**UNIVERZITET U KRAGUJEVCU**

**FAKULTET TEHNICKIH NAUKA U CACKU**



**PROJEKTNI RAD**

**Savremene softverske arhitekture**

**Tema: Facebook klon zasnovan na mikroservisima**

|  |  |
| --- | --- |
| **Profesor:** | **Studenti:** |
| Prof. dr Vladimir Mladenovic | Vladimir Dukanac 262/2018 |

Aleksandar Petrovic 102/2018 **Asistenti:**

Mr Mladen Janjic Dr Faruk Selimovic

Cacak, 2022.

Sadrzaj

[1 Uvod 3](#_Toc106635423)

[1.1 Opis sistema 3](#_Toc106635424)

[2 Specifikacije nosilaca i interesa 3](#_Toc106635425)

[2.1 Specifikacije nosilaca 4](#_Toc106635426)

[2.2 Specifikacija interesa 5](#_Toc106635427)

[2.3 Asocijativna matrica interesa i nosilaca 5](#_Toc106635428)

[3 Specifikacija softverskih zahteva 6](#_Toc106635429)

[3.1 Funkcionalni zahtevi 6](#_Toc106635430)

[3.2 Nefunkcionalni zahtevi 7](#_Toc106635431)

[3.3 Procesni zahtevi 8](#_Toc106635432)

[4 UML dijagram modela 4+1 9](#_Toc106635433)

[4.1 Logicko glediste 9](#_Toc106635434)

[4.1.1 Dijagrami klasa 9](#_Toc106635435)

[4.2 Procesno glediste 10](#_Toc106635436)

[4.2.1 Dijagram aktivnosti 10](#_Toc106635437)

[4.3 Razvojno glediste 12](#_Toc106635438)

[4.3.1 Dijagram komponenti 12](#_Toc106635439)

[4.4 Slucajevi koriscenja 13](#_Toc106635440)

[4.4.1 Use case dijagram 13](#_Toc106635441)

[5 Prezentacioni sloj 14](#_Toc106635442)

[5.1 Opis aplikacije 14](#_Toc106635443)

[6 Zakljucak 14](#_Toc106635444)

# Uvod

U ovom seminarskom radu, opisan je nacin razvoja informacionog sistema za aplikaciju Facebook klon zasnovanu na mikroservisima.

Ova web aplikacija omogucava svojim korisnicima da vrlo brzo i lako koriscenje ove drustvene mreze kao i kacenje slika, lajkovanje istih, pregled slika, njihovo azuriranje, kao I azuriranje svog profila.

Cilj ovog seminarskog rada je zapravo efikasno i brzo umrezavanje korsinika, ali i njihova bezbednost na ovom sajtu, u vidu vrlo sigurnih cuvanja lozinki i podataka. Ova aplikacija bi olaksala svojim korisnicima umrezavanje, ali i pronalazanje novih korisnika sa slicinim interesima i stapanje novih prijateljstava na osnovih tih interesa.

## Opis sistema

Ova drustvena mreza kao referentu tacku ima mrezu Facebook, pa i samim tim ima slicne funckionalnosti kao i ona. Naravno funkcionalnosti nisu kopirane I radjene su na autentican nacin i realizovane svrsishodno potrebama aplikacije.

Korisnici ce moci da se registruju na ovu mrezu pomocu imena i sifre. Sifra je kriptovana I kao takva sacuvana u bazi podataka. Korisnici ce takodje moci da se loguju pomocu te sifre i imena, prilikom cega im je kao lista ponudjeni svi korisnici te drustvene mreze u sa profilnom slikom cilju, kao sto je receno, lakseg umrezavanja.

Klikom na odredjeni profil, korisnik ce imati uvid u druge korisnike, njihov opis, interesovanja, slike, kao i mogucnost razogovra sa njima u vidu malog chat sistema koji je realizovan.

Na svom profilu korisnik ce moci menjati svoje podatke u zavisnosti od potrebe i svojih prohteva, azurirati I ubacivati slike na svoj profil, kao I videti svoj inbox, outbox I neprocitane poruke. Takodje moze I videti korisnike koji mu se svidjaju, kao i korisnike koji ma se on svidja.

# Specifikacije nosilaca i interesa

Pod pojmom nosilac podrazumevamo onu osobu koja ima prava i interese sa sistemom ili nekim njegovim atributima koji zadovoljavaju potrebe tog nosioca.

Krajnji korisnici sistema su one zainteresovane strane i one osobe za koje je projekat i informacioni siste, kao I program, zapravo kreiran. Jedni su on najvaznijih aktera u sistemu. Imajuci to u vidu mora se napraviti informacioni sistem koji jasno identifikuje nosioce sistema i njihove tipove, kao I njihove interese I veze medju njima

## Specifikacije nosilaca

Pod pojmom nosioci moramo viditi kakvi su njihovi interesi, pa samim tim razlikujemo:

* Korisnike
* Vlasnike
* Developere
* I odrzavaoce tog sistema

Nosioci su one osobe:

* Imaju odredjenu vrstu interesa u sistemu
* Odrzavaju sistem
* Operisu nad tim sistemom
* Imaju odredjenu korist od sistema
* Regulisu aktere sistema, ili
* Na bilo koji nacin uticu na funkcionisanje tog sistema

U tabeli ispod prikazane su specifikacije nosilaca:

|  |  |  |
| --- | --- | --- |
| Naziv | Opis | Interesi |
| Vlasnik | Osoba koji kontrolise i drzi aplikaciju, sama rec kaze, vlasnik te aplikacije | Efikasnost aplikacije I njen kvalitet |
| Developer | Projektant sistema I njegov razvojnik koji obraca paznju I stavlja akcenat na specifikacije sistema | Zarada |
| Korisnik | U ovom slucaju to je korisnik aplikacije, tj osoba osoba koja ima profil na drustvenoj mrezi i uziva u njenim pogodnostima | Efikansnost i kvalitet, kao I kod vlasnika sistema, imaju slicne interese |
| Odrzavaoci | Ljudi koji redovno odrzavaju sistem i obracaju paznju na njegove funkcionalnost, efikasnost i stanje | Odrzavanje i zarada |

Tabela nosilaca

## Specifikacija interesa

Pod pojmom interes podrazumevamo zainteresovanost nosilaca za dati informacioni sistem, a u to spadaju kvalitet, trosak, pristupacnost, distribucija, sigurnost i enkripcija podataka, integracija sa drugim sistemima…

U tabeli ispod data je specifikacije interesa sistema u vidu njihovih naziva, opisa kao i ko od nosilaca ima odredjeni interes tj. veza izmedju interesa i nosioca:

|  |  |  |
| --- | --- | --- |
| Naziv | Opis | Interesi |
| Funkcionalnost | Podrazumeva da li je website lak i efikasan za koriscenje | Korisnik, vlasnik |
| Odrzavanje | Na osnovu zahteva korisnika, a u cilju odrzavanja funkcionalnosti i efikasnosti web aplikacije, potrebno je odrzavati aplikaciju sa vremena na vreme | Odrzavalac |
| Zarada | Projektant ovog sistema(developer) prevashodno ima za cilj zaradu, ali i vlasnik aplikacije je ima takodje | Developer, ali i vlasnik |
| Efikasnost | Obezbedjivanje svih zahteva u datom sistemu i njihovo efikasno izvrsavanje | Vlasnik |

Specifikacija interesa sistema

## Asocijativna matrica interesa i nosilaca

U tabeli ispod prikazana je asocijativna matrica interesa i nosilaca sistema:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nosioci Interesi | VLASNIK | DEVELOPER | KORISNIK | ODRZAVAOCI SISTEMA |
| FUNKCIONALNOST |  | R | P |  |
| ODRZAVANJE |  | R | P | O |
| ZARADA | K | K |  | K |
| EFIKASNOST | R |  | P |  |

Legenda:

K – predstavlja odredjenu vrstu koristi u sistemu

O – predstavlja obavezu nosioca

R – nosilac regulise i propisuje odredjenu stvar

P – nosilac je pogodjen odredjenim interesom

# Specifikacija softverskih zahteva

Uslovi i tacno odredjene usluge koje informacioni sistem mora da ispuni predstavljaju softverse zahteve sistema. Ni se predstavljaju i opisuju pomocu UML dijagrama, modela use-case dijagrama. Tu je data veza izmedju use-case dijagrama i aktera.

Use case dijagram ili u prevodu dijagram slucajeva koriscenja opisuje skup mogucih scenarija koji se desavaju u datom sistemu, na strani aktera. Specifikacija zahteva predstavlja opis potreba koje odredjeni informacioni sistem treba da ispuni. Neophodno je dokumentovati i priloziti potrebe radi lakse komunikacije klijenta sa razvojim timom.

Softverski zahtevi se mogu podeliti u 3 grupe:

* Funkcionalni
* Nefukcionalni
* Procesni

## Funkcionalni zahtevi

Prepoznaju se i identifikuju pomocu istrazivanja tehnicke dokumentacije realnom sistema u realnom vremenu, u vidu razgovora i interakcije sa korisnicima sistema i naruciocima. Cilj je opis funkcionalnosti sistema.

|  |  |  |  |
| --- | --- | --- | --- |
| RB | Zahtev | Segment | Prioritet |
| 1 | Pregled svih registrovanih korisnika u vidu liste | Pregled | О |
| 2 | Azuriranje profila korisnika u vidu podataka, ubacivanja slika i azuriranje postojecih slika | Update | О |
| 3 | Umrezavanje korisnika u vidu cet sistema | ? | V |

Tabela funkcionalnih zahteva

Legenda:

O - obavezni

M - moguci

V- vazni

B - buduci

## Nefunkcionalni zahtevi

Predstavljaju neka ogranicenja sistema(performanse, dostpunost, bezbednost, pouzdanost) I neke sistemske karakterisitka softvera, kao I odredjene standarde(integracija, pogodnost za odrzavanje, propisi, prenosivost).

Svi nefunkcionalni zahtevi su prikazani u tabeli sa jasnim pogledom i opsiom. Raspored nefunkcionalnih zahteva se vrsi po prioritetu na osnovu vaznosti i svakom zahtevu je pripisan odredjeni redni broj(RB):

|  |  |  |  |
| --- | --- | --- | --- |
| RB | Zahtev | Segment | Prioritet |
| 1 | Odredjenim fukncionalnostima sistema moze pristupiti samo registrovani korisnik nakon logovanja | Bezbednost i integritet | О |
| 2 | Rednovno azuriranje sistema(update) nakon registrovanja novih korisnika | Odrzavanje | О |
| 3 | Vreme cekanje na izvlacenje podataka iz baze podataka mora biti svedeno na minimum | Performanse | О |
| 4 | Cross-platform aplikacija(moze se koristiti na vise platformi) | Performanse | V |
| 5 | Cuvanje sifara korisnika u vidu enkriptovanih i hesovanih podataka u tabeli | Bezbednost | О |

Nefunkcionalni zahtevi

Legenda:

O - obavezni

M - moguci

V- vazni

B - buduci

## Procesni zahtevi

Pod procesnim zahtevima podrazumevamo sam razvoj softvera. Postavlja ih proizvodjac softvera ili u nekim posebnim slucajevima, ako ima potrebe za tim i ako se iskaze zelj – kupac. U sustini to je neko ogranicenje za razvoj samog softvera kao sto je da (u slucaju ovog informacionog sistema) mora biti razvijen u ASP.NET Core i Angular tehnologijama.

Klasifikuju se prema prioritetu I svakom je dodeljen redni broj, kao i kod funkcionalnih i nefunkcionalnih zahteva. Prikazani su na tabeli ispod:

|  |  |  |  |
| --- | --- | --- | --- |
| RB | Zahtev | Karakteristika | Prioritet |
| 1 | Bice razvijen pomocu ASP.NET Core i Angular okruzenjima | Razvoj | О |
| 2 | Testiranje baze podataka, koja je u ovom slucaju SQLite se vrsi podacima u realnom vremenu | Testiranje | V |
| 3 | Sistem ima sigurnu autentifikaciju i autorizaciju na odredjenim stranama, kao i prilikom logovanja | Bezbednost | O |
| 4 | Podaci u bazi podataka moraju biti enkriptovani odredjenim algoritmom | Sigurnost | O |

Procesni zahtevi

Legenda:

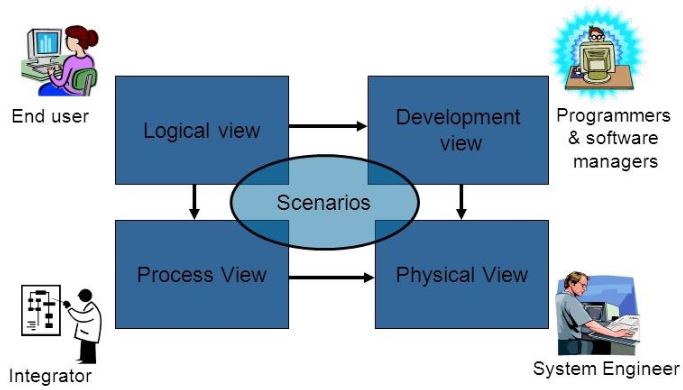
O - obavezni

M - moguci

V- vazni

B - buduci

# UML dijagram modela 4+1



*UML model 4+1*

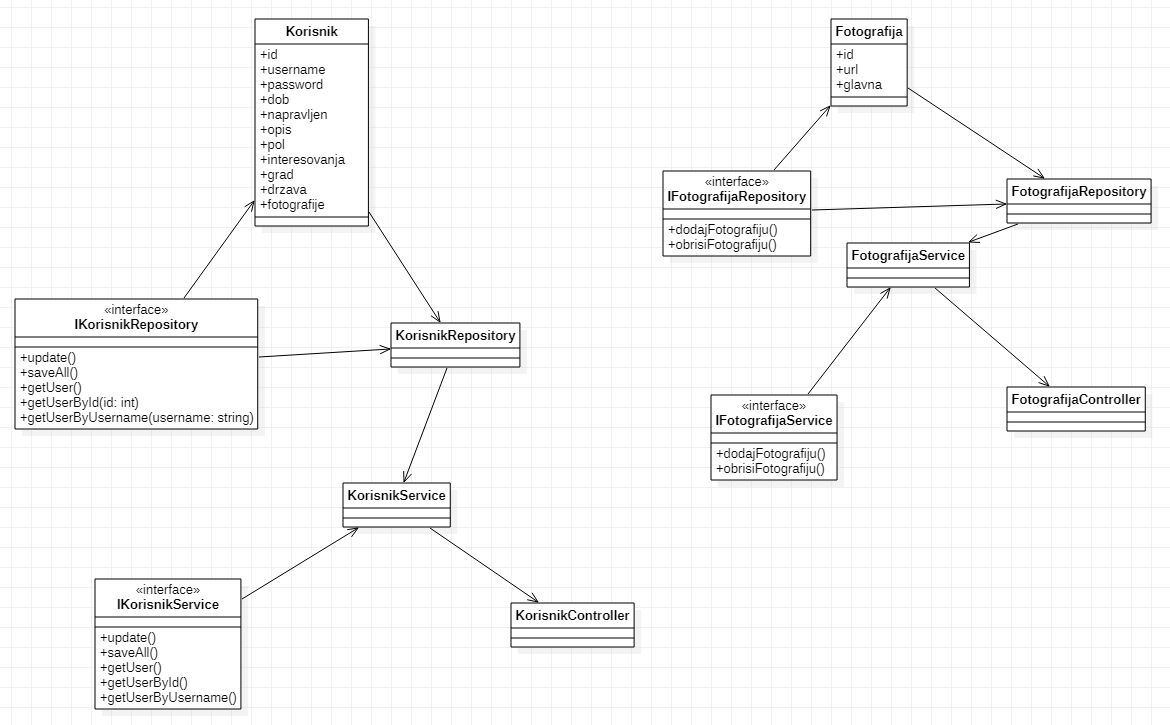
## Logicko glediste

Ovaj stav vidjenja tj. glediste, definise nacin na koji se posmatra odredjeni predmetni sistem. Ono mora podrzavati funkcionalne softverske zahteve i precizno klasifikovati servise koji sistem treba da obezvedi svojim korisnicima.

### Dijagrami klasa

Dijagrami klasa su modeli koje prikazuju klase u odredjenom sistemu kao i veze medju njima. One imaju svoje nazive, atribute i metode. Svaki atribut klase mora imati svoj tip koji moze biti int, bool, double, float…

Relacije izmedju klasa jesu zapravo veze izmedju objekata klasa i postoji nekoliko tipova: asocijacija, generalizacija, realizacija I agregacija. Predstavljaju se linijama karakteristicnim I unikatnim za svaku.



*Dijagram klasa*

## Procesno glediste

Podrzava nefunkcionalne zahteve koji su u nadleznosi developera. Njime se specifikuje procesna arhitektura sistema.

Trebalo bi ispuniti sledece zahteve:

* Odredjenim fukncionalnostima sistema moze pristupiti samo registrovani korisnik nakon logovanja
* Rednovno azuriranje sistema(update) nakon registrovanja novih korisnika
* Vreme cekanje na izvlacenje podataka iz baze podataka mora biti svedeno na minimum
* Cross-platform aplikacija(moze se koristiti na vise platformi)

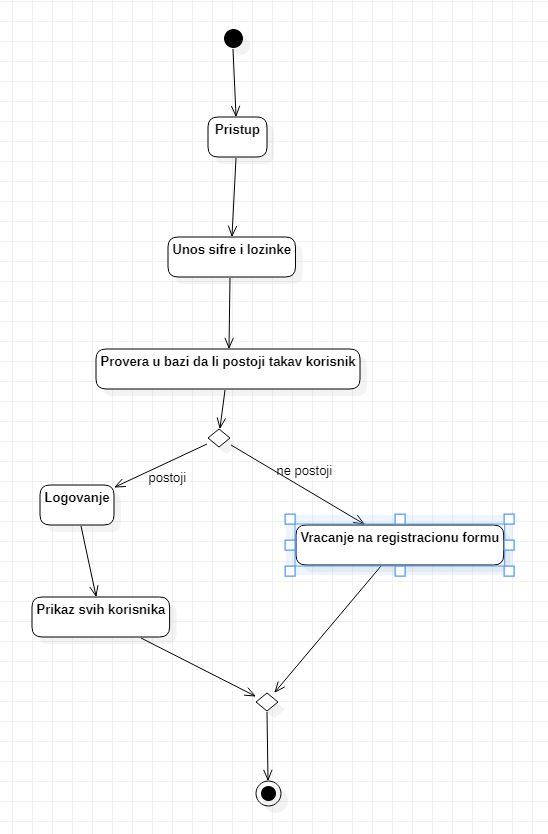
### Dijagram aktivnosti

Oni se pravi u svrhu modeliranja dinamickog aspekta sistema tj. opisuju ponasanje sistema prikazujuci sekvencijalne korake u procesu obrade. Tretiraju se kao specijalni slucajevi dijagrama stanja.

Cvorovi ustvari prikazuju akcije, a to su metode klasa ili aktivnosti u nasem sistemu. Akcija tj cvoor je osnovna jedinica specifikacije I ponasanja i osnovni izvrsni element aktivnosti. Dijagrami aktivnosti sadrze aktivnosti, tranzicije medju aktivnostima, raskrsca(tacke) odluka…

Kao UML dijagrami predstavljaju se kao zaobljeni pravougaonici, prelazi kao strelice, tacke odluka kao rombovi, a sinhroni barovi kao zadebljani vertikalne ili horizontalne linije.

#### Dijagram aktivnosti logovanja korisnika



*Dijagram aktivnosti logovanja korisnika*

Dijagram aktivnosti iznad opisuje pristup korisnika sistemu i prikaz forme za unos sifre i lozinke. Nakon unosa te dve stavke, prosledjuju se na bazu podataka i servisnu logiku(biznis logiku) i vrsi se provera da li postoji takav korisnik u bazi podataka. Dolazimo do racvanja, cvorista gde ako postoji korisnik, salje se na pocetnu stranu, odobravama mu se ulaz i daje prikaz svih korisnika u vidu liste. Ako pak ne postoji takav korisnik u bazi, softverska logika ga vraca na registracionu formu da bi se prvo registrovao pa potom logovao.

## Razvojno glediste

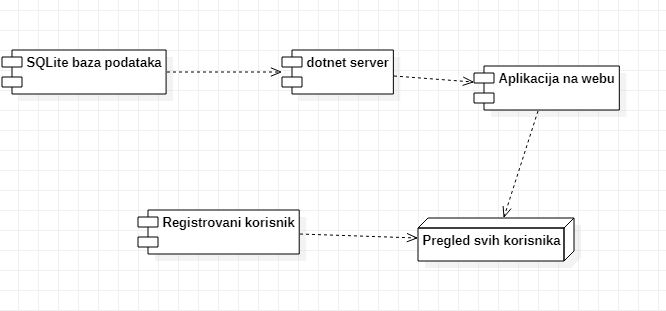
Sa druge strane ovaj stav gledista na sistem podrazumeva njegovo rasclanjavanje na vise manjih podsistema i analizira organizaciju modula I komponenata, kao i viseslojnu arhitekturu informacionog sistema. Pomocu njega se mozemo reci upravlja softverskim projektom.

Procesni zahtevi koji moraju biti ispunjeni su:

* Bice razvijen pomocu ASP.NET Core i Angular okruzenjima
* Sistem ima sigurnu autentifikaciju i autorizaciju na odredjenim stranama, kao i prilikom logovanja
* Podaci u bazi podataka moraju biti enkriptovani odredjenim algoritmom

### Dijagram komponenti

Dijagram komponenti je specijalan sucaj dijagrama klasa za komponente unutar softversog sistema. Jedna komponenta je zamenljiv deo sistema koji realizuje odredjeni skup interfejsa I sluzi da prikaze fizicku organizaciju i zavisnosti medju komponentama.



*Dijagram komponenti*

## Slucajevi koriscenja

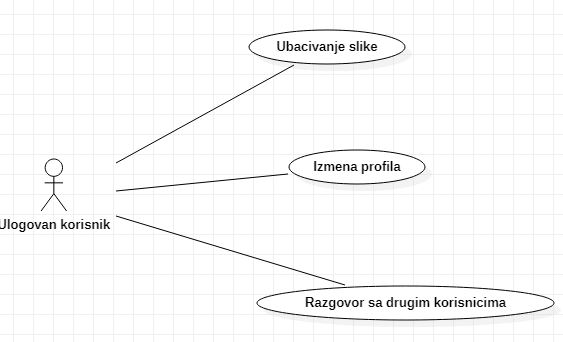
|  |
| --- |
| **UseCase: Logovanje korisnika** |
| **Opis:** Registrovani korisnik vrsi unos imena i lozinke  **Akter:** Registrovani korisnik  **Preduslov:** Korisnik mora biti registrovan u aplikaciji  **Posledice:** Pristup svim funkcionalnostima sistema |
| **UseCase: Ubacivanje nove slike na nalogu** |
| **Opis:** Ulogovan korsinik ubacuje(vrsi insert) nove slike na svom profilu  **Akter:** Ulogovan korisnik  **Preduslov:** Logovanje na aplikaciju  **Posledice:** Postavljena nova fotografija na galeriju, profil korisnika |

### Use case dijagram

Prikazuje slucajeve korsiscenja i aktere. Daje uvid u ponasanje sistema, klasa, pa i interfejsa. Sluzi da korisnik ima uvid sta sistem radi, kako se ponasa i kako sistem funkcionise.

Use case dijagram se sastoji od:

* Aktera
* Use case dijagrama
* Veza
* Veza asocijacije
* Extend veza
* Include veza
* Veza generalizacije



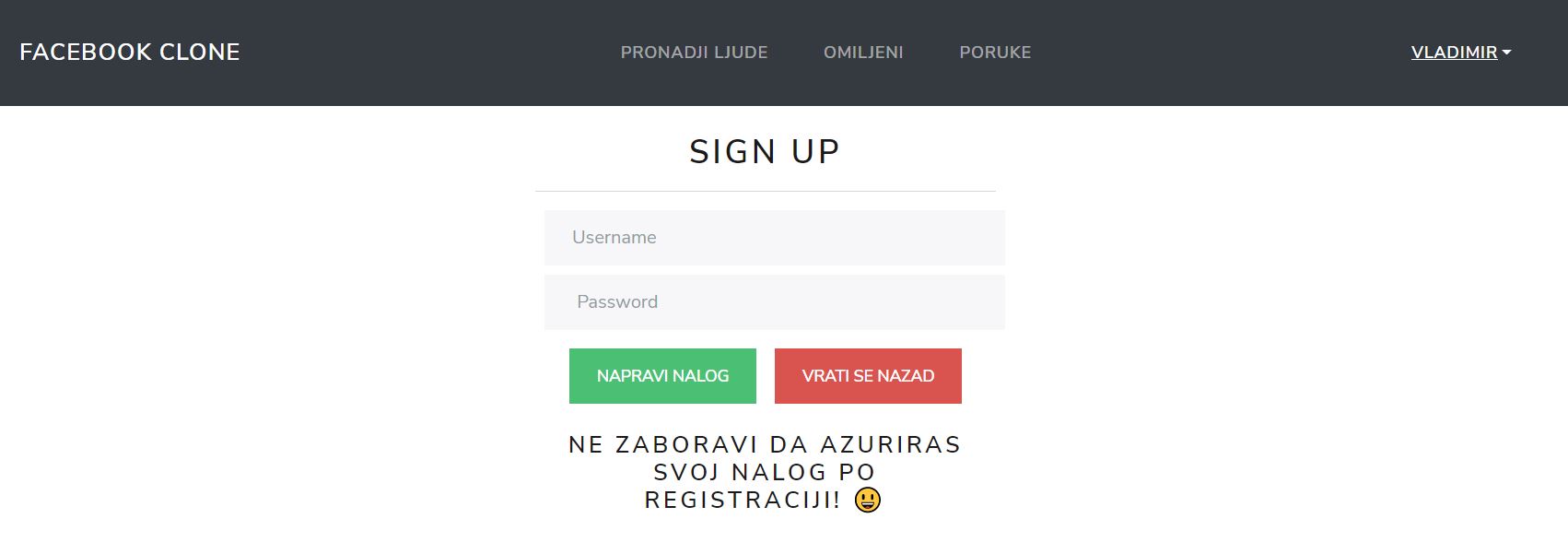
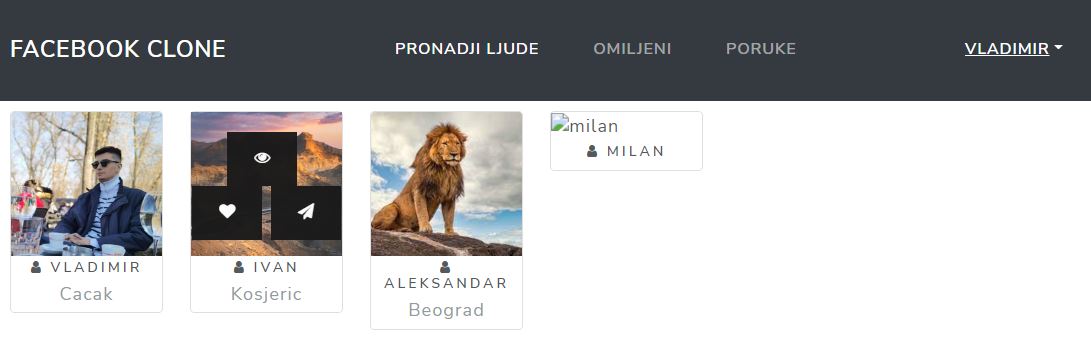
*Dijagram slucajeva koriscenja*

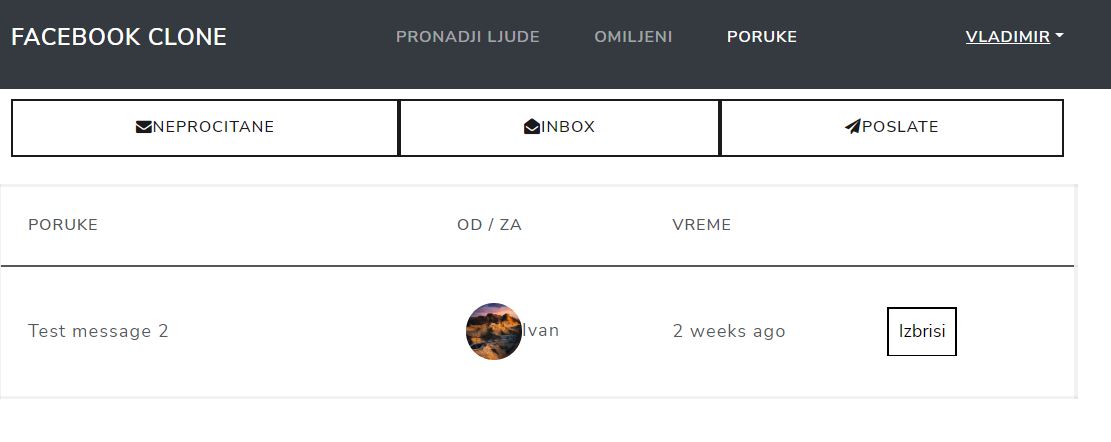
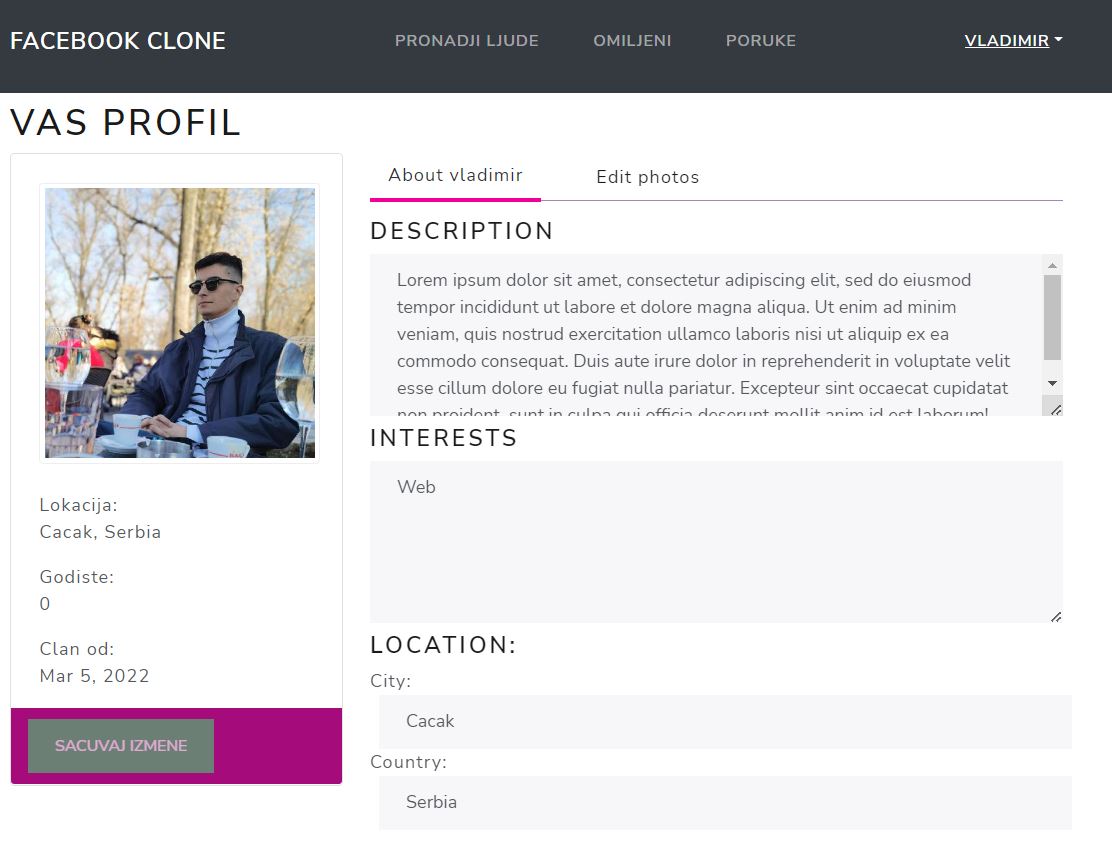
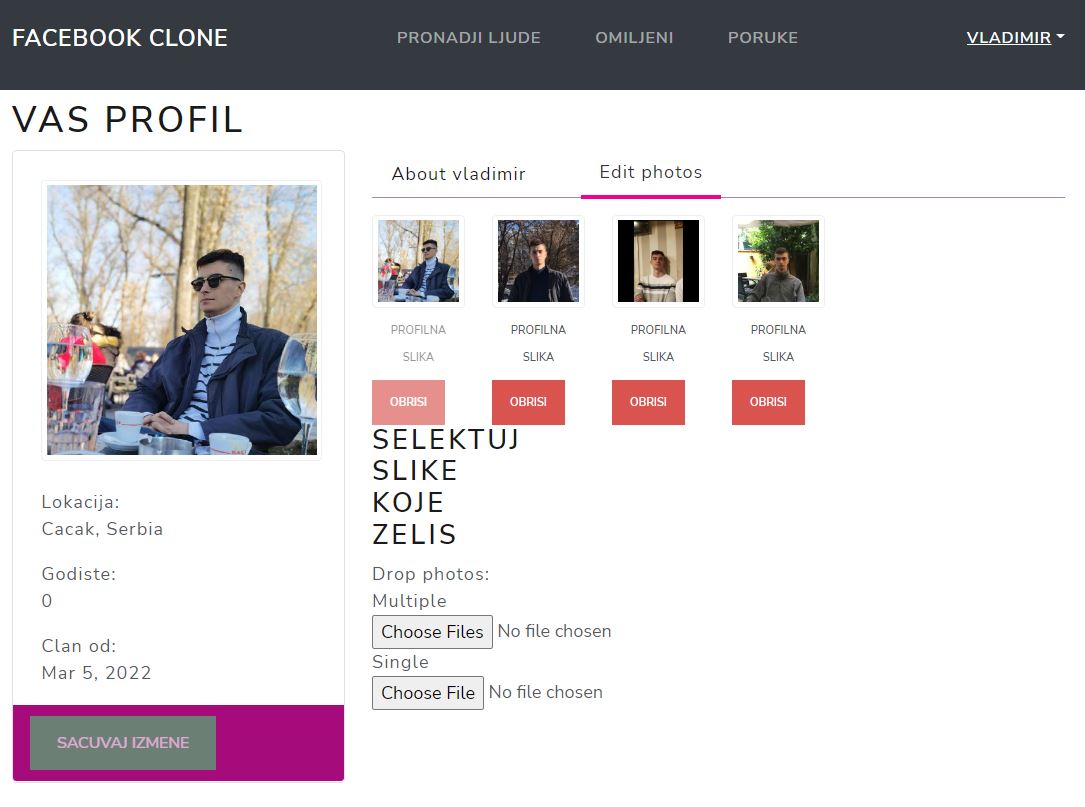
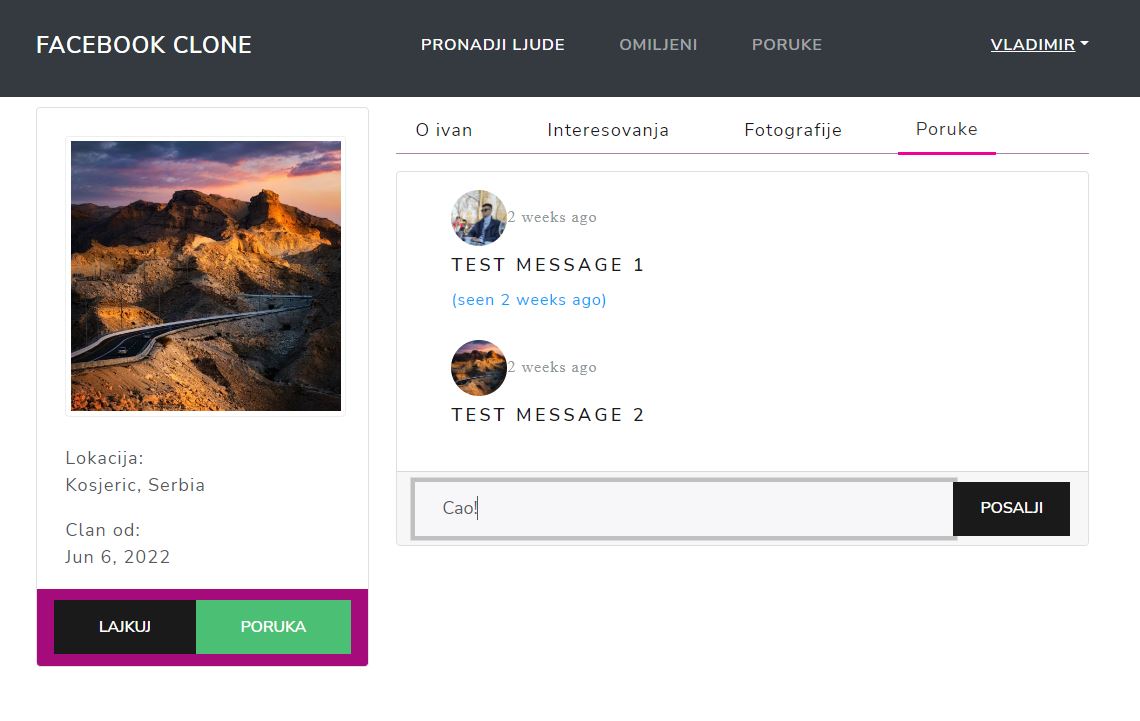
# Prezentacioni sloj

Ono sto preostaje u ovakoj aplikaciji jeste zapravo i njena realizacija, tj. realizacija informacionog sistema na osnovu Facebook-a radi lakseg umrezavanja ljudi na internetu.

Aplikacija je izradjena u ASP.NET Core tehologije na serverskoj strani(API), a na strani klijenta korisceno je Angular okruzenje.

## Opis aplikacije



# Zakljucak

Ova aplikacija I informacioni sistem moze pomoci korisnicima da pronadju neke nove prijatelje, upoznaju se sa njima, vide njihova interesovanja, umreze se, ali I da sa postojecim prijatejima I vezama ostvare i odrze kontakt i neki vid druzenja.