И/N

Izgubljeno/Nađeno

Arhitekturni projekat

Verzija 1.0

Pregled izmena

| **Datum** | **Verzija** | **Opis** | **Autor** |
| --- | --- | --- | --- |
| 21.04.2023. | 1.0 | Inicijalna verzija | Marko |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Sadržaj

[**1. Cilj dokumenta 5**](#_heading=h.gjdgxs)

[**2. Opseg dokumenta 5**](#_heading=h.30j0zll)

[**3. Reference 5**](#_heading=h.1fob9te)

[**4. Predstavljanje arhitekture 5**](#_heading=h.2et92p0)

[**5. Ciljevi i ograničenja arhitekture 5**](#_heading=h.tyjcwt)

[**6. Pogled na slučajeve korišćenja 5**](#_heading=h.3dy6vkm)

[6.1 Dijagrami slučajeva korišćenja 6](#_heading=h.2bn6wsx)

[6.2 Kratak opis slučajeva korišćenja 7](#_heading=h.qsh70q)

[6.2.1 Korišćenje funkcionalnosti sistema 7](#_heading=h.3as4poj)

[6.2.2 Menjanje statusa oglasa 7](#_heading=h.1pxezwc)

[6.2.3 Brisanje oglasa 7](#_heading=h.49x2ik5)

[6.2.4 Odobravanje oglasa 7](#_heading=h.2p2csry)

[6.2.5 Registracija 8](#_heading=h.147n2zr)

[6.2.6 Kontaktiranje pronalazača 8](#_heading=h.3o7alnk)

[6.2.7 Pregled oglasa po lokaciji 8](#_heading=h.23ckvvd)

[6.2.8 Pregled oglasa po starosti 8](#_heading=h.ihv636)

[6.2.9 Pregled po kategoriji 8](#_heading=h.32hioqz)

[6.2.10 Primanje obaveštenja 8](#_heading=h.1hmsyys)

[6.2.11 Unos opisa oglasa 8](#_heading=h.41mghml)

[6.2.12 Dodavanje fotografija oglasa 8](#_heading=h.2grqrue)

[6.2.13 Obeležavanje lokacije oglasa 8](#_heading=h.bi3vpq1gi0uq)

[**7. Pogled na logičku arhitekturu sistema 8**](#_heading=h.28h4qwu)

[7.1 Pregled arhitekture – organizacija paketa i podsistema u slojeve 9](#_heading=h.nmf14n)

[7.1.1 Korisnički interfejs 9](#_heading=h.37m2jsg)

[7.1.2 Aplikaciona logika 9](#_heading=h.1mrcu09)

[7.1.3 Pristup podacima 9](#_heading=h.46r0co2)

[7.1.4 React.js 9](#_heading=h.2lwamvv)

[7.1.5 .NET Core 9](#_heading=h.111kx3o)

[7.1.6 MySQL 9](#_heading=h.3l18frh)

[**8. Pogled na procese 10**](#_heading=h.206ipza)

[8.1 Procesi 10](#_heading=h.4k668n3)

[8.1.1 Web čitač 10](#_heading=h.2zbgiuw)

[8.1.2 Web server 10](#_heading=h.1egqt2p)

[8.1.3 React-DOM 10](#_heading=h.3ygebqi)

[8.1.4 MySQL 10](#_heading=h.2dlolyb)

[**9. Pogled na raspoređivanje sistema 11**](#_heading=h.sqyw64)

[9.1 Klijent 11](#_heading=h.3cqmetx)

[9.2 Web server 11](#_heading=h.1rvwp1q)

[9.3 DBMS server 11](#_heading=h.4bvk7pj)

[**10. Pogled na implementaciju sistema 11**](#_heading=h.2r0uhxc)

[10.1 Model domena 11](#_heading=h.1664s55)

[10.2 Šema baze podataka 12](#_heading=h.3q5sasy)

[10.3 Komponente sistema 12](#_heading=h.25b2l0r)

[10.3.1 Komponente korisničkog interfejsa 12](#_heading=h.kgcv8k)

[10.3.2 Komponente aplikacione logike 13](#_heading=h.34g0dwd)

[10.3.3 Komponente za pristup podacima 14](#_heading=h.1jlao46)

[**11. Performanse 14**](#_heading=h.43ky6rz)

[**12. Kvalitet 14**](#_heading=h.2iq8gzs)

Arhitekturni projekat

# Cilj dokumenta

Cilj ovog dokumenta je detaljni opis arhitekture И/N aplikacije.

# Opseg dokumenta

Dokument se odnosi na И/N aplikaciju koja će biti razvijena od strane JJM-a. И/N predstavlja skraćenicu za izgubljeno/nađeno. Namena sistema je efikasno pretraživanje, kreiranje i održavanje oglasa za izgubljene stvari.

# Reference

Spisak korišćene literature:

1. И/N – Predlog projekta, SWE-И/N-01, V1.0, 2023, JJM.
2. И/N – Planirani raspored aktivnosti na projektu, V1.0, 2023, JJM.
3. И/N – Plan realizacije projekta, V1.0, 2023, JJM.
4. И/N – Vizija sistema, V1.0, 2023, JJM.
5. И/N – Specifikacija zahteva, V1.0, 2023, JJM.

# Predstavljanje arhitekture

Arhitektura sistema u dokumentu je prikazana kao serija pogleda na sistem: pogled na slučajeve korišćenja, pogled na logičku arhitekturu sistema, pogled na procese, pogled na razmeštaj komponenti sistema i pogled na implementaciju. Ovi pogledi su predstavljeni odgovarajućim UML dijagramima.

# Ciljevi i ograničenja arhitekture

Ključni zahtevi i sistemska ograničenja koja imaju značajan uticaj na izbor arhitekture i projektovanje sistema su:

1. И/N aplikacija će biti implementirana kao Web aplikacija zasnovana na React.js-u i .NET-u , zajedno sa MySQL bazom podataka [4].
2. Klijentski deo И/N aplikacije će biti optimizovan za sledeće Web čitače: Opera 8.0 i noviji, kao i Firefox (Mozilla) [4].
3. Svi zahtevi u pogledu performansi dati u [5] moraju biti uzeti u obzir pri izboru arhitekture i razvoju sistema.
4. Klasifikaciju publikacija treba standardizovati sa postojećom tipologijom propisanom od strane Ministarstva za nauku.

# Pogled na slučajeve korišćenja

U ovom odeljku je dat pogled na slučajeve korišćenja definisane u specifikaciji zahteva [5].

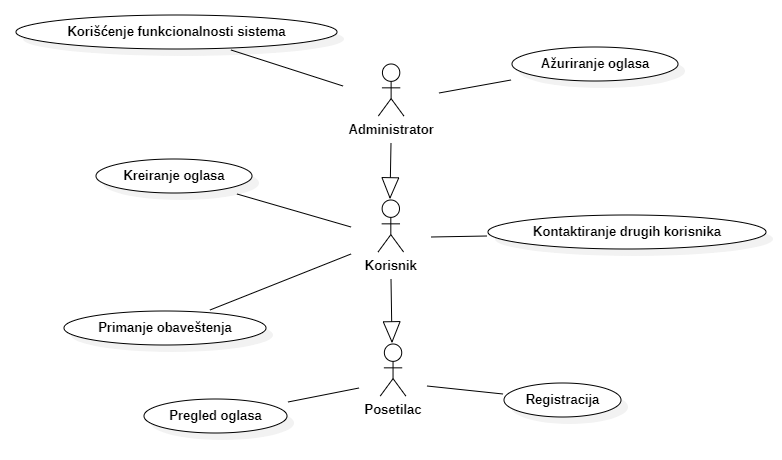
Slučajevi korišćenja И/N aplikacije su:

* Korišćenje funkcionalnosti sistema
* Ažuriranje oglasa
  + Menjanje statusa oglasa
  + Brisanje oglasa
  + Odobravanje oglasa
* Registracija
* Kontaktiranje pronalazača
* Pregled oglasa
  + Pregled oglasa po lokaciji
  + Pregled oglasa po starosti
  + Pregled po kategoriji
* Primanje obaveštenja
* Unos oglasa
  + Unos opisa oglasa
  + Dodavanje fotografija oglasa
  + Obeležavanje lokacije oglasa

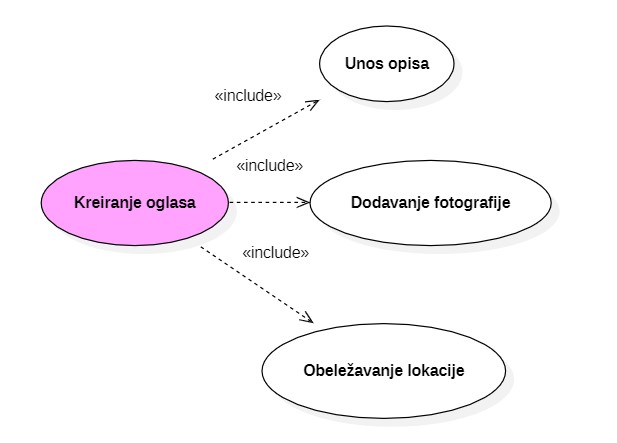
Ove slučajevi korišćenja mogu da iniciraju administrator aplikacije, pronalazač izgubljenje imovine (skraćeno pronalazači) i potencijalni vlasnik izgubljenje imovine.

## Dijagrami slučajeva korišćenja

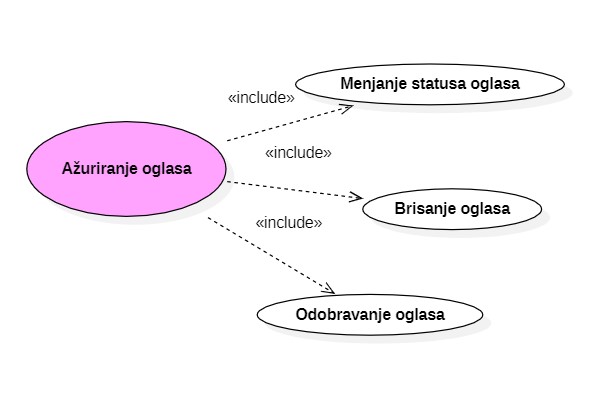
Osnovni UML dijagram koji prikazuje korisnike i slučajeve korišćenja И/N aplikacije prikazan je na sledećoj slici:



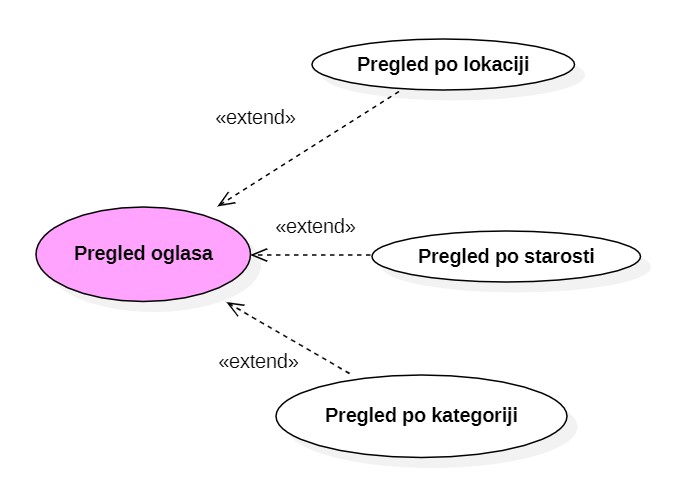
Slučajevi korišćenja *kreiranje oglasa, ažuriranje oglasa* i *pregled oglasa* obuhvataju složenije radnje koje se mogu dalje razložiti na pojedinačne slučajeve korišćenja.

Detaljni UML dijagram za slučaj korišćenja *kreiranje oglasa* je prikazan na sledećoj slici:

Detaljni UML dijagram za slučaj korišćenja *ažuriranje oglasa* je prikazan na sledećoj slici:



Detaljni UML dijagram za slučaj korišćenja *pregled oglasa* je prikazan na sledećoj slici:



## Kratak opis slučajeva korišćenja

### Korišćenje funkcionalnosti sistema

Kratak opis: Pregled oglasa i pregled korisnika.

Akteri koji iniciraju slučaj korišćenja: Administrator.

### Menjanje statusa oglasa

Kratak opis: Izmena statusa razrešenosti oglasa (razrešen/nerazrešen).

Akteri koji iniciraju slučaj korišćenja: Administrator.

### Brisanje oglasa

Kratak opis: Uklanjanje oglasa iz evidencije.

Akteri koji iniciraju slučaj korišćenja: Administrator.

### Odobravanje oglasa

Kratak opis: Dozvola za puštanje oglasa u evidenciju.

Akteri koji iniciraju slučaj korišćenja: Administrator.

### Registracija

Kratak opis: Unošenje i verifikacija podataka neophodnih za registraciju aktera u aplikaciju.

Akteri koji iniciraju slučaj korišćenja: Posetilac, Korisnik, Administrator.

### Kontaktiranje pronalazača

Kratak opis: Stupanje u kontakt sa potencijalnim pronalazačem.

Akteri koji iniciraju slučaj korišćenja: Posetilac, Korisnik, Administrator.

### Pregled oglasa po lokaciji

Kratak opis: Prikaz oglasa na izabranoj lokaciji.

Akteri koji iniciraju slučaj korišćenja: Posetilac, Korisnik, Administrator.

### Pregled oglasa po starosti

Kratak opis: Grupisanje i sortiranje evidencije na osnovu starosti oglasa

Akteri koji iniciraju slučaj korišćenja: Posetilac, Korisnik, Administrator.

### Pregled po kategoriji

Kratak opis: Prikaz oglasa na osnovu izabrane kategorije.

Akteri koji iniciraju slučaj korišćenja: Posetilac, Korisnik, Administrator.

### Primanje obaveštenja

Kratak opis: Korisnik koji je prijavljen u aplikaciju, prima obaveštenje preko korisničkog interfejsa.

Akteri koji iniciraju slučaj korišćenja: Korisnik, Administrator.

### Unos opisa oglasa

Kratak opis: Unošenje tekstualnog opisa oglasa.

Akteri koji iniciraju slučaj korišćenja: Korisnik, Administrator.

### Dodavanje fotografija oglasa

Kratak opis: Postavljanje deskriptivne slike uz odgovarajući oglas.

Akteri koji iniciraju slučaj korišćenja: Korisnik, Administrator.

### Obeležavanje lokacije oglasa

Kratak opis: Obeležavanje lokacije oglasa.

Akteri koji iniciraju slučaj korišćenja: Korisnik, Administrator.

# Pogled na logičku arhitekturu sistema

U ovom odeljku je dat pregled logičke arhitekture sistema. Ovaj pogled sadrži opis najznačajnijih klasa, njihove organizacije u pakete i podsisteme, i organizacija podsistema u slojeve. U cilju opisivanja dinamičkih aspekata arhitekture, ovaj odeljak može da uključi opise realizacije najznačajnijih slučajeva korišćenja. Da bi se ilustrovala veza između arhitekturno značajnih klasa, podsistema, paketa ili slojeva moguće je uključiti i odgovarajuće dijagrame klasa.

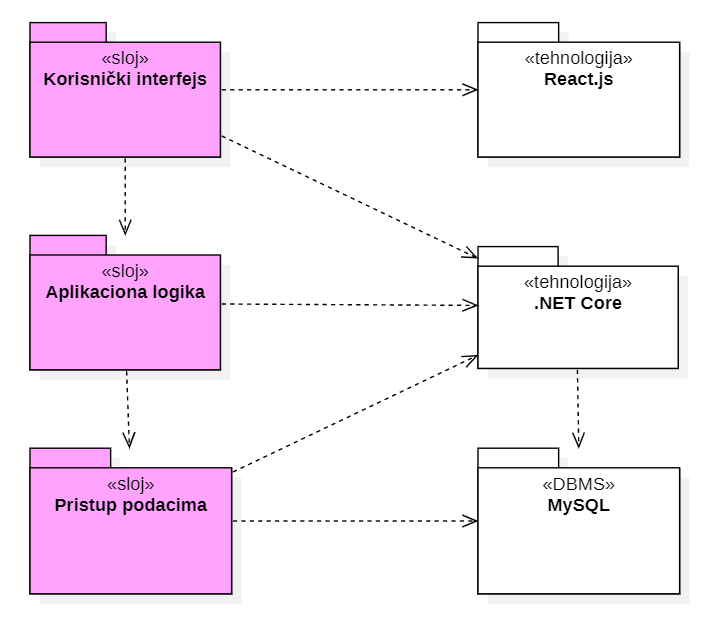
Logički pogled na И/N aplikaciju obuhvata 3 glavna paketa: Korisnički interfejs, Aplikaciona logika, Pristup podacima.

Paket *Korisnički interfejs* sadrži Web stranice, React.js fajlove i multimedijalni sadržaj koji realizuju grafički dizajn i forme preko kojih korisnici sistema komuniciraju sa sistemom.

Paket *Aplikaciona logika* predstavlja srednji sloj sistema koji sadrži .NET fajlove zadužene za realizaciju funkcionalnosti specifičnih za domen sistema koji se razvija.

Paket *Pristup podacima* sadrži MySQL fajlove koji predstavljaju interfejs za pristup, dodavanje i ažuriranje podataka koji se čuvaju u bazi podataka.

## Pregled arhitekture – organizacija paketa i podsistema u slojeve



### Korisnički interfejs

sloj

Ovaj sloj realizuje korisnički interfejs aplikacije. U njemu su sadržane svi React.js i multimedijalni sadržaji koji generišu React stranice preko kojih korisnici komuniciraju sa sistemom.

Sloj korisničkog interfejsa zavisi od sloja aplikacione logike, kao i tehnologija React.js i .NET.

### Aplikaciona logika

sloj

Sloj aplikacione logike je srednji sloj u troslojnoj arhitekturi И/N aplikacije. Sadrži React.js fajlove koji realizuju funkcionalnost karakterističnu za domen primene aplikacije i uspostavljaju vezu između korisničkog interfejsa i sloja za pristup podacima.

Ovaj sloj zavisi od sloja za pristup podacima i .NET paketa.

### Pristup podacima

sloj

Sloj za pristup podacima se nalazi na dnu troslojne arhitekture i sadrži .NET skripte zadužene za pribavljanje, dodavanje i ažuriranje podataka koji se čuvaju u MySQL bazi podataka.

Ovaj sloj ne zavisi od drugih slojeva, ali je zavisan od paketa .NET i MySQL baza podataka.

### React.js

tehnologija

Tehnologija React.js-a definiše gradivne elemente stranica koje se prikazuju u Web čitaču i koje omogućavaju prikaz formatiranih informacija i realizaciju formi za unos i ažuriranje podataka.

### .NET Core

tehnologija

Tehnologija .NET-a obezbeđuje mehanizam za pisanje i izvršavanje fajlova na strani servera, ovi fajlovi pristupaju bazi podataka u cilju pribavljanja, unosa i ažuriranja podataka.

### MySQL

DBMS

MySQL predstavlja sistem za upravljanje bazama podataka koji će se koristiti za realizaciju И/N aplikacije.

# Pogled na procese

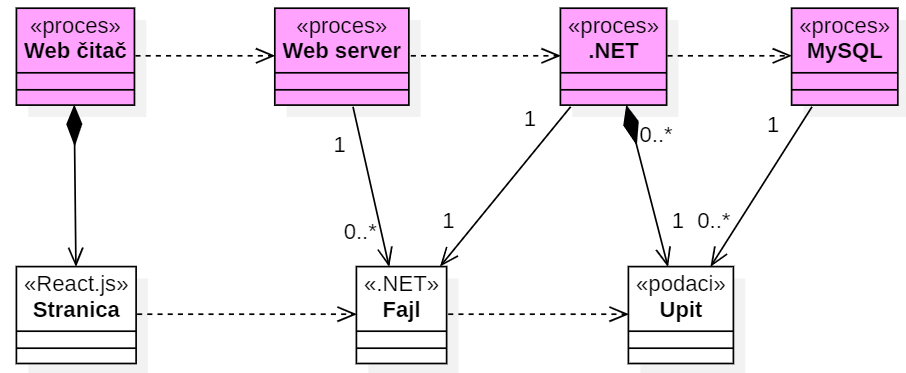
U ovom odeljku je sadržan pogled na procesnu arhitekturu sistema. Ovaj opis treba da sadrži specifikaciju različitih zadataka (procesa i niti) uključenih u rad sistema. Takođe je potrebno dati dijagrame koji pokazuju njihovu interakciju i konfiguraciju. Dodela objekata i klasa na određene zadatke takođe spada u opis procesne arhitekture.

Web aplikacije zasnovane na .NET-u imaju relativno jednostavan procesni model koji je u potpunosti pod kontrolom Web servera. Sa stanovišta projektanta .NET Web aplikacije nije potrebno voditi računa o načinu rada Web servera i načinu izvršavanja fajlova.

Ilustracije radi u nastavku je dat opis procesa uključenih u izvršenje И/N, kao Web aplikacije.

## Procesi

Na sledećem UML dijagramu klasa prikazani su procesi koji učestvuju u izvršenju И/N aplikacije.. Dijagram je opšteg tipa i može se primeniti na bilo koju Web aplikaciju zasnovanu na .NET-u i MySQL bazi podataka.



### Web čitač

Web čitač je proces koji izvršava funkcionalnosti aplikacije za prikaz HTML stranica dobijenih od nekog Web servera. U najopštijem slučaju Web čitač u jednom trenutku može da prikazuje samo jednu HTML stranicu.

Web čitač zavisi od Web servera koji generiše i vraća odgovarajuću HTML stranicu na zahtev.

### Web server

Web server je proces koji izvršava funkcionalnost opsluživanja zahteva prispelih sa više Web čitača. Ukoliko je zahtevana stranica React.js fajl, Web server inicira izvršenje React-DOM procesa koji obrađuje odgovarajući skript i generiše sadržaj koji se vraća čitaču. Web server može paralelno da inicira veći broj React-DOM procesa.

### React-DOM

React-DOM proces obavlja posao obrade zadatog React.js fajla i generiše odgovarajući tekstualni sadržaj koji Web server šalje Web čitaču. Za izvršenje React.js fajlova ovaj proces može da zahteva usluge MySQL servera-a. Komunikacija između React-DOM procesa i MySQL servera se obavlja preko prosleđivanja upita i vraćanja rezultata.

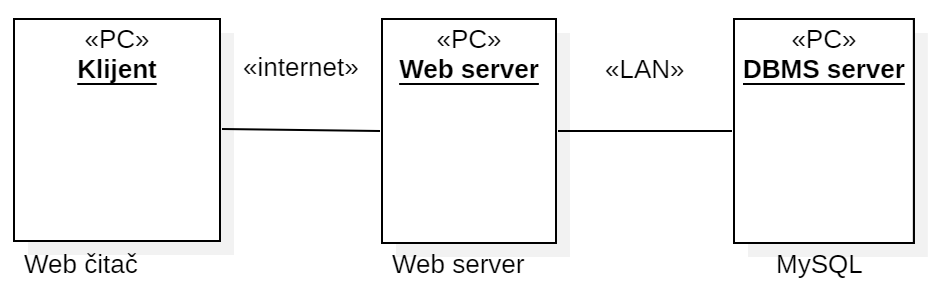
### MySQL

MySQL je sistem baze podataka. Ovaj sistem može konkurentno da prihvati i usluži određen broj upita, izvrši ih nad bazom podataka i vrati rezultate procesu koji je upite postavio.

# Pogled na raspoređivanje sistema

Pogled na raspoređivanje sistema prikazuje različite fizičke čvorove za najopštiju konfiguraciju sistema. Fizičkim čvorovima koji predstavljaju procesore vrši se dodeljivanje identifikovanih procesa.

Na sledećoj slici dat je UML dijagram raspoređivanja И/N Web aplikacije.



## Klijent

Pristup И/N Web stranici se obavlja preko klijentskih računara na kojima radi Web čitač. Za povezivanje između klijenta i Web servera koristi se Internet infrastruktura tako da nema ograničenja u pogledu lokacije klijenta.

## Web server

Računar na kome se izvršava Web server opslužuje više klijenata koji pristupaju preko Interneta. Pored osnovnog procesa koji realizuje funkcionalnost Web servera, na ovom računaru mogu da se izvršavaju i procesi .NET-a koji vrše obradu zadatih zahteva. U najopštioj konfiguraciji DBMS se izvršava na posebnoj mašini koja je sa Web serverom u lokalnoj mreži (LAN).

## DBMS server

DBMS server je računar na kome se nalazi MySQL . Server proces koji realizuje funkcionalnost sistema za upravljanje bazama podataka. Zbog sigurnosti podataka koji se na ovom računaru čuvaju pristup bazi je ograničen samo na računare iz lokalne mreže (LAN).

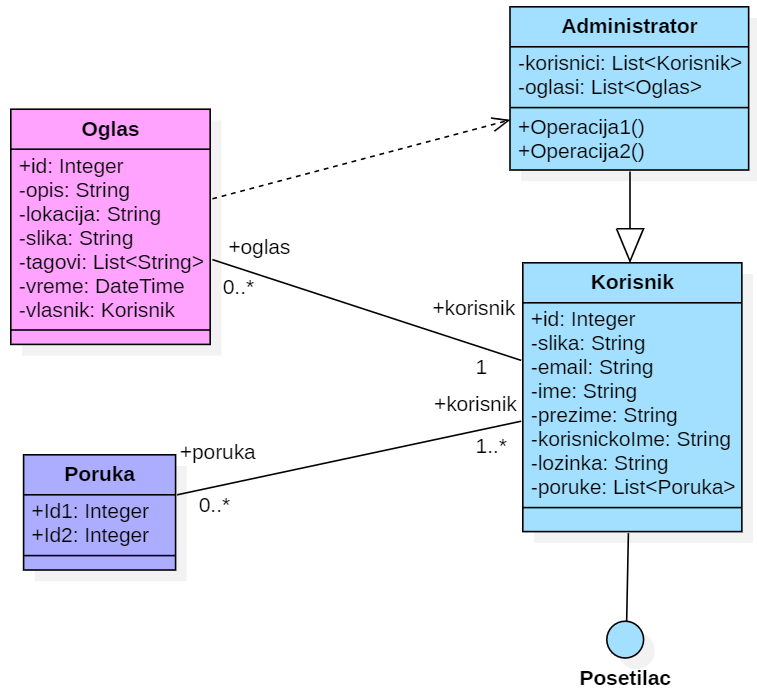
# Pogled na implementaciju sistema

Pogled na implementaciju prikazuje različite aspekte bitne za implementaciju sistema. U slučaju И/N aplikacije, ovaj odeljak sadrži model domena, šemu baze podataka i prikaz komponenti sistema razvrstanih u ranije identifikovane pakete.

## Model domena

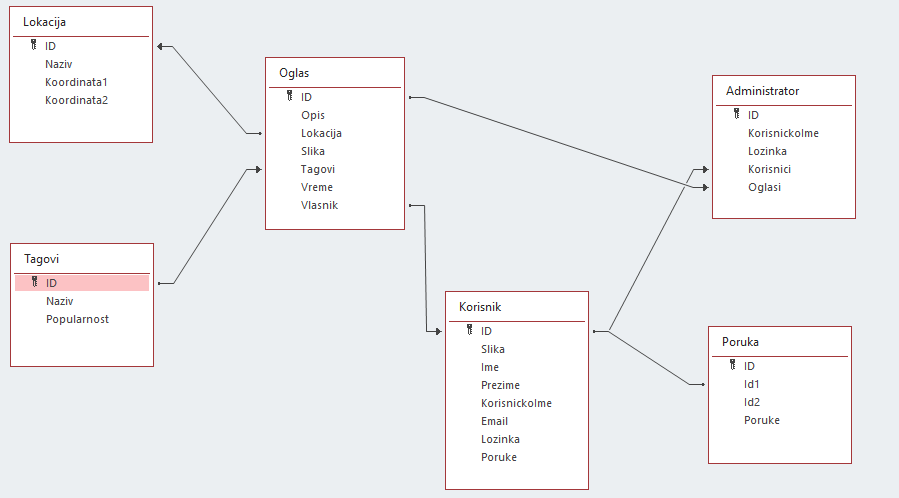
Model domena za koji se И/N aplikacija projektuje je ilustrovan UML dijagramom klasa. U njemu su prikazane domenske klase, neki od njihovih atributa, kao i veze koje se mogu identifikovati između njih.

Model domena predstavlja osnovu za projektovanje baze podataka, ali i identifikaciju nekih od komponenti koje će biti implementirane.



## Šema baze podataka

Detaljna šema baze podataka je prikazana na sledećem dijagramu. Baza podataka i dijagram su kreirani korišćenjem *MS Access*-a, dok je migracija na MySQL obavljena pomoću *MySQL Migration Toolkit*-a.

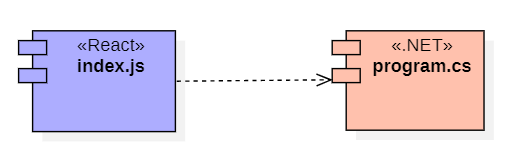


## Komponente sistema

Komponente sistema И/N aplikacije su .NET fajlovi čiji će pregled biti dat po arhitekturnim slojevima. Za ilustraciju će biti korišćeni UML dijagrami komponenti, ali i dijagrami klasa. U slučajevima gde je .NET fajl prikazan kao klasa atributi predstavljaju ulazne podatke koji se uzimaju iz GET ili POST dela HTTP poruke, dok metodi predstavljaju funkcije definisane u okviru fajla.

### Komponente korisničkog interfejsa

Dizajn korisničkog interfejsa je obuhvaćen dvema komponentama:

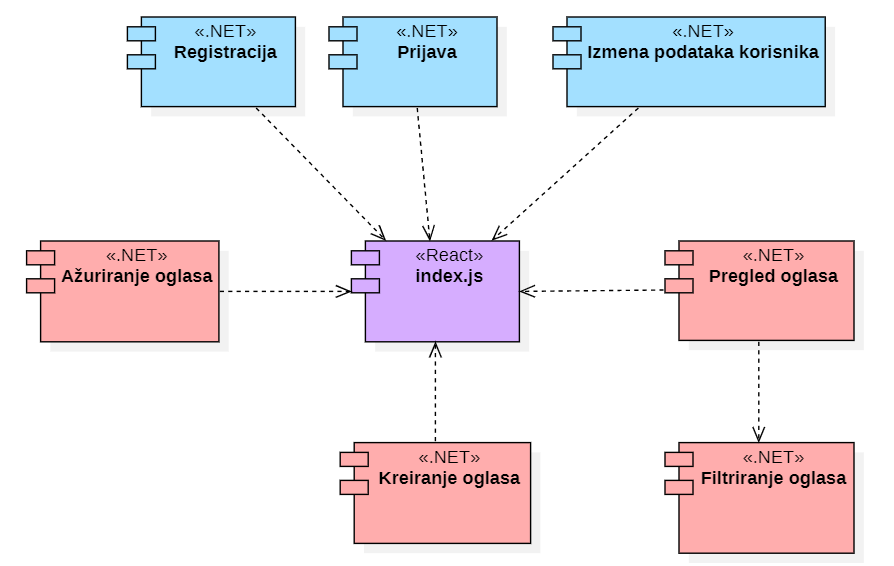


Komponenta **program.cs** implementira stranicu aplikacije čiji sadržaj može da varira od parametra koji joj se proslede u zahtevu.

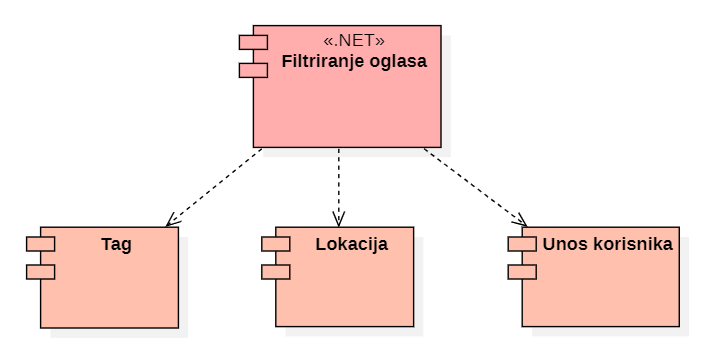
Komponenta **index.js** predstavlja opis strukture stranice i komponenti čiji prikaz zavisi od tipa korisnika i zahteva koji se upute .NET-u.

### Komponente aplikacione logike

Komponente koje realizuju domen problema se uključuju isključivo preko **index.js** komponente korisničkog interfejsa. Na taj način zadržavaju sva podešavanja stila definisana u ovom skriptu. Na sledećem dijagramu su prikazane komponente ovog sloja i njihove međusobne zavisnosti:



* + - 1. *Filtriranje oglasa*



### Komponente za pristup podacima

Pristup bazi podataka se vrši preko HTTP GET, POST, PUT i DELETE metoda iz .NET kontrolera.



# Performanse

Izabrana arhitektura softvera podržava zahteve u pogledu broja korisnika koji mogu simultano pristupati sistemu i vremena odziva za pristup bazi podataka specificirane u zahtevima u pogledu performansi [5]:

1. Sistem će da podrži do 1500 simultanih pristupa korisnika aplikaciji.
2. Vreme potrebno za pristupanje bazi podataka u cilju izvršenje nekog upita ne sme da bude veće od 5 sekundi.

Zahtevane performanse su zadovoljene izborom tehnologija na kojima će sistem biti razvijen i definisane hardverske platforme [5].

# Kvalitet

Izabrana arhitektura softvera podržava zahteve u pogledu dostupnosti i srednjeg vremena između otkaza specificirane u zahtevima u pogledu pouzdanosti [5]:

1. И/N aplikacija će biti dostupana 24 časa dnevno, 7 dana u nedelji. Vreme kada aplikacija nije dostupna ne sme da pređe 5%.
2. Srednje vreme između dva sukcesivna otkaza ne sme da padne ispod 120 sati.