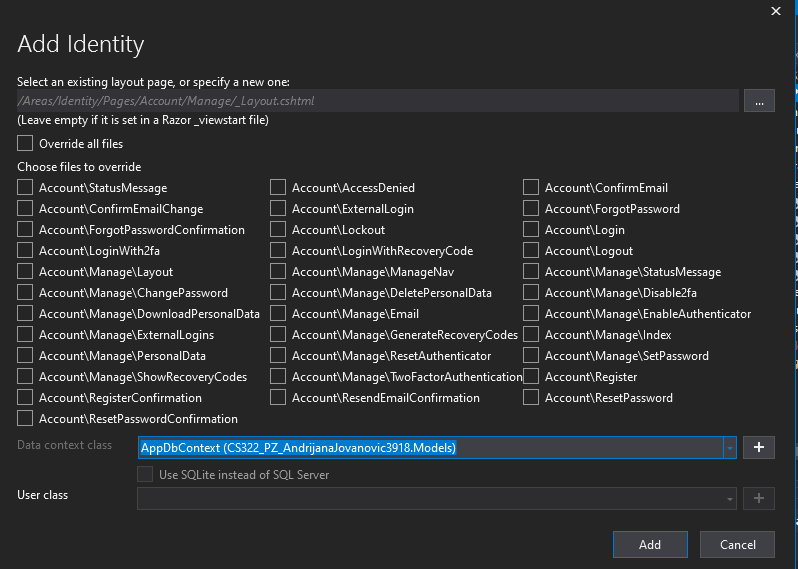




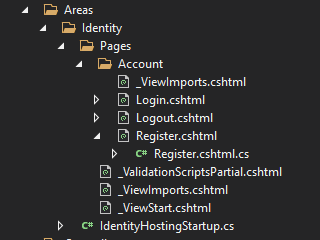


Zatim idemo desni klik na projekat Add- New Scaffolded Items. Navodimo datoteku koju zelimo da generisemo u nasu aplikaciju a onda mozemo da je menjamo(Izabrala sam Login, Logout, Register) I usmetavamo je na nas AppDbContext.

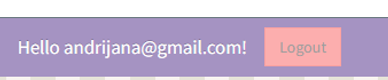




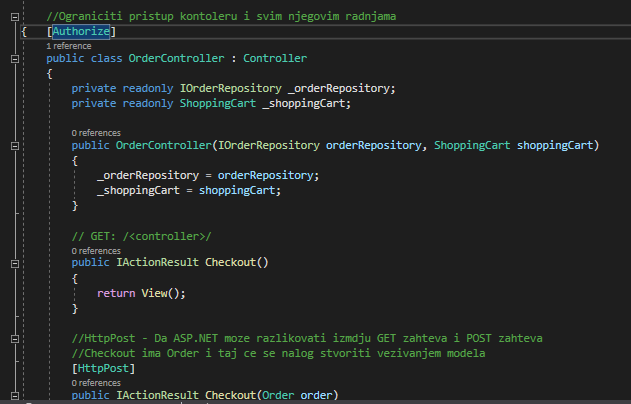
Kreira folder Areas. Datoteke skele generisace se pomocu okvira ASP.Net Core Razor Pages, unutar u Account vidimo prepisane datoteke.



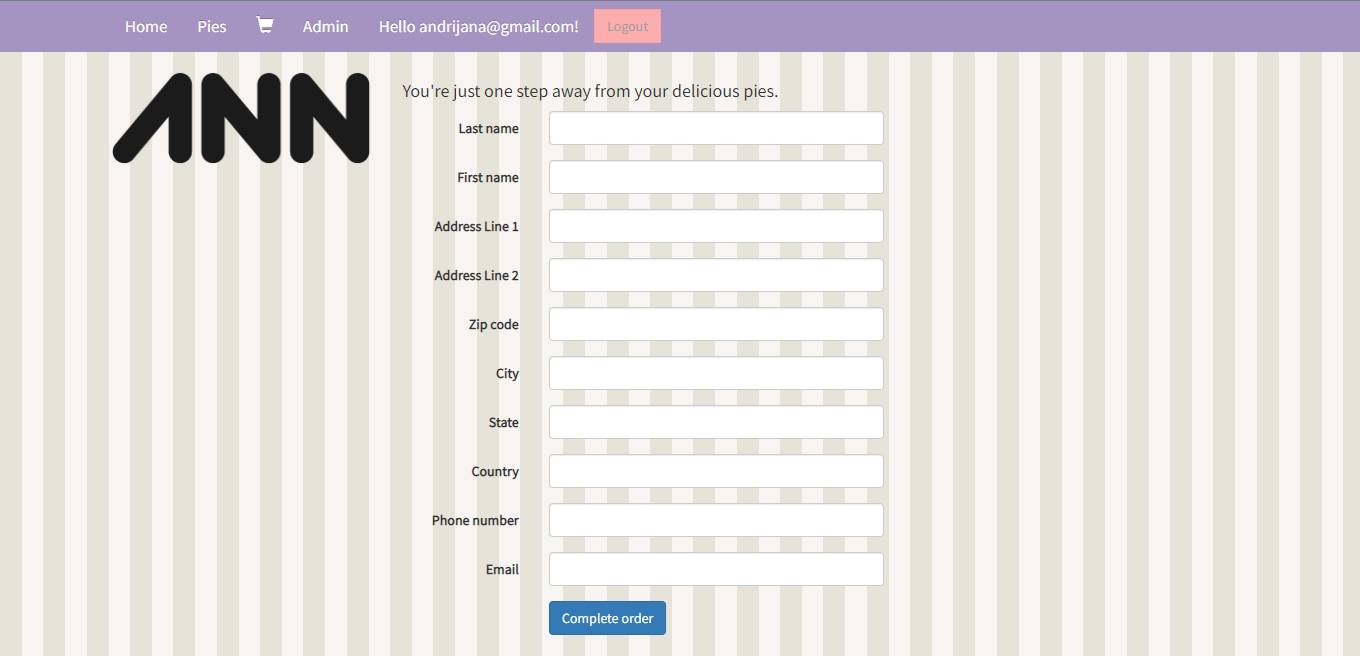
U Shared folderu I u \_LoginPartial proveravamo da li je korisnik prijavljen. Ukoliko je prijavljen moci ce da vidi dugme za odjavu i svoj profil.

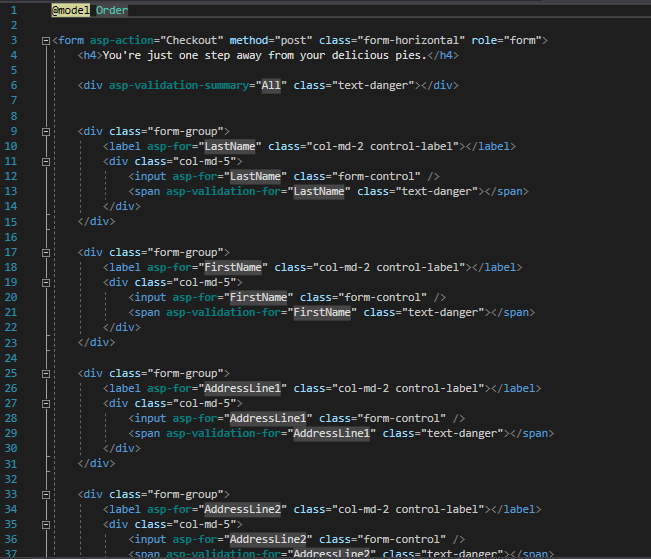


Takodje je potrebno u OrderPieControlleru ograniciti pristup [Authorize].

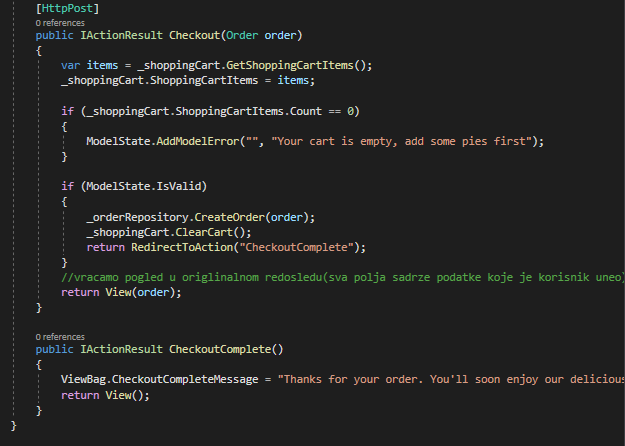


Nakon uspesne prijave korisniku se prikazuje forma sa poljima: Last name, First name, Adress Line 1, Adress Line 2, Zip code, City, State , Country, Phone number, Email.

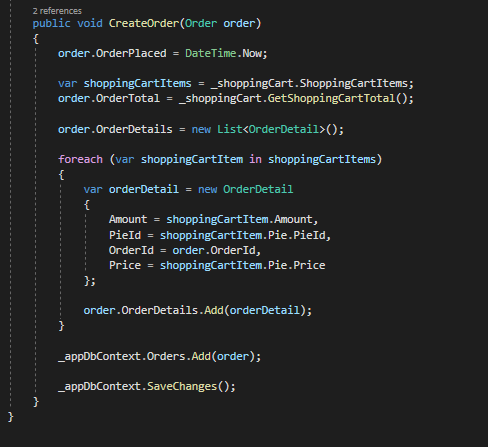




Kada korisnik popuni sve te podatke. Onda se Kreira Order na osnovu popunjenih podataka iz inputa. Klikom na dugme Complate order pokrece se metoda koja se nalazi u OrderController-u gde prosledjujemo nas Order. Zatim vracamo sve proizvode iz baze u items I dodeljujemo taj items nasoj listi koja se nalazi u ShoppingCart-u. Ukoliko je lista prazna vracamo poruku da je prazna. Ukoliko je validna onda kreiramo nasu porudzbinu preko metode CreateOrder(). Ta metoda je implementirana u OrderRepository. Zatim cistimo nasu korpu I usmeravamo na CheckoutComplete gde nam se prikazuje poruka o uspesnosti porucivanja.

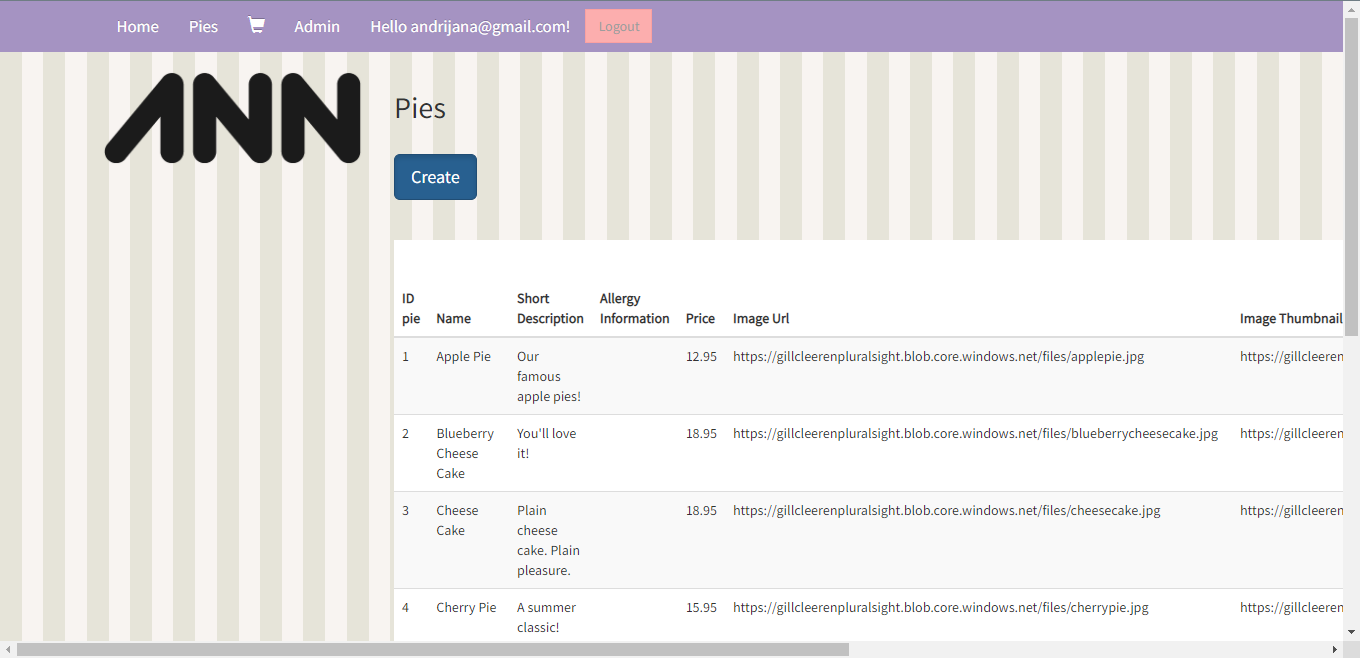


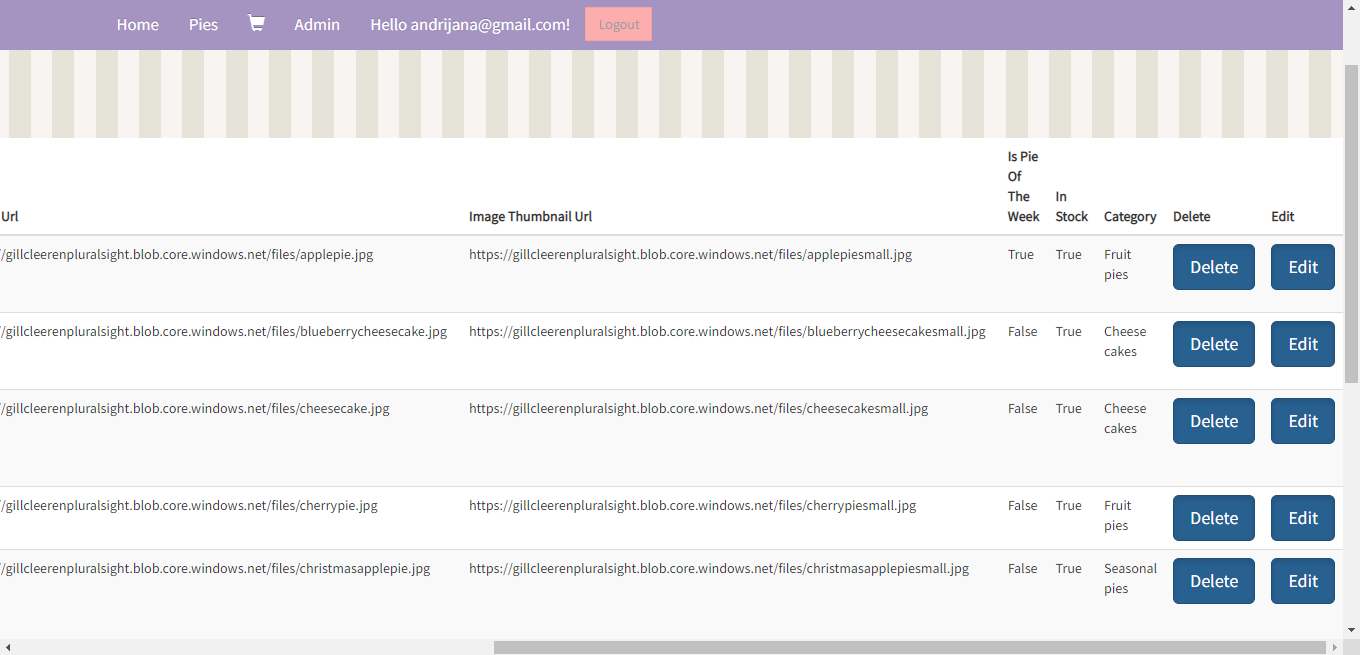
Metoda CreateOrder(Order order) nam dopunjuje nas order dodavanjem vremena i ukupne cene. Zatim prolazimo kroz nasu korpu I kreiramo orderDetail(popunjvamo kolicinu, PieId, OrderID, I cenu) I dodajemo u listu koja se nalazi u Order-u nas OrderDerail. Dodajemo kreirane OrderDetail u nasu bazu u tabelu Orders i sacuvamo sve izmene.



Ukoliko se uloguje admin u nav baru ce se prikazati dodatno dugme(“Admin”).

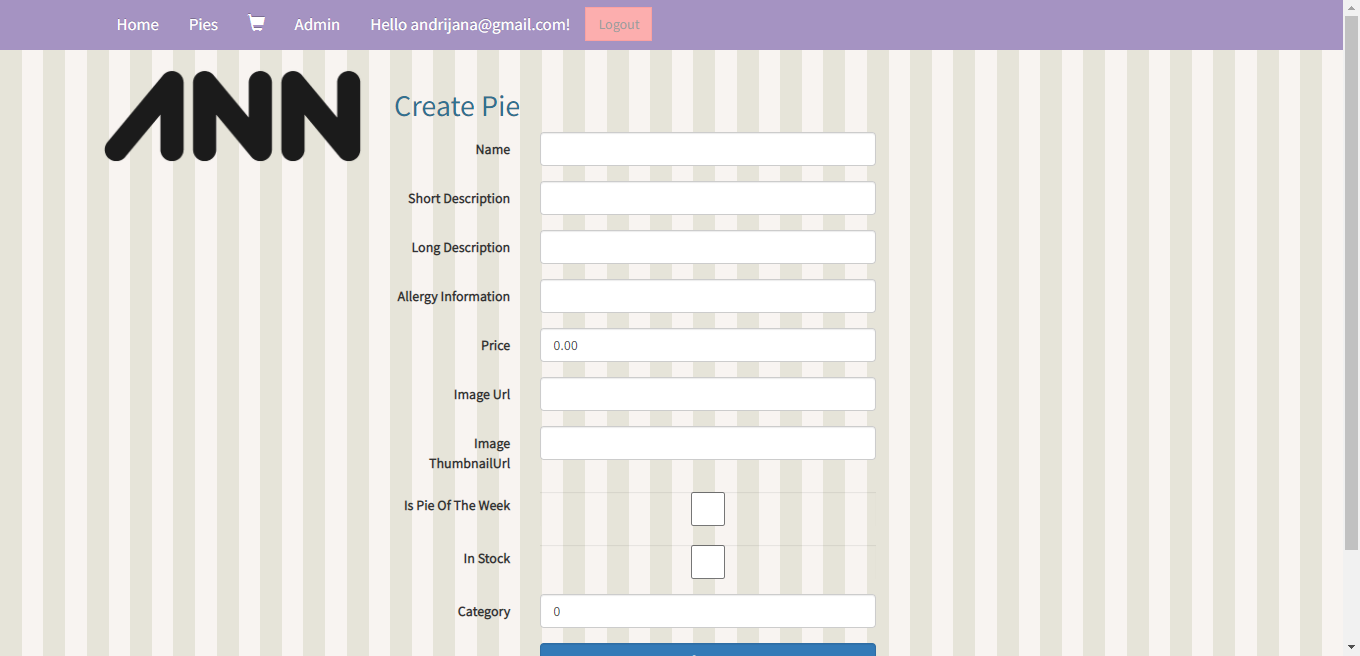
Kada admin klikne na to dugme moci ce da doda novu pitu, da obrise postojecu ili da izmeni.





Klikom na dugme “Create” adminu se prikazuje forma za kreiranje nove pite.

Admin popunjava ta polja i potvrdjuje dodavanje.



**7.Zakljucak**

Ova aplikacija svojom strukturom i načinom na koji je implemetirana predstavlja podlogu za bilo koji vid prodavnice ili poslovnog objekta koji se bavi trgovinom. Lako je prilagodljiva i njena funkcionalnost se može lako nadograditi. Koriscenjem MySQL baze omogucava lak prikaz svih pita i porudzbina zaposlenom. Kod je pisan u C# programskom jeziku primenjujuci gradivo iz predmeta „C# Programski jezik“, kao i naprednije principe programiranja.

**8.Literatura**

<https://docs.microsoft.com/en-us/aspnet/mvc/overview/older-versions/hands-on-labs/aspnet-mvc-4-entity-framework-scaffolding-and-migrations>

<https://www.c-sharpcorner.com/article/sql-server-database-connection-to-mvc-application/>

<https://www.youtube.com/watch?v=CzRM-hOe35o>

<https://docs.microsoft.com/en-us/aspnet/mvc/overview/getting-started/introduction/getting-started>