Projekat OOAD

**eCopy**

dokumentacija

Grupa: 9

Tim: Int elligence

Tema: eCopy

Članovi:

Ajla Panjeta

Berina Omerašević

Nudžejma Zukorlić

Sadržaj

[Opis 3](#_Toc516314752)

[Dijagrami 4](#_Toc516314753)

[Use Case dijagrami 4](#_Toc516314754)

[Dijagram klasa (MVVM) 6](#_Toc516314755)

[Dijagrami aktivnosti 7](#_Toc516314756)

[Dijagrami sekvenci 10](#_Toc516314757)

[Dijagrami komunikacija 11](#_Toc516314758)

[Dijagram komponenti 13](#_Toc516314759)

[UWP, ASP.NET i REST API 14](#_Toc516314760)

[Izvještaj o aktivnostima 20](#_Toc516314761)

[Tabela očekivanih i implementiranih funkcionalnosti 21](#_Toc516314762)

[Refaktoring 22](#_Toc516314763)

[Refaktoring pomoću design patterna 24](#_Toc516314764)

# Opis

Aplikacija eCopy je aplikacija koja omogućava svojim korisnicima (kupcima) da koriste usluge kopirnice bez potrebe da je posjećuju, tj. online. Naime, zahvaljujući ovoj aplikacije korisnik može putem svog uređaja (mobilnog telefona, laptopa i sl.), poslati dokumente za printanje, te odabrati način kopiranja koji želi (boja, format, uvez...). Pored usluga printanja na papiru, kopirnica nudi i druge usluge kao što je pravljenje personalizovanih šoljica, majica, termos boca, naljepnica te plakata, na korisniku je samo da putem ove aplikacije odabere željeni predmet, boju, natpis i/ili sliku), a kopirnica(odnosno radnik kopirnice) ima zadatak da obavi ono sto je korisnik tražio i pošalje mu proizvod na kućnu adresu (ukoliko je on to odabrao). Kopirnica također nudi usluge izrade reklamnih materijala za firme u vidu printanja na raznim predmetima koje korisnik može odabrati.

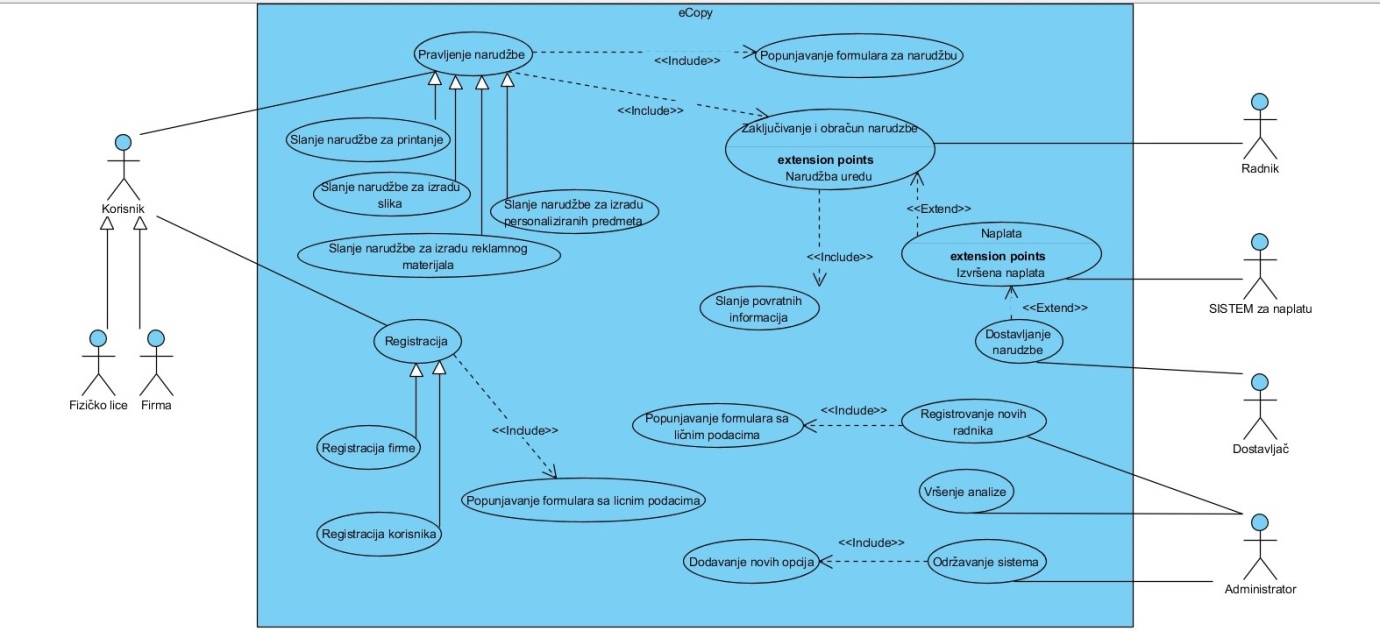
Korisnici aplikacije imaju mogućnost da kreiraju vlastiti account kako bi ostvarili razne pogodnosti koje kopoirnica nudi, ali i olakšali sebi način naručivanja jer ne bi morali svaki put navoditi svoje podatke i mogli bi izvršiti online plaćanje. Ukoliko ne žele kreirati nalog mogu se prijaviti i kao gost. Pored korisnika (kupaca) i radnici kopirnice imaju svoj account na aplikaciji kako bi mogli primati narudžbe, te slati povratne informacije klijentima.

Konačan projekat se nalazi na GitHub-u unutar foldera Projekat.

# 

# Dijagrami

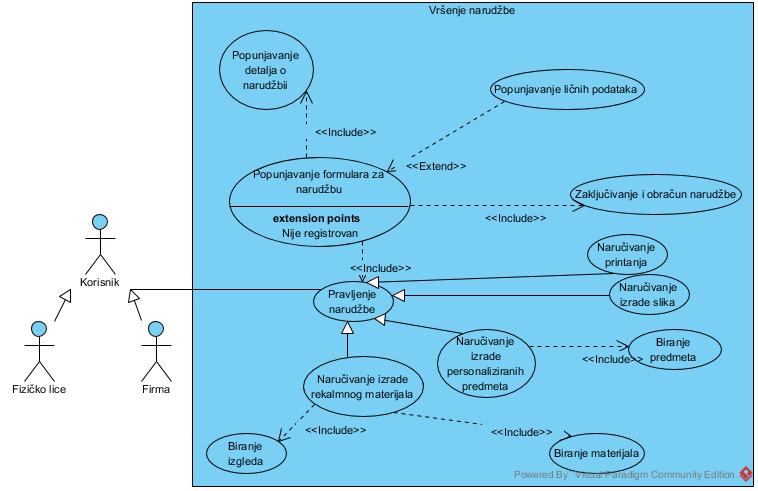
## Use Case dijagrami



Slika 1. Glavni Use Case dijagram



Slika 2. UseCase za registraciju korisnika

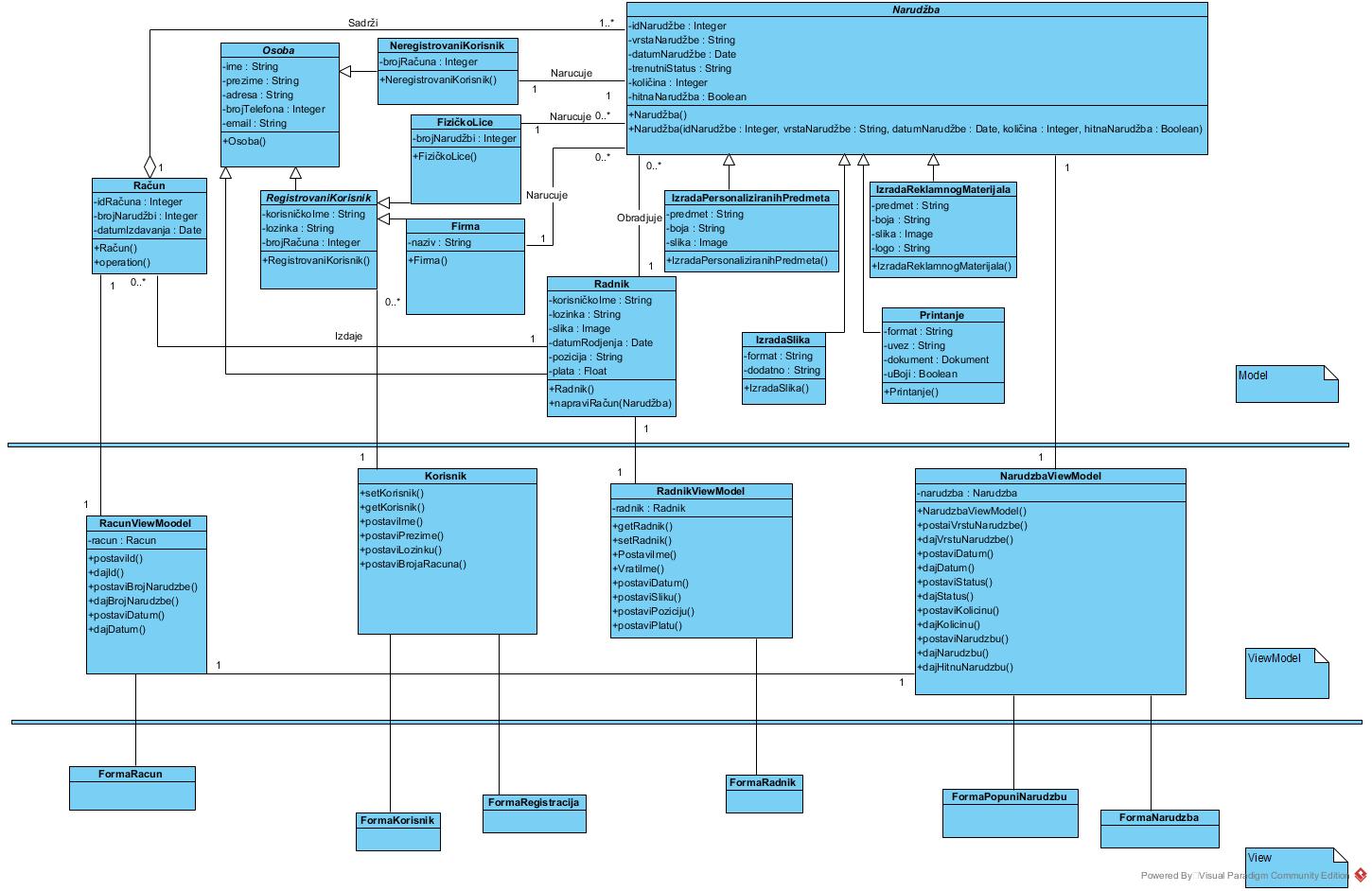


Slika 3. UseCase dijagram za obavčjanje narudžbe



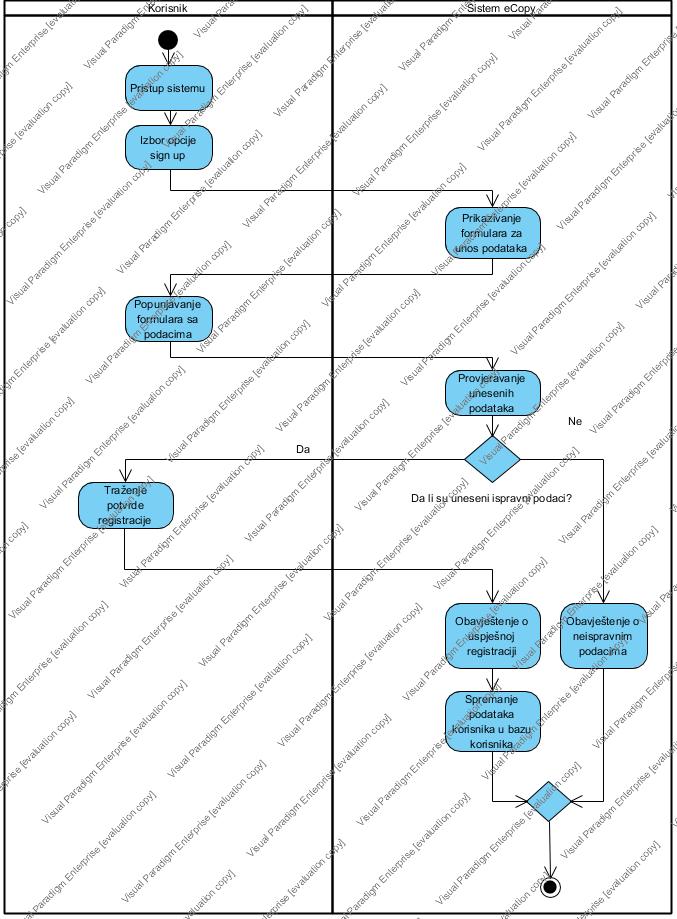
Slika 4. UseCase za obradu narudžbe

## Dijagram klasa (MVVM)

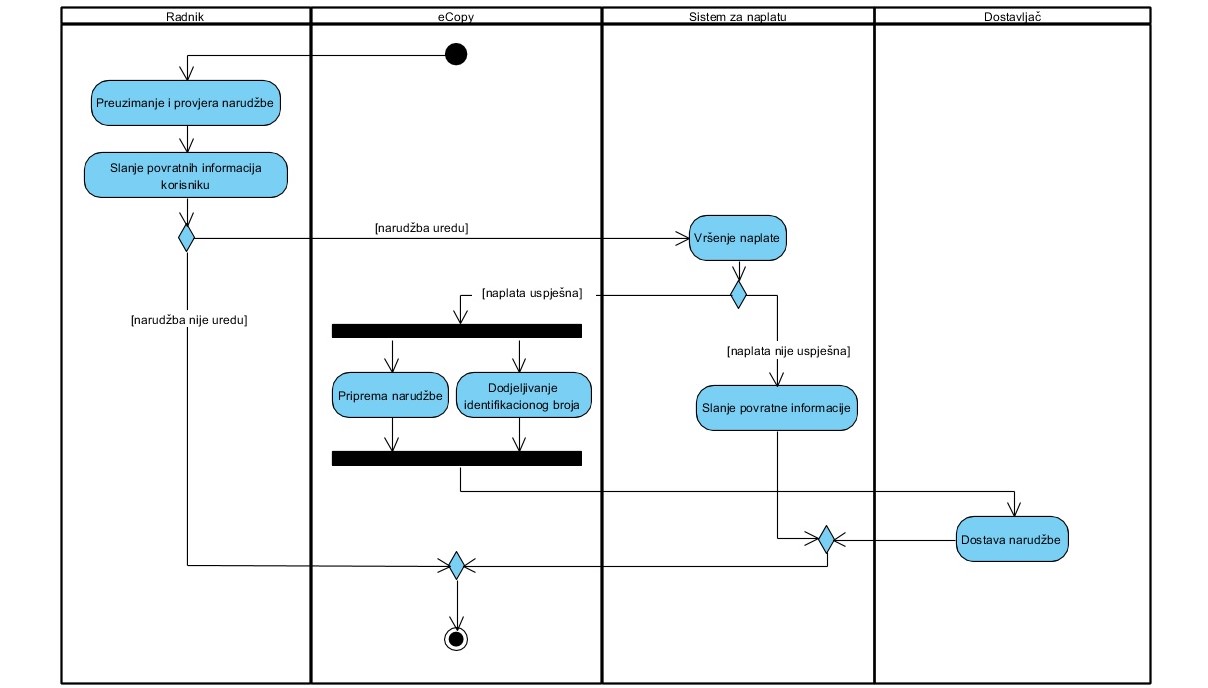


Slika 5. Dijagram klasa

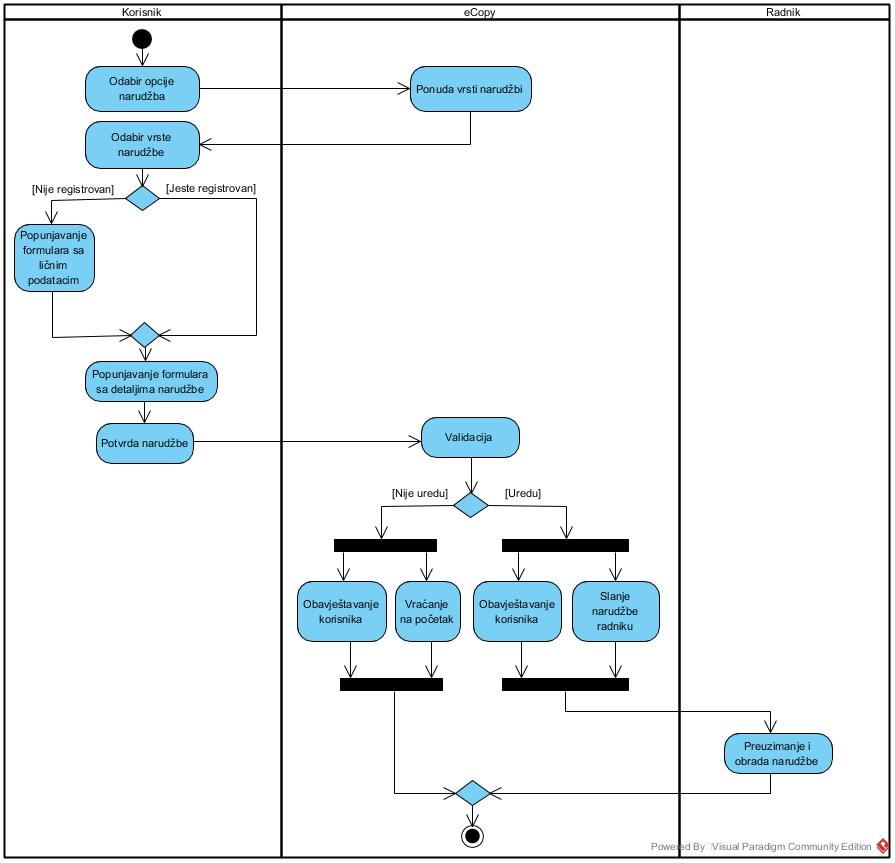
## Dijagrami aktivnosti



Slika 6. Dijagram aktivnosti za registraciju

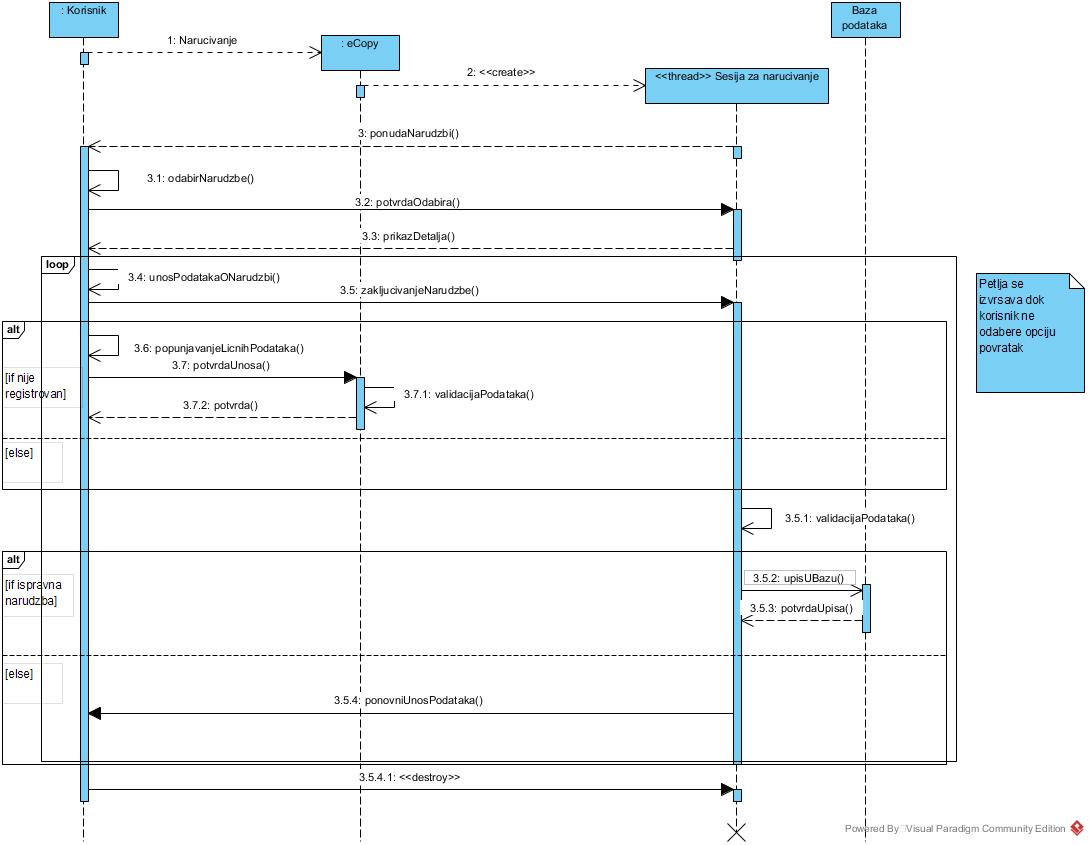


Slika 7. Dijagram aktivnosti za zaključivanje narudžbe

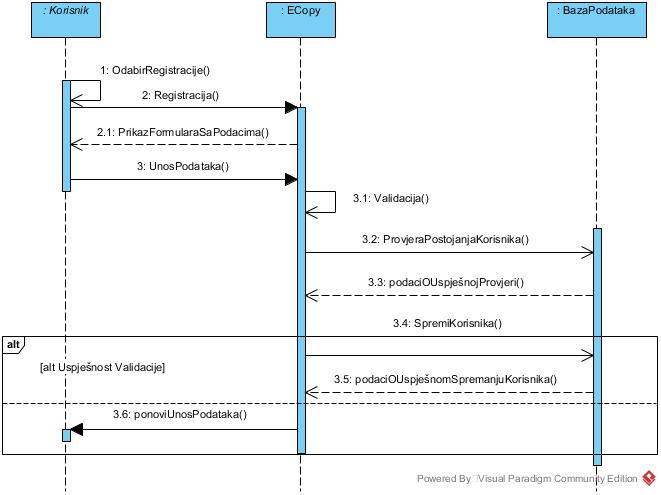


Slika 8. Dijagram aktivnosti za obavljanje narudžbe

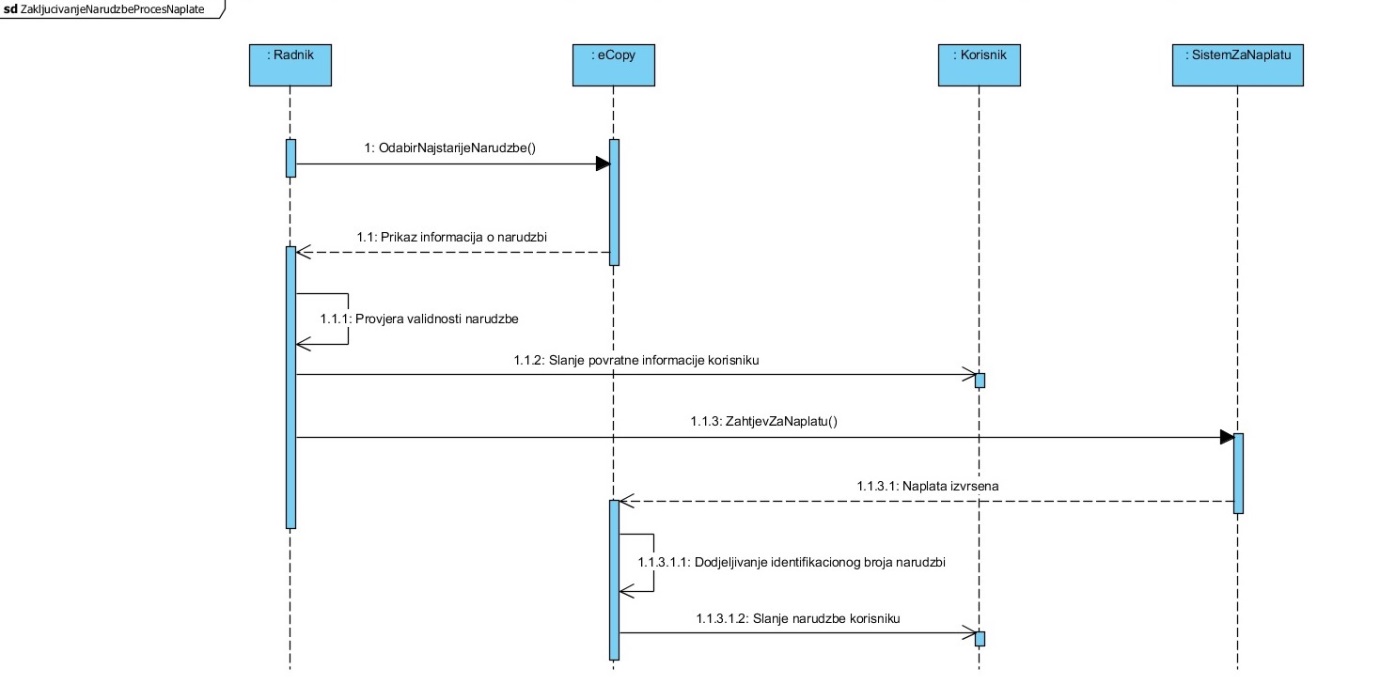
## Dijagrami sekvenci



Slika 9. Dijagram sekvenci za obavljanje narudžbe

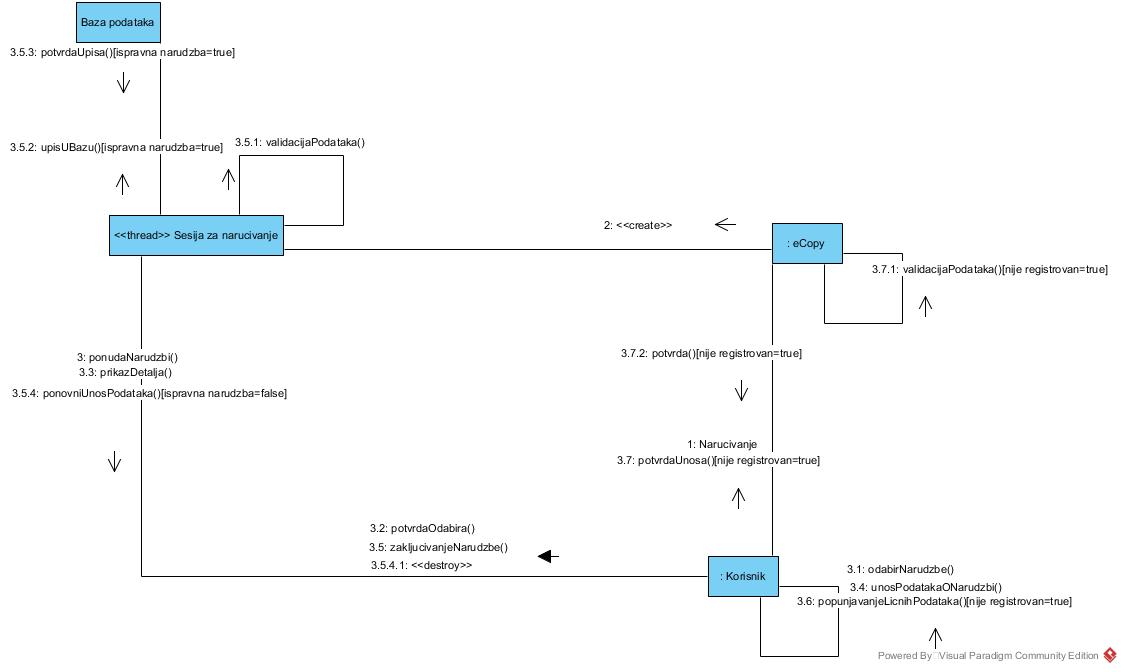


Slika 10. Dijagram sekvenci za registraciju korisnika

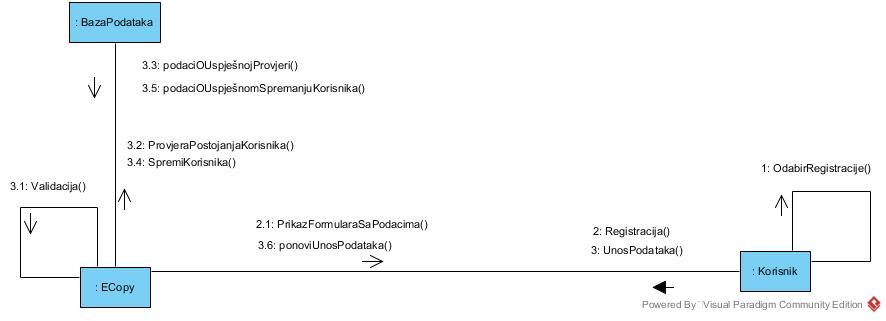


Slika 11. Dijagram sekvenci za proces naplate narudzbe

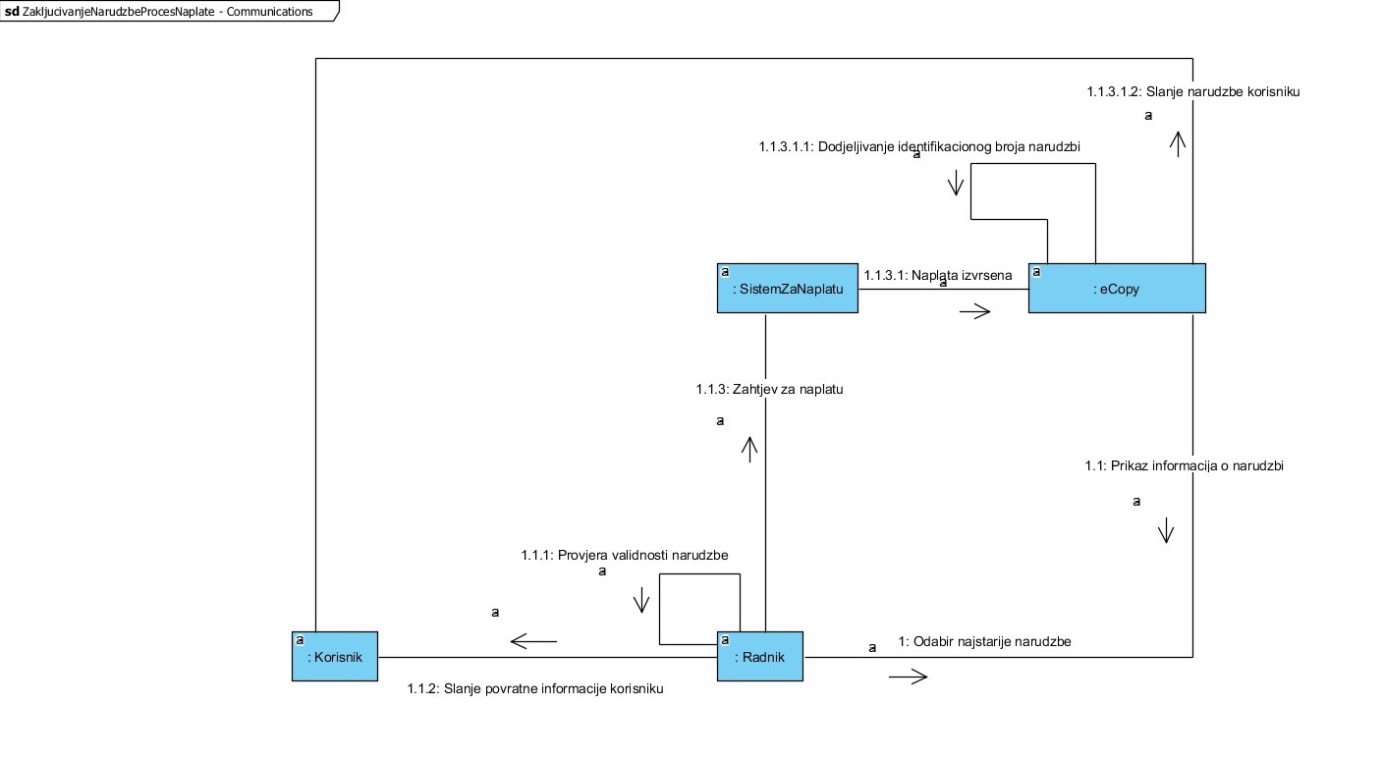
## Dijagrami komunikacija



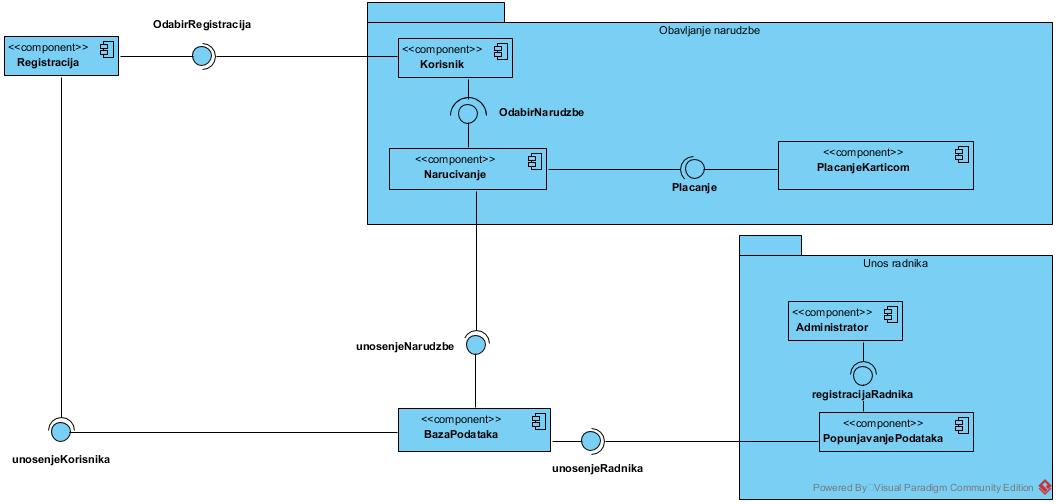
Slika 12. Dijagram sekvenci za obavljanje narudžbe



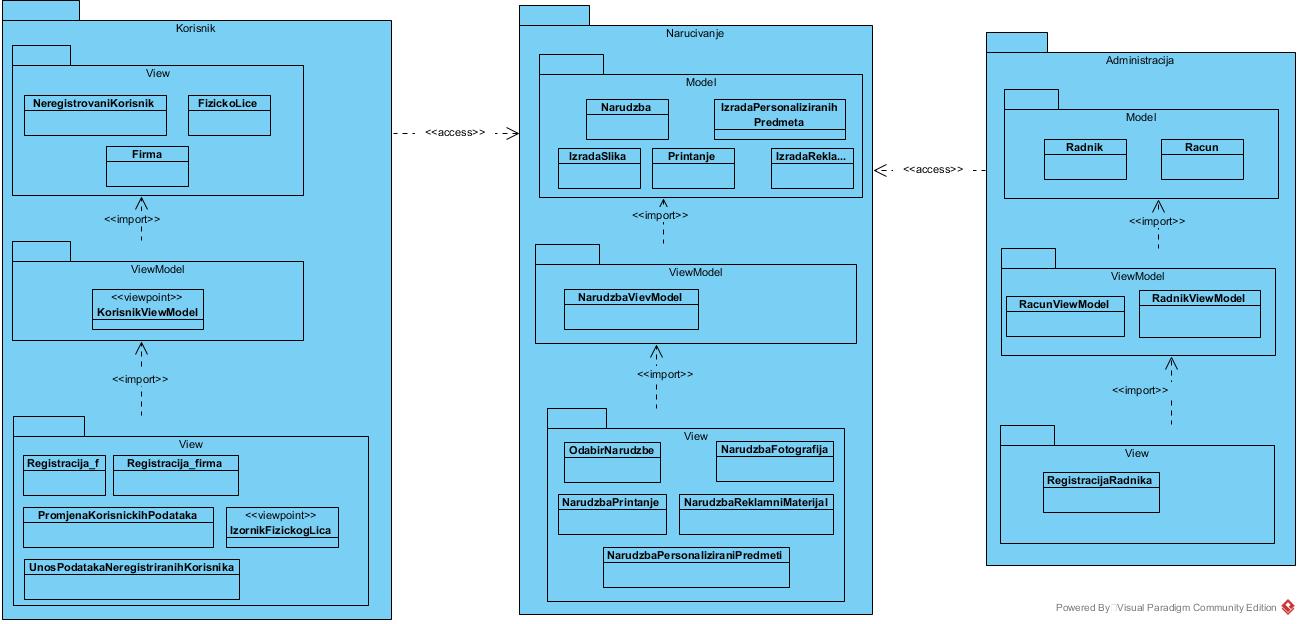
Slika 13. Dijagram komunikacija za registraciju korisnika

Slika 14. Dijagram komunikacija za proces naplate

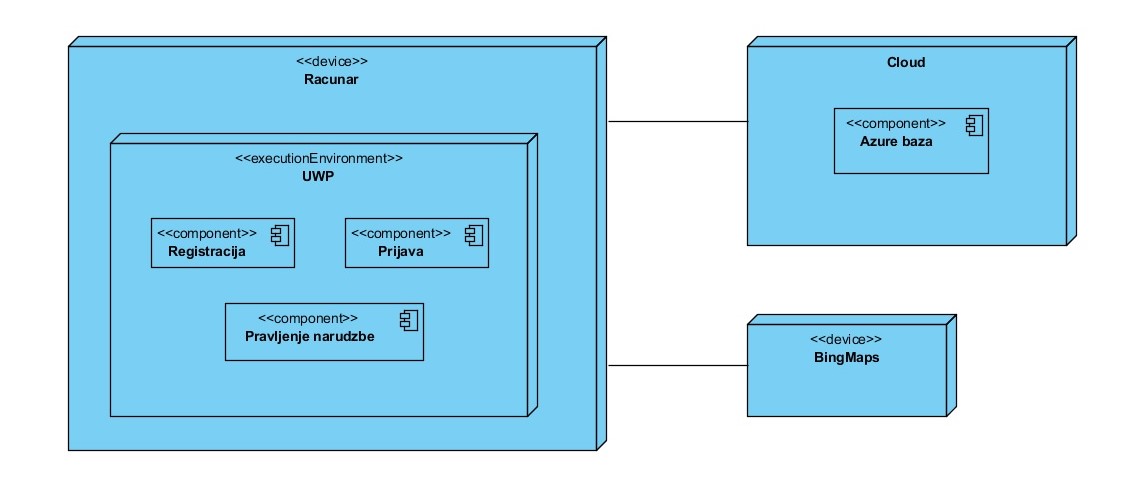
## Dijagram komponenti



Slika 15. Dijagram komponenti



Slika 16. Dijagram paketa

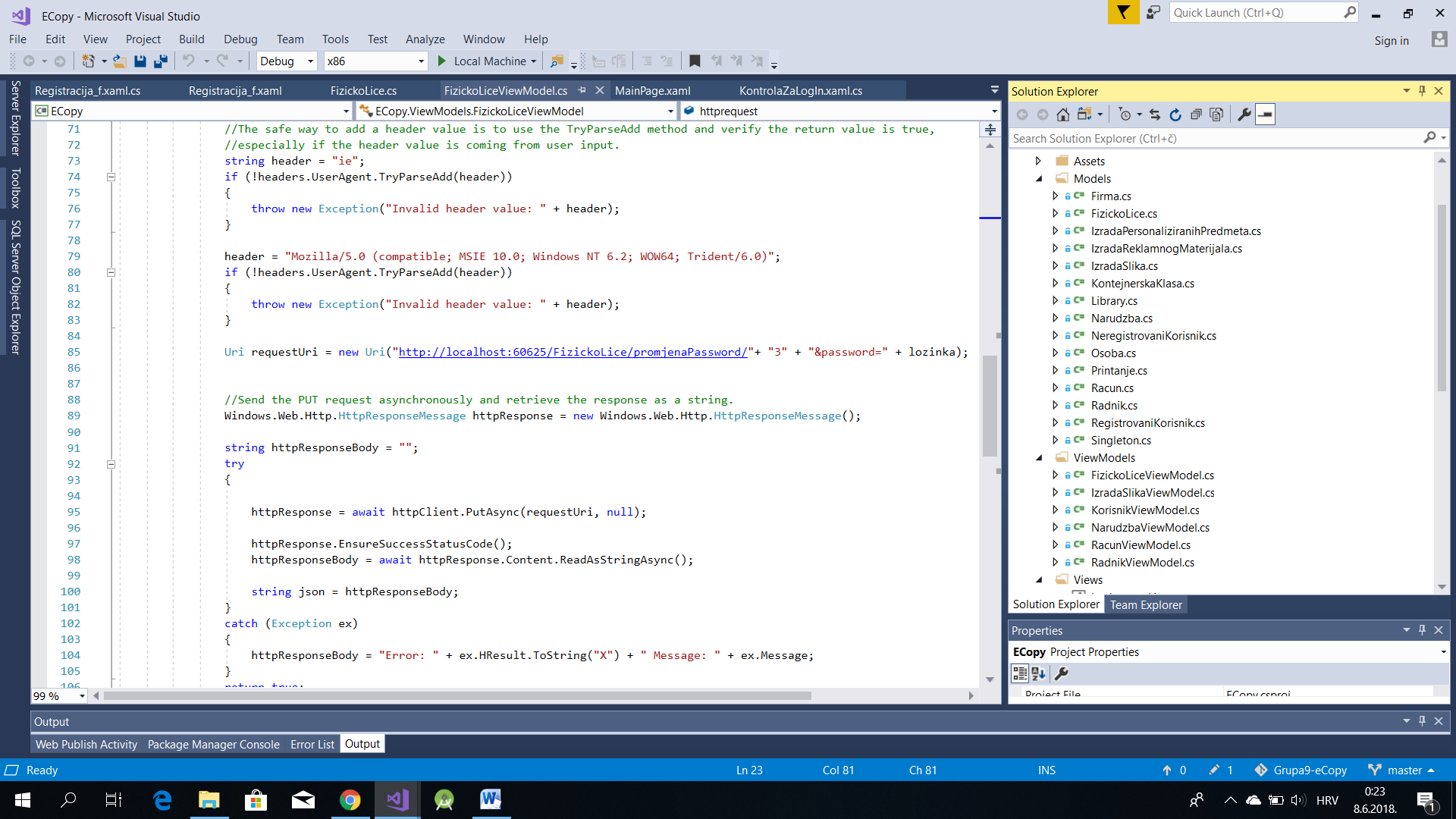
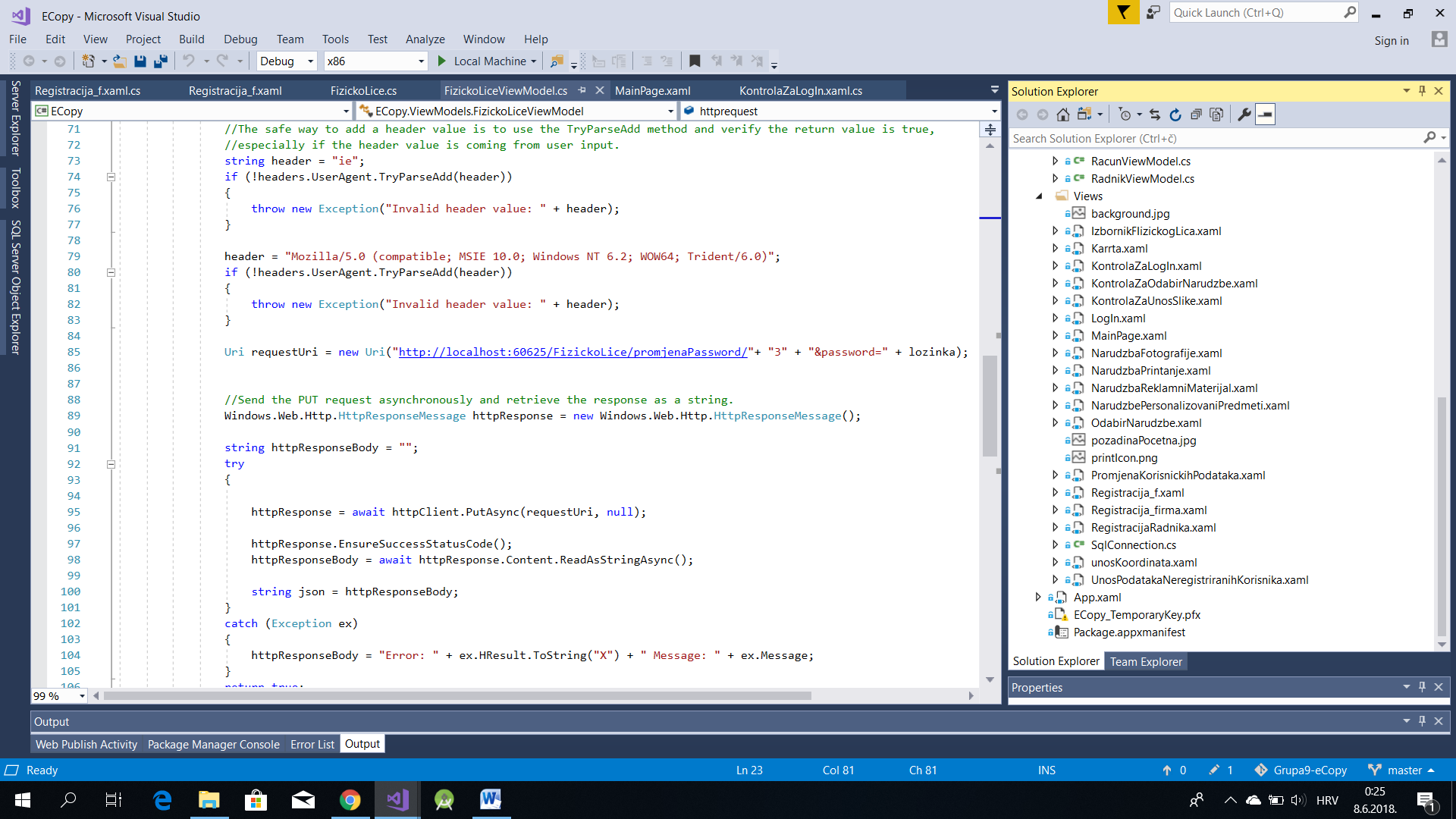


Slika 17. Dijagram raspoređivanja

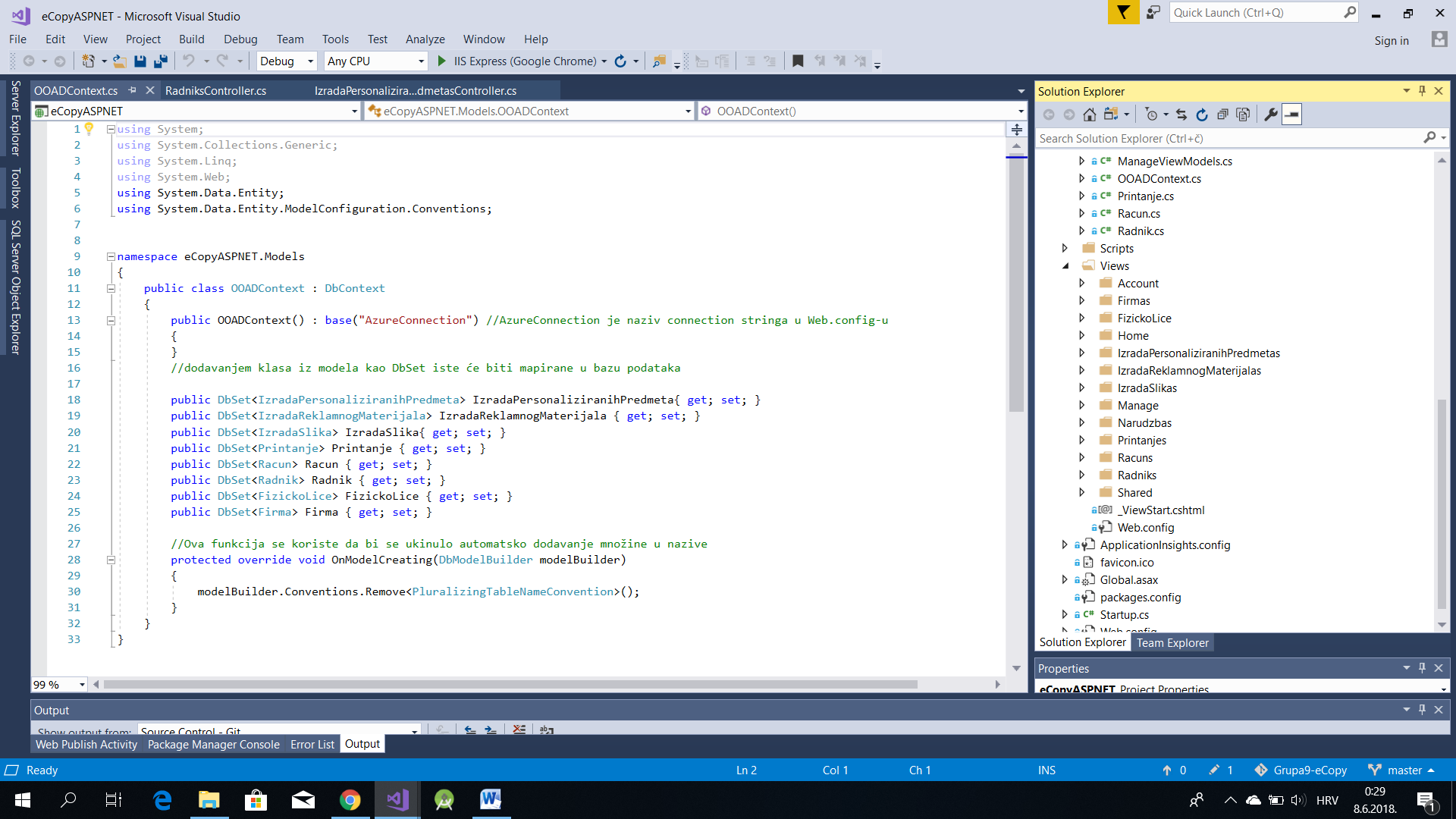
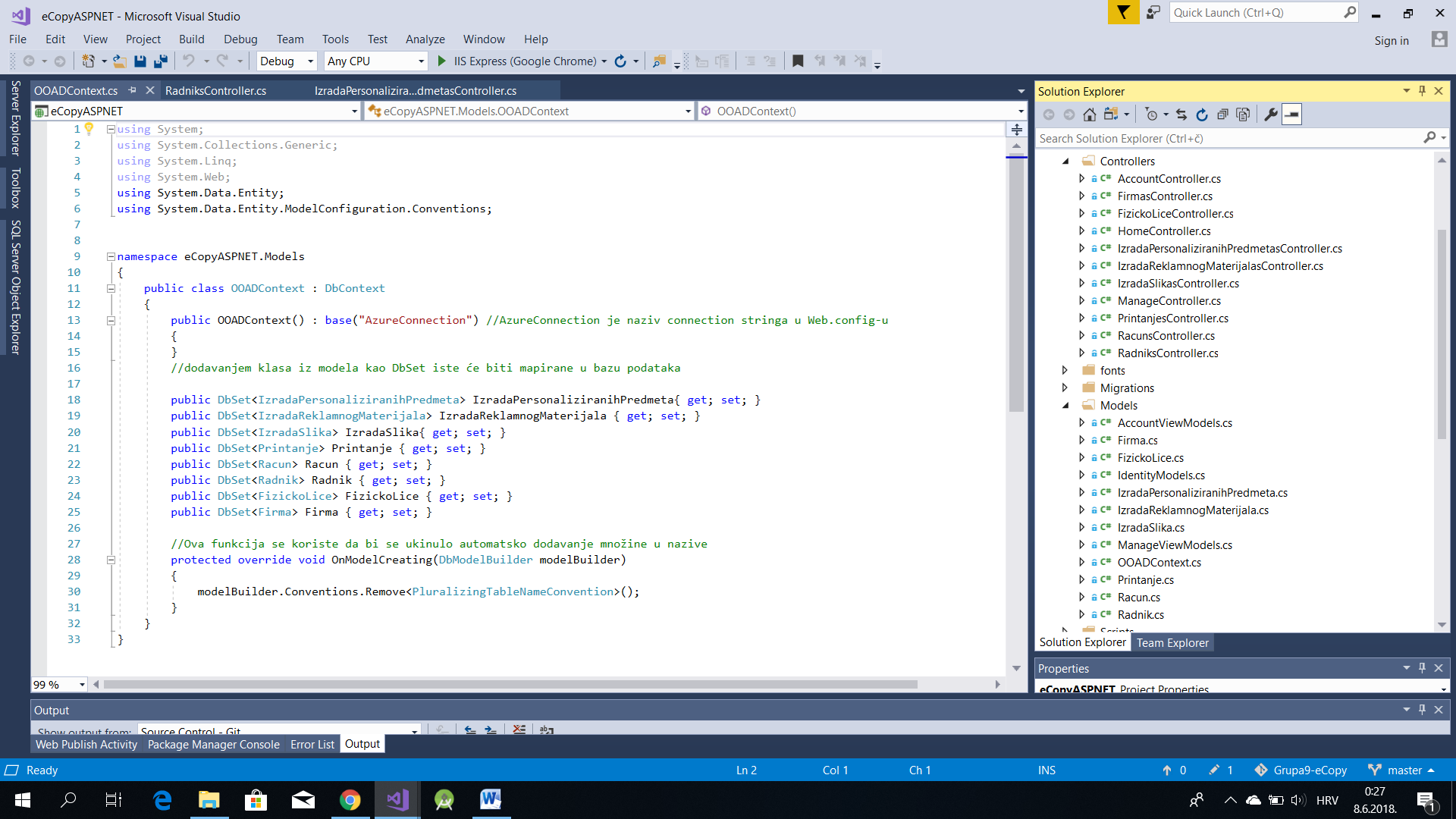
# UWP, ASP.NET i REST API

Napravljene sve potrebne klase i view-ovi za rad aplikacije u uwp-u.

Implementiran je i adaptivni interfejs tako da se aplikacija može pokretati na raznim uređajima.

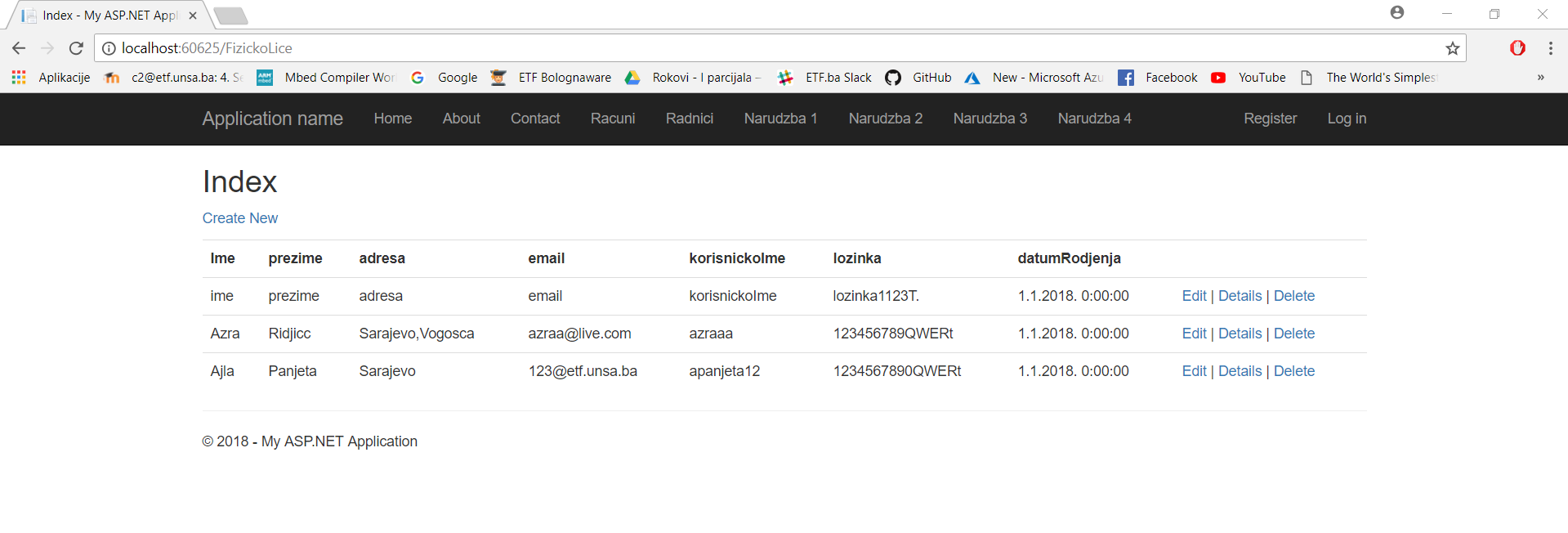
 

Napravljeni svi modeli, view-ovi i controleri u asp.net-u (postovana MVC arhitektura)

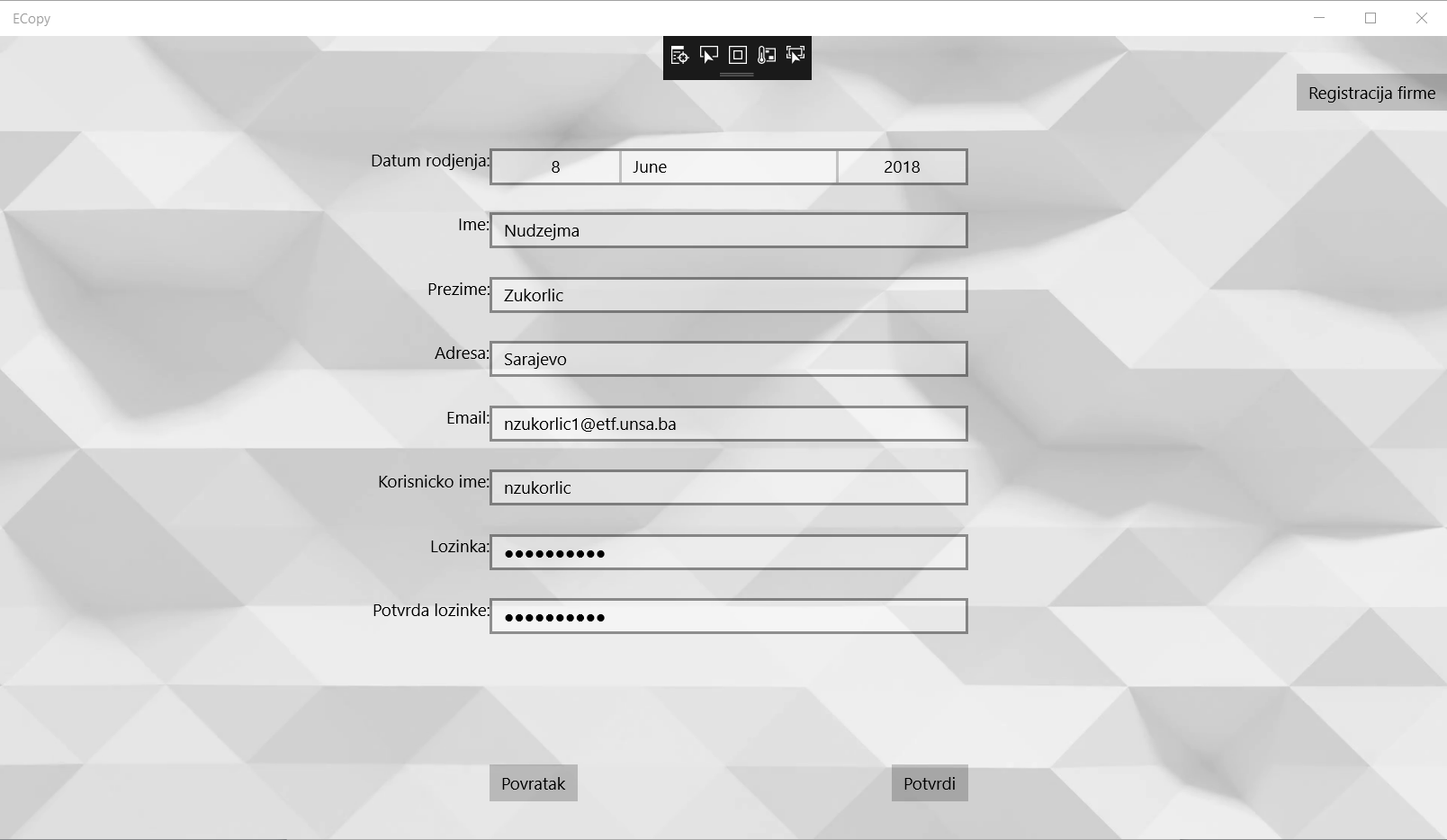


Povezano na bazu eCopyBaza na serveru ooad2018ecopy (na Azuru)

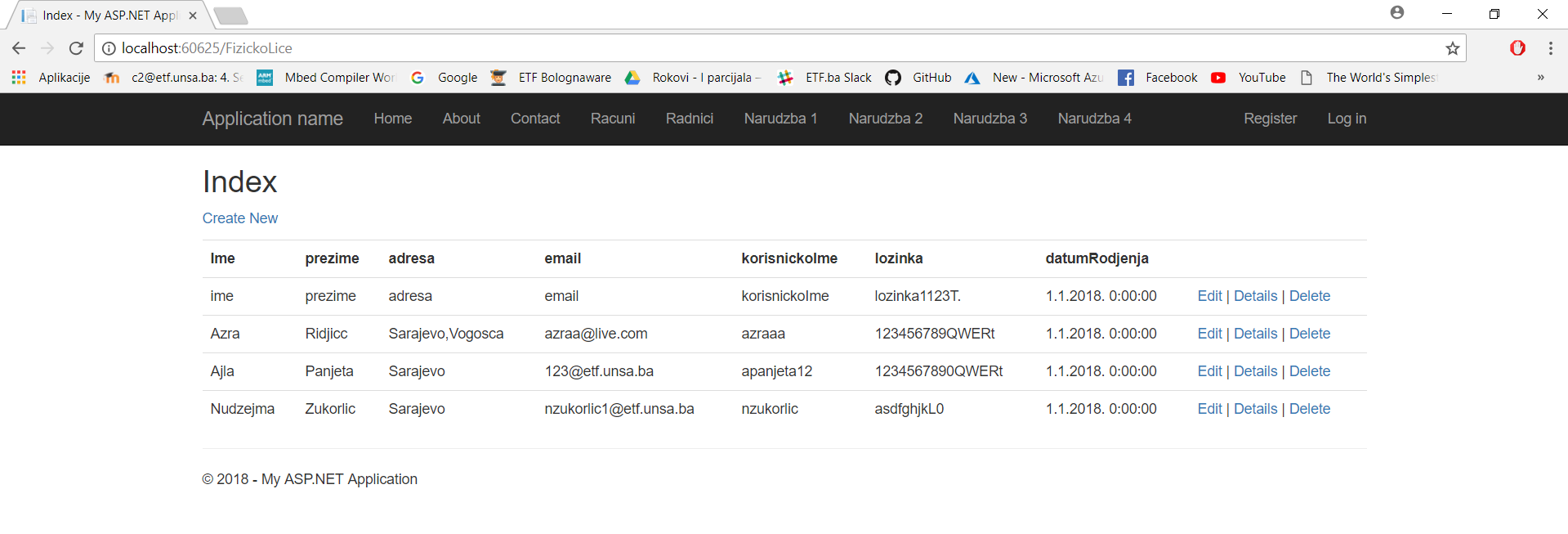
Unos podataka u tu bazu može se vršiti pomoću UWP aplikacije korištenjem http requesta (Potrebno je da bude pokrenu ASP.NET u pozadini)



Slika 18. Podaci u bazi prije unosa

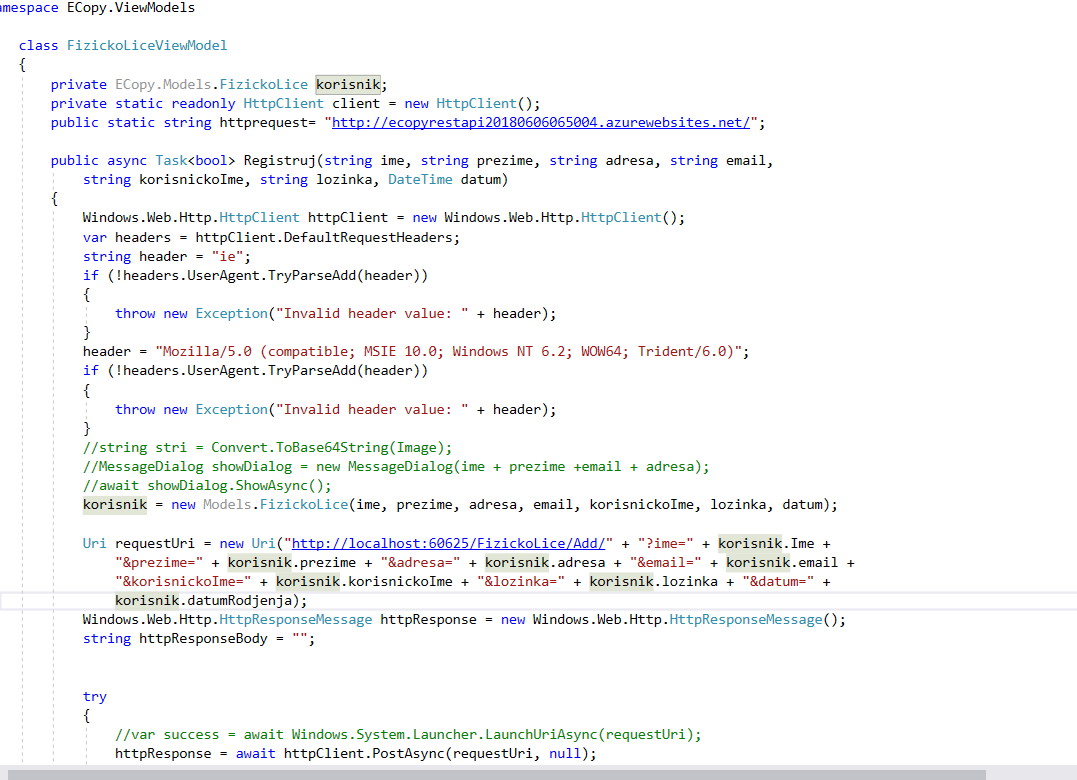


Slika 19. Unos u UWP-u



Slika 20. Baza nakon unosa

Dio koda koji obavlja ovu funcionalnost nalazi se u ViewModelu FizickoLiceViewModel u metodi registruj



Slika 21. Naznaceni dio sa http requestom

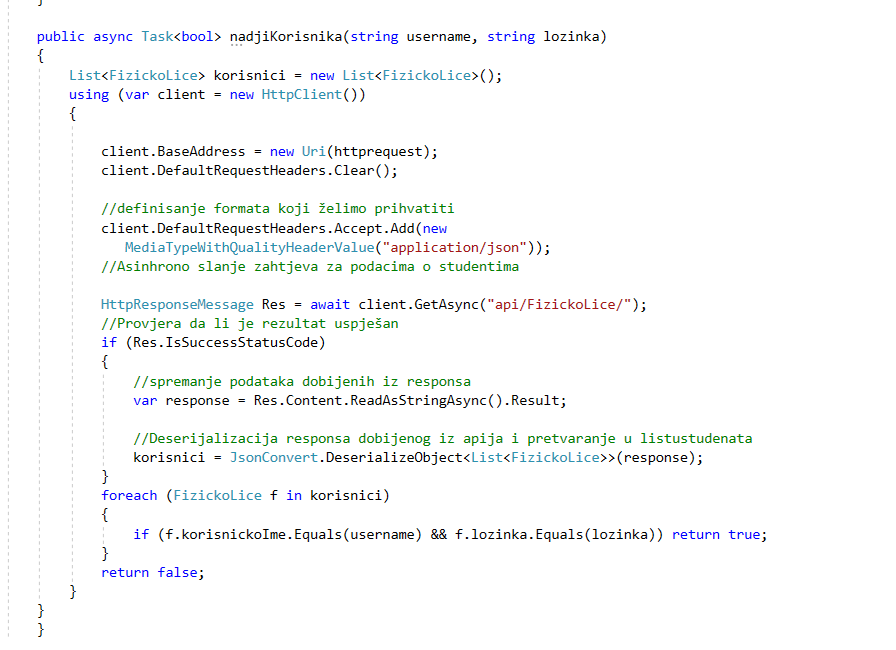
Na isti način je realizovano: unosenje Firme (druga vrsta korisnika), unosenje radnika, unosenje narudžbe (trenutno je implementirano samo za jednu vrst narudžbe, narudzbaSlika)

Implementiran ASP.NET web api service (napravljen je novi projekat eCopyRestAPI)

Tu su ponovo dodani kontroleri, modeli i view-ovi za sve klase, te za svaku klasu omogućene metode GET, POST, PUT i DELETE, pomoću kojih se može vršiti rad sa bazom (unošenje, čitanje, promjena...)

Omogućen i poziv WEB API servisa iz UWP aplikacije, za sve one klase (prethodno nabrojane) za koje je moguće unositi podatke, mogu se čitati podaci pomoću GET API-ja u UWP-u

Dio koda koji to pokazuje nalazi se u UWP-u unutar FizickoLiceViewModel klase (prikazano samo za fizičko lice u ovom slučaju iako je implementitrano i za ostale):



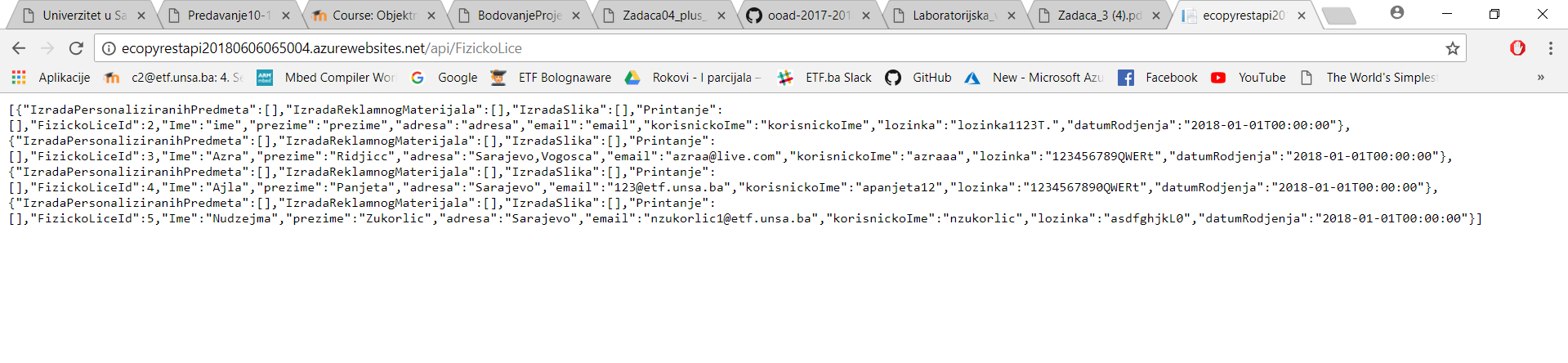
Slika 21.

String httprequest je u ovom slučaju

public static string httprequest= "http://ecopyrestapi20180606065004.azurewebsites.net/";

**Ovo je link koji se dobije nakon publish-a aplikacije na Azure.**

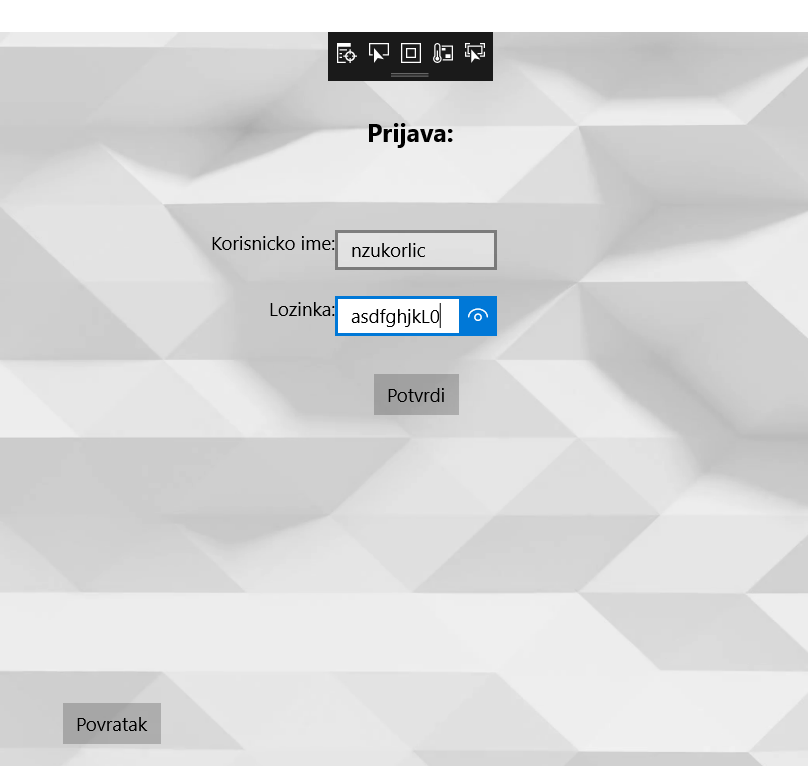
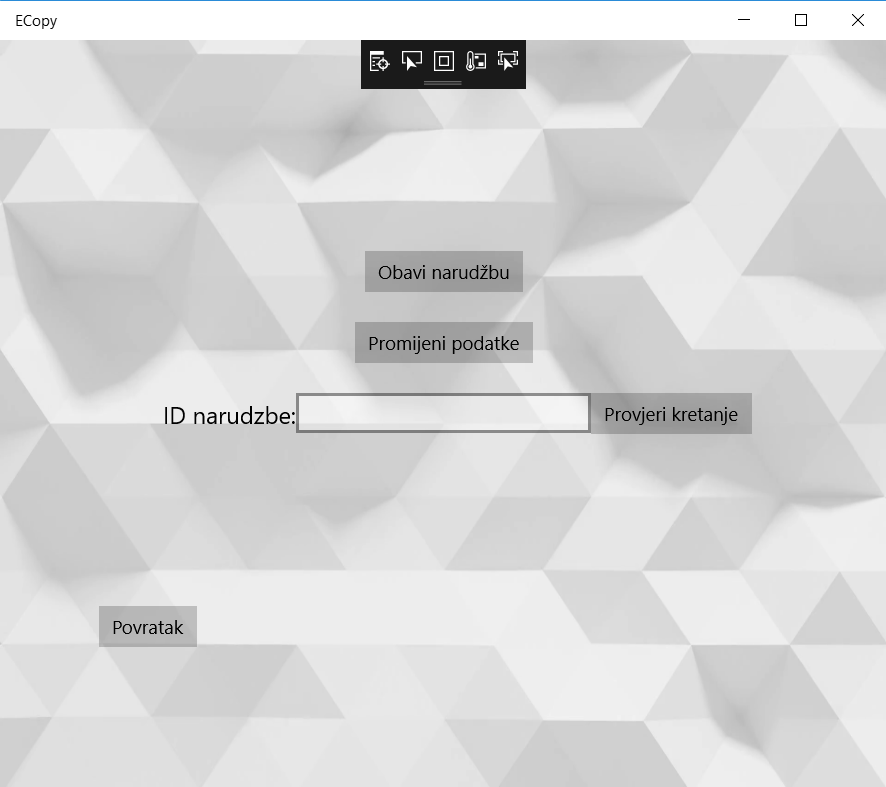
Možemo vidjeti da kucanjem ovog linka na web browser-u sa dodatkom api/FizickoLice dobijamo sve elemente te tabele u json formatu (Upravo se iz UWP-a vrši čitanje iz JSON formata)



Vidimo da se i uneseni podatak za FizickoLice (koji smo maloprije unijeli) automatski dodao i ovdje.

Kada se pokusamo u uwp-u prijaviti kao korisnik sa ovim podacima, otvara nam se view za korisnika, sto znaci da aplikacija ispravno nalazi podatki u pazi pomoću API-ja

Pogledajmo na sljedećim slikama:

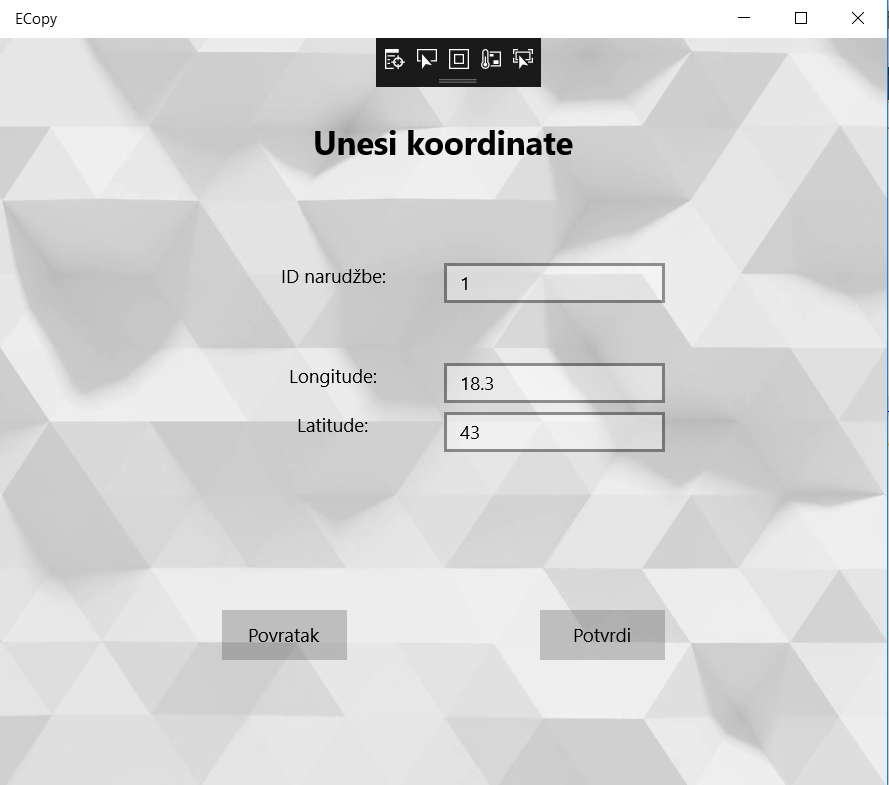
Slika 22. Prikaz login-a

Dakle aplikacija se sastoji iz UWP dijela, ASP.NET i RestApi (gdje je implementiran web service i uradjen publish aplikacije).

Sva tri dijela su povezana preko jedne baze, tako kad unesemo novog korisnika, radnika, narudzbu.. u UWP aplikaciji, te podatke možemo vidjeti u ASP.NETu i na linku <http://ecopyrestapi20180606065004.azurewebsites.net/>

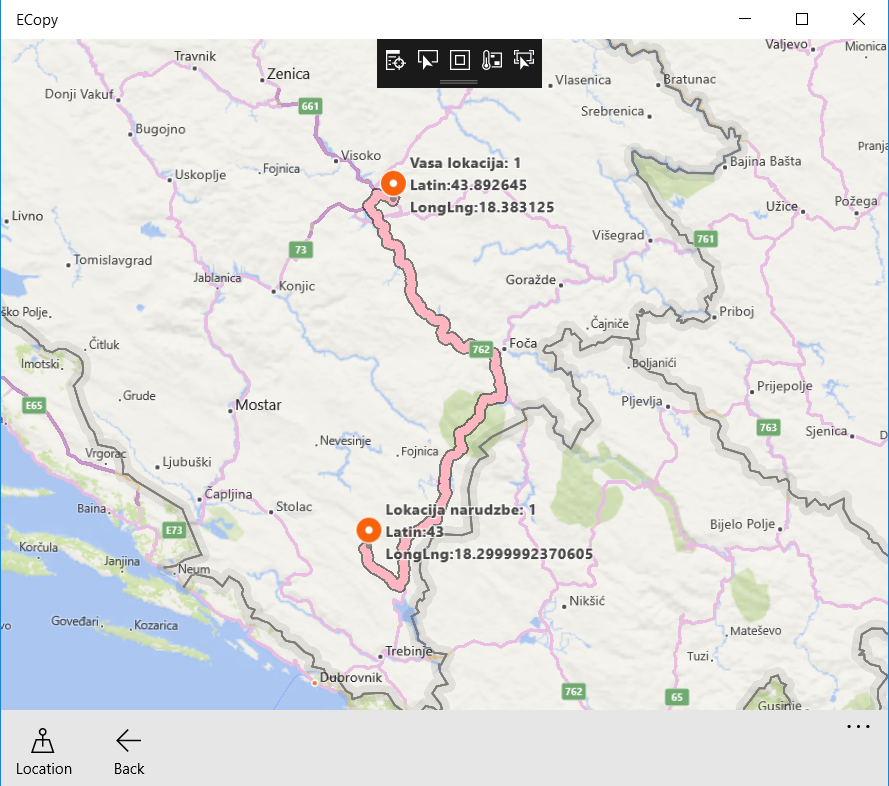
Takođe je moguće pristupiti tim podacima i mijenjati ih iz UWP aplikacije

Od specifičnih fukcionalnosti, implementirana je mapa koja prikazuje mjesto gdje se nalazi korisnik i njegova narudzba, te put između te dvije lokacije. Korisnik unosi id narudzbe i prikazuje mu se mapa, koordinate natudzbe mijenja radnik(dostavljac), ukoliko narudzba jos nije „krenula“ prikazuje se da se nalazi u kopirnici.



Slika 23. View pomocu kojeg radnik unosi koordinate određene narudžbe (on te koordinate treba da učita pomoću gps uređaja, ali to nije implementirano u aplikacij i- za potrebe testiranja unosimo koordinate ručno)

Radnik na svom profilu može unijeti narudžbu i kliknuti Provjeri kretanje button (Taj view vidimo na slici 22) nakon čega mu se otvara mapa kao na slici (na slici su ucitane koordinate koje je unio radnik)



# Izvještaj o aktivnostima

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Redni br. | Aktivnost | Ucesnici | Vrijeme | Napomena |
|  | Određivanje funkcionalnosti | Zajedno | 14.03.2018. |  |
|  | Izrada svih klasa i view-ova u UWP-u | Zajedno | 25.03.2018. – 04.04.2018. |  |
|  | Unos novog radnika u UWP | Berina | 04.04.3018. |  |
|  | Unos novg korisnika (fizickoLice) u UWP | Ajla | 07.04.2018. |  |
|  | Unos novog korisnika (firma) u UWP | Nudžejma | 09.04.2018. |  |
|  | Unos nove narudžbe (za izradu slika) u UWP | Ajla | 26.05.2018. |  |
|  | Log in kao administrator u UWP | Berina | 09.04.2018. |  |
|  | Log in kao korisnik u UWP | Nudžejma | 09.04.2018. |  |
|  | Unos koordinata narudžbe u UWP | Ajla | 02.06.2018. |  |
|  | Prikaz „puta“ narudžbe na karti (unosenjem id-a) i crtanje rute na karti (UWP) i očitavanje trenutne lokacije na karti (u UWP i ASP.NET) | Ajla | 03.06.2018. |  |
|  | Implementirano sve u ASP.NET-u | Zajedno | 26.05.2018. |  |
|  | Veza sa Azure bazom (iz ASP.NET-a, eCopyRestAPI i UWP-a) | Zajedno | 26.05.2018. |  |
|  | Implementacija ASP.NET WEB API servisa (kreiran projekat eCopyRestApi gdje je to implementirano) | Ajla | 05.06.2018. |  |
|  | Upis i čitanje svih podataka u istu bazu pomoću WEB API servisa | Ajla | 05.06.2018. |  |
|  | Deployment ASP.NET WEB API servisa na AZURE | Ajla | 06.06.2018. |  |
|  | Dokumentacija | Zajedno | 08.06.2018. |  |

Uradjene su sve zahtjevane aktivnosti.

# Tabela očekivanih i implementiranih funkcionalnosti

|  |  |  |
| --- | --- | --- |
| **Akter** | **Funkcionalnost** | **Implementirano** |
| Korisnik (fizicko lice) | Prijava | Da |
| Korisnih (fizicko lice) | Izmjena podataka | Da |
| Korisnih (fizicko lice) | Obavljanje narudžbe | Donekle (urađeno samo za jednu vrstu narudžbe, ostale se implementiraju slično) |
| Korisnik (sve vrste) | Pregled „puta“ narudžbe na mapi | Da |
| Korisnik(firma) | Prijava | Da |
| Korisnik(firma) | Izmjena podataka | Da |
| Korisnik(firma) | Obavljanje narudžbe | Ne |
| Gost | Obavljanje narudžbe | Donekle (ista funkcionalnost kao i za korisnika) |
| Gost | Registracija kao fizičko lice | Da |
| Gost | Registracija kao firma | Da |
| Radnik | Pregled i obrada natudžbe | Ne |
| Radnik | Unos koordinata za narudžbu | Da |
| Administrator | Unos novog radnika | Da |
| Administrator | Dodavanje novih pogodnosti | Ne |

# Refaktoring

* **Zamjena magičnih brojeva sa konstantama**

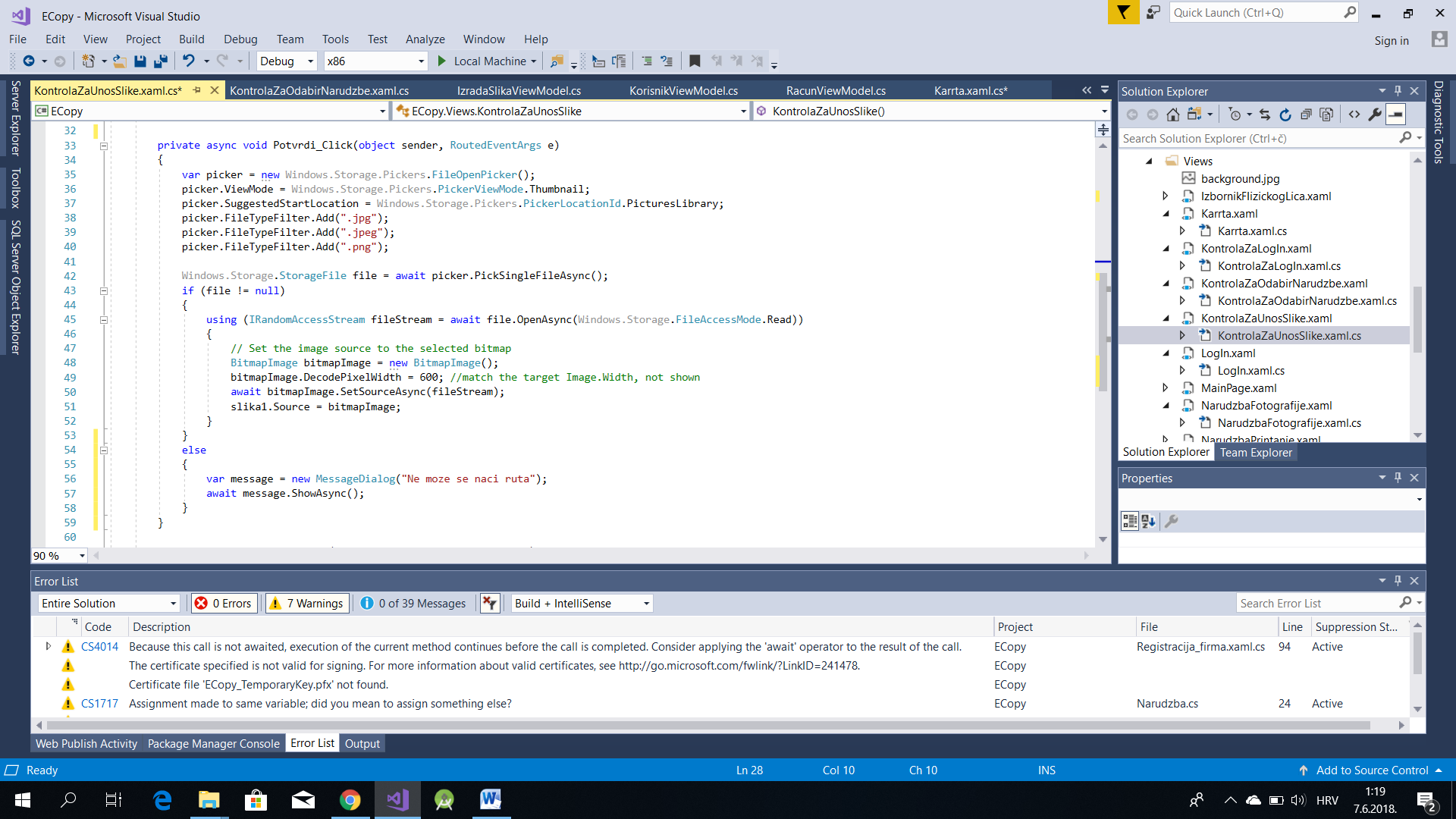
S obzirom da korisnik ima mogucnost da vidi na karti dokle je njegova narudzba stigla , potrebno je cuvati koordinate same kopirnice u varijablama, jer, ukoliko narudžba nije jos poslana, korisnik treba vidjeti da je ona u kopirnici. Da bi kod bio čitljiviji, najbolje rješenje jeste korištenje magičnih brojeva, tj. čuvanje tih vrijednosti u nekim varijablama sa jasnim nazivom (a i ukoliko dođe u nekom trenutku do promjene lokacije, lakše ih je pronaći i zamijeniti novim).



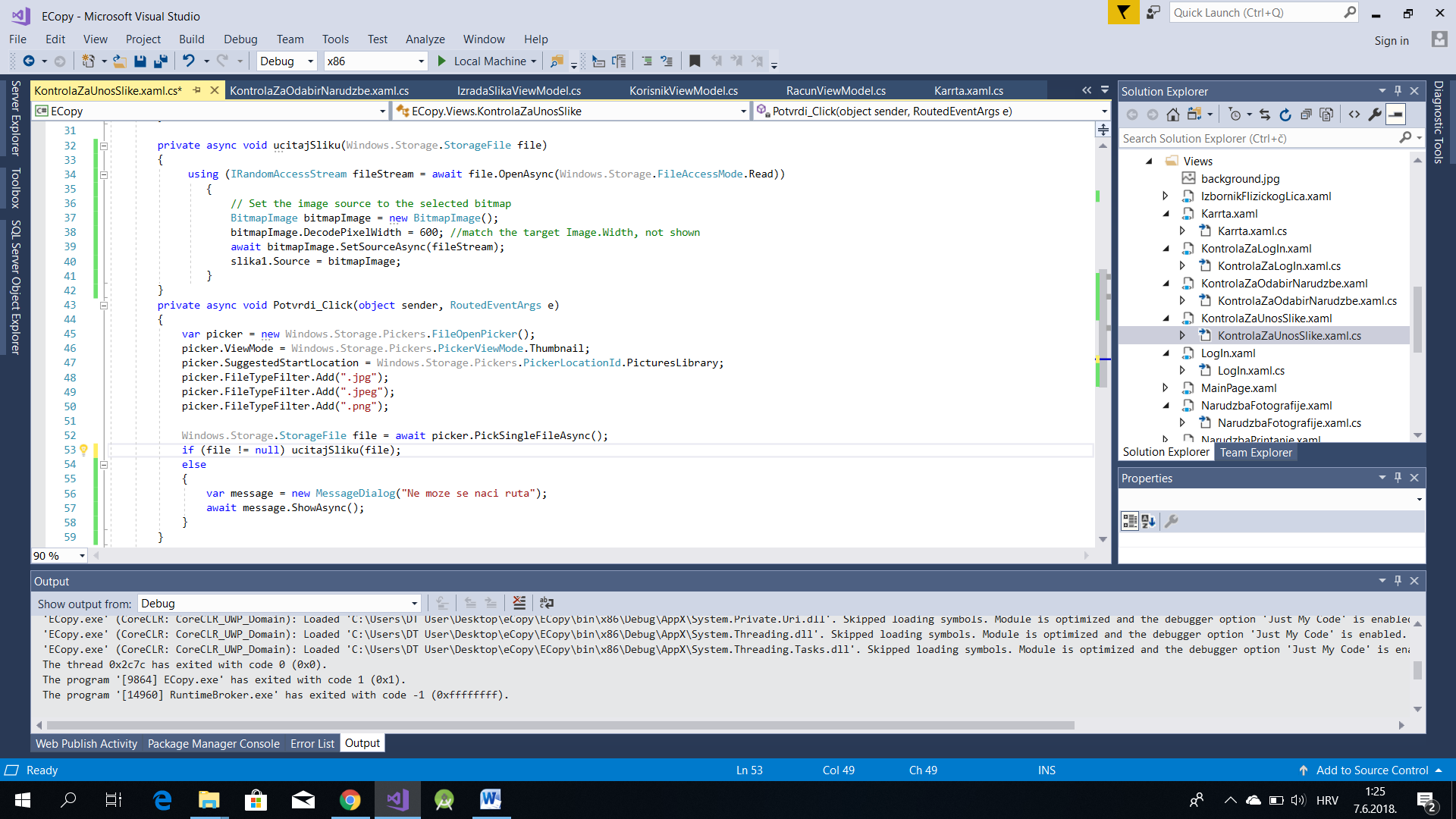
* **Za komplikovan uslovni (if-then-else) iskaz izdvojiti metodu za uslov dio**

Ovaj refaktoring je poželjan radi bolje čitljivosti koda i lakših izmjena u kodu.

Prije refaktoringa:



Poslije refaktoringa



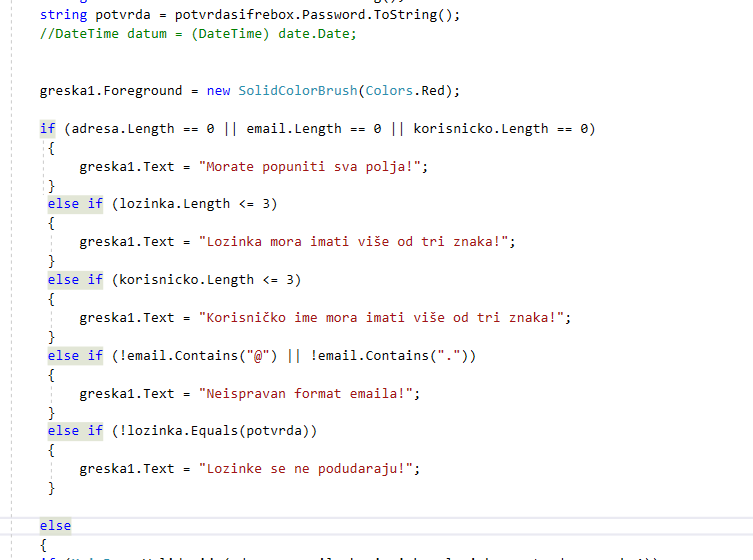
* **Polimorfizam i nasljeđivanje kod narudžbi**

Nasljeđivanje se koristi kako bi se izbjegli složeni if iskazi i switch case. Takođe, nasljeđivanjem se program otvara za nadogradnju (uveđenje novih funkcionalnosti i slično).

* **Ponavljanje dijelova koda u različitim view-ovima**

U svim view-ovima se koristi isti dio za validaciju gdje se koriste if i else if iskazi pa je taj dio odvojen u posebnu metodu koja se sad poziva više puta. Time je postignuta veća čitljivost koda te izbjegnuto ponavljanje istog dijela koda više puta. Takođe, sada je jednostavnije praviti promjene jer ih je potrebno vršiti samo na jednom mjestu.

Dio koda prije izvršenog refaktoringa:



Dio koda nakon refaktoringa:

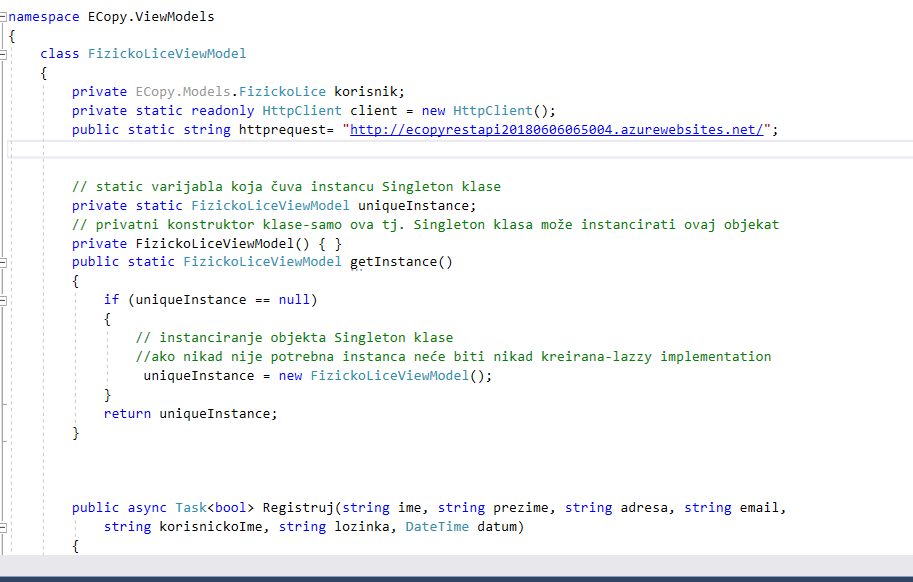


## Refaktoring pomoću design patterna

Singleton patern - KREACIJSKI PATERNI

Uloga Singleton paterna je da osigura da se klasa može instancirati samo jednom i da osigura globalni pristup kreiranoj instanci klase. Singleton pattern koristimo kod klase FizickoLiceViewModel jer je za svako korištenje te klase potrebna samo jedna instanca.

Klasa FizickoLiceViewModel je Singleton klasa i sadrži mehanizam za jedinstveno instanciranje same sebe. Unutar klase je private static varijabla (uniqueInstance) koja čuva jednu/jedinstvenu instancu klase, static metoda (getInstance) preko koje se pristupa Singleton klasi.



|  |  |  |
| --- | --- | --- |
|  | Refaktoring | Implementiro/la |
| 1. | Zamjena magičnih brojeva sa konstantama | Zajedno |
| 2. | Za komplikovan uslovni (if-then-else) iskaz izdvojiti metodu za uslov dio | Zajedno |
| 3. | Polimorfizam i nasljeđivanje kod narudžbi | Zajedno |
| 4. | Korištenje razumljivih i jasnih naziva klasa, atributa i metoda | Zajedno |
| 5. | Ponavljanje dijelova koda na različitim dijelovima zamjenjeno jednom metodom koja se poziva više puta | Ajla Panjeta |
| 6. | Singleton patern | Ajla Panjeta |
|  |  |  |