GETOut

Web aplikacija

Arhitekturni projekat

Verzija 1.0

Pregled izmena

|  |  |  |  |
| --- | --- | --- | --- |
| **Datum** | **Verzija** | **Opis** | **Autor** |
| 07.05.2021. | 1.0 | Inicijalna verzija | Mateja Pancic  Vukasin Popovic |
| 23.05.2021 | 1.1 | Izmene | Anđelija Veličković |
|  |  |  |  |
|  |  |  |  |

Sadržaj

1. Cilj dokumenta 5

2. Opseg dokumenta 5

3. Reference 5

4. Predstavljanje arhitekture 5

5. Ciljevi i ograničenja arhitekture 5

6. Pogled na slučajeve korišćenja 5

6.1 Dijagrami slučajeva korišćenja 6

6.2 Kratak opis slučajeva korišćenja 7

6.2.1 Osnovni pregled sajta 7

6.2.2 Registracija 8

6.2.3 Prijavljivanje 8

6.2.4 Ažuriranje profila 8

6.2.5 Ocenjivanje lokala 8

6.2.6 Pregled korisnika 8

6.2.7 Ažuriranje podataka o lokalu 8

6.2.8 Rezervisanje/otkazivanje/izmena rezervacije 8

6.2.9 Brisane/dodavanje/izmena događaja 8

6.2.10 Brisanje korisnika 8

6.2.11 Uređivanje moderatora 8

6.2.12 Brisanje i izmena lokala 8

7. Pogled na logičku arhitekturu sistema 8

7.1 Pregled arhitekture – organizacija paketa i podsistema u slojeve 9

7.1.1 Korisnički interfejs 9

7.1.2 Aplikaciona logika 9

7.1.3 Pristup podacima 9

7.1.4 HTML 9

7.1.5 Angular 10

7.1.6 MySQL 10

8. Pogled na procese 10

8.1 Procesi 10

8.1.1 Web čitač 10

8.1.2 Web server 10

8.1.3 Angular-CGI 10

8.1.4 MySQL Server 11

9. Pogled na raspoređivanje sistema 11

9.1 Klijent 11

9.2 Web server 11

9.3 DBMS server 11

10. Pogled na implementaciju sistema 11

10.1 Model domena 11

10.2 Šema baze podataka 12

10.3 Komponente sistema 13

10.3.1 Komponente korisničkog interfejsa 13

10.3.2 Komponente aplikacione logike 13

10.3.3 Komponente za pristup podacima 13

11. Performanse 14

12. Kvalitet 14

Arhitekturni projekat

# Cilj dokumenta

Cilj ovog dokumenta je detaljni opis arhitekture PeNcIL portala.

# Opseg dokumenta

Dokument se odnosi na PeNcIL portal koji će biti razvijen od strane SWETeam-a. PeNcIL predstavlja skraćenicu za Portal naučno-istraživačke laboratorije. Namena sistema je efikasno prezentovanje, kreiranje i održavanje sadržaja vezanih za rad jedne naučno-istraživačke laboratorije.

# Reference

Spisak korišćene literature:

1. GETOut – Predlog projekta, Outsiders-GETOut-01, V1.0, 2021, Outsiders.
2. GETOut – Planirani raspored aktivnosti na projektu, V1.0, 2021, Outsiders.
3. GETOut – Plan realizacije projekta, V1.0, 2021, Outsiders.
4. GETOut – Vizija sistema, V1.0, 2021, Outsides.
5. GETOut – Specifikacija zahteva, V1.0, 2021, Outsides.

# 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. GETOut će biti implementiran kao Web aplikacija zasnovana na Angular i JavaScript skripting jeziku i MySQL bazi podataka [4].
2. Klijentski deo GETOut aplikacije će biti optimizovan za sledeće Web čitače: Internet Explorer 6.0 i noviji, Opera 8.0 i noviji, Firefox (Mozilla) i Google Chrome [4].
3. Svi zahtevi u pogledu performansi dati u [5] moraju biti uzeti u obzir pri izboru arhitekture i razvoju sistema.

# Pogled na slučajeve korišćenja

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

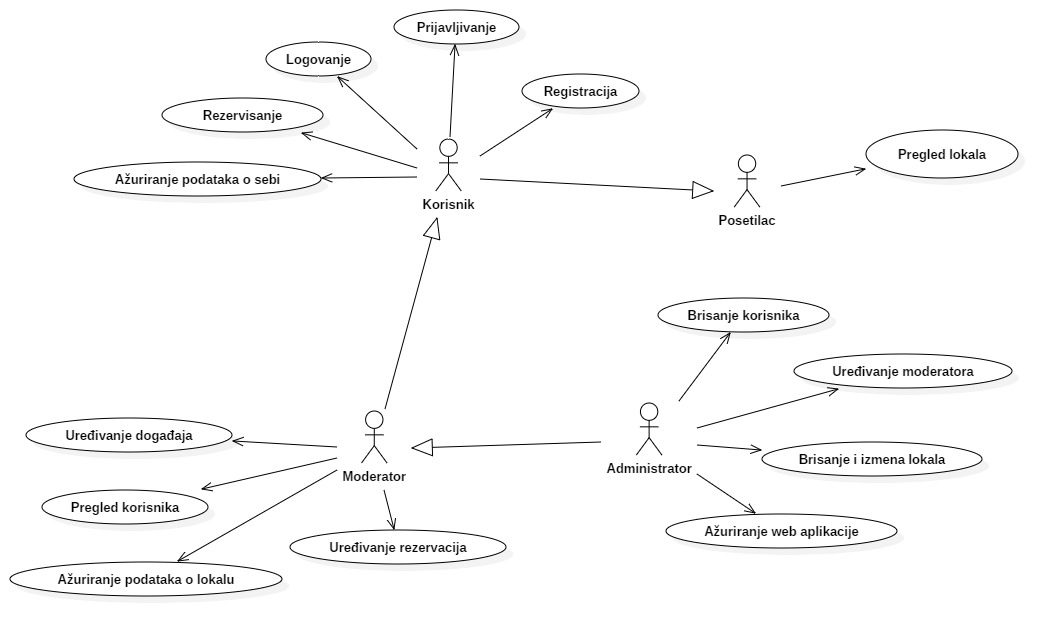
Slučajevi korišćenja GETOut aplikacije su:

* Osnovni pregled sajta
* Registracija
* Prijavljivanje
* Ažuriranje profila
* Fukcionalnosti sajta
  + Ocenjivanje lokala
  + Pregled korisnika
  + Ažuriranje podataka o lokalu
    - Dostupna mesta
    - Događaji
  + Rezervisanje/otkazivanje/izmena rezervacije
  + Brisanje/dodavanje/izmena događaja
  + Brisanje korisnika
  + Uređivanje moderatora
    - Dodaj moderatora
    - Ukloni moderatorska prava
  + Brisanje i izmena lokala
    - Dodaj lokal
    - Obrisi/izmeni lokal

Ove slučajevi korišćenja mogu da iniciraju posetilac posetilac, korisnik, moderator ili administrator.

## Dijagrami slučajeva korišćenja

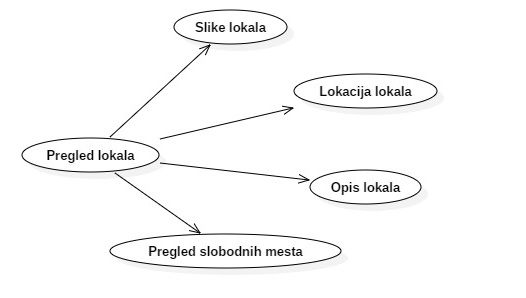
Osnovni UML dijagram koji prikazuje korisnike i slučajeve korišćenja GETOut portala prikazan je na sledećoj slici:



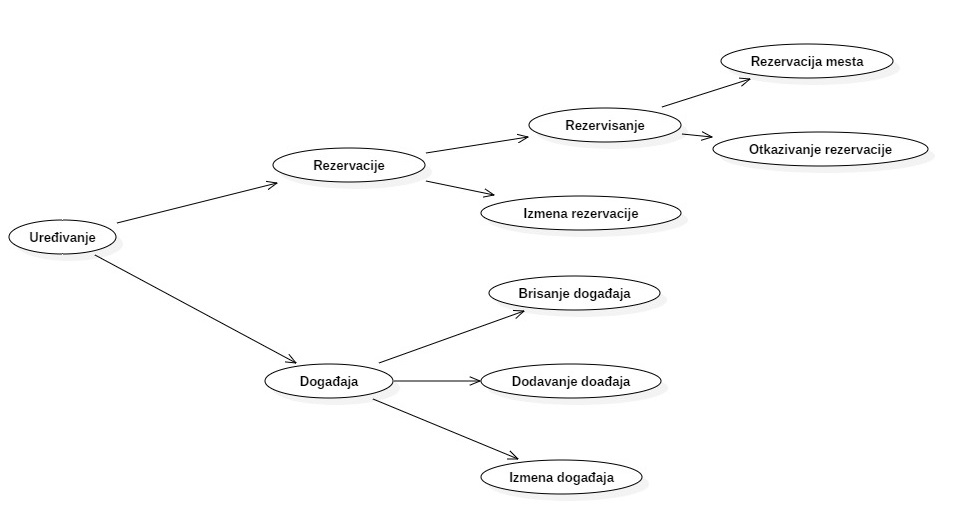
Slučajevi korišćenja *pregled lokala, uređivanje događaja i rezervacije, izmena, rezervisanje i brisanje rezervacije obuhvataju* složenije radnje koje se mogu razložiti dalje na pojedinačne slučajeve korišćenja.

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

Detaljni UML dijagram za slučaj korišćenja *kreiranje, brisanje i arhiviranje članova* je prikazan na sledećoj slici:



Detaljni UML dijagram za slučaj korišćenja *uređivanje događaja i rezervacije, izmena, rezervisanje i brisanje rezervacije* je prikazan na sledećoj slici:



## Kratak opis slučajeva korišćenja

### Osnovni pregled sajta

Kratak opis: Prikaz stranice Web aplikacije sa svim neophodnim informacijama o objektu.

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

### Registracija

Kratak opis: Prikaz forme za registraciju radi mogucnosti prijavljivanja.

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

### Prijavljivanje

Kratak opis: Prijavljivanje korisnika u cilju pristupa specifičnim funkcijama koje zahtevaju autorizaciju.

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

### Ažuriranje profila

Kratak opis: Prijavljivanje korisnika na Web aplikaciju, a potom i izmena sopstvenih informacija na profilu.

Akteri koji iniciraju slučaj korišćenja: Korisnik, Administrator.

### Ocenjivanje lokala

Kratak opis: U okviru detaljnog prikaza oglasa, korisnik može da oceni lokal.

Akteri koji iniciraju slučaj korišćenja: Korisnik.

### Pregled korisnika

Kratak opis: Pregled informacija o svim korisnicima.

Akteri koji iniciraju slučaj korišćenja: Moderator, Administrator.

### Ažuriranje podataka o lokalu

Kratak opis: Mogućnost promene izgleda sajta i ažuriranja dostupnih mesta i događaja.

Akteri koji iniciraju slučaj korišćenja: Moderator, Administrator.

### Rezervisanje/otkazivanje/izmena rezervacije

Kratak opis: Rezervacija slobodnog mesta ili otkazivanje postojeće rezervacije.

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

### Brisane/dodavanje/izmena događaja

Kratak opis: Dodavanje, izmena ili brisanje događaja.

Akteri koji iniciraju slučaj korišćenja: Administrator.

### Brisanje korisnika

Kratak opis: Brisanje postojećih korisničkih naloga.

Akteri koji iniciraju slučaj korišćenja: Član laboratorije, Šef laboratorije, Vođa projekta, Administrator.

### Uređivanje moderatora

Kratak opis: Davanje ili oduzimanje moderatorskih prava.

Akteri koji iniciraju slučaj korišćenja: Administrator.

### Brisanje i izmena lokala

Kratak opis: Izmena/dodavanje ili brisanje lokala.

Akteri koji iniciraju slučaj korišćenja: 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.

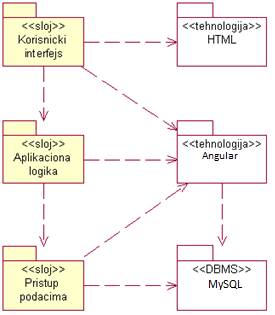
Logički pogled na GETOut web aplikaciju obuhvata 3 glavna paketa: Korisnički interfejs, Aplikaciona logika, Pristup podacima.

Paket *Korisnički interfejs* sadrži Web stranice, Angular skripte 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 Angular skripte zadužene za realizaciju funkcionalnosti specifičnih za domen sistema koji se razvija.

Paket *Pristup podacima* sadrži Angular skripte koje 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 portala. U njemu su sadržane sve HTML, multimedijalni sadržaji i Angular skripte koje generišu HTML stranice preko kojih korisnici komuniciraju sa sistemom.

Sloj korisničkog interfejsa zavisi od sloja aplikacione logike, kao i paketa HTML i Angular.

### Aplikaciona logika

sloj

Sloj aplikacione logike je srednji sloj u troslojnoj arhitekturi GETOut web aplikacije. Sadrži Angular skripte koje realizuju funkcionalnost karakterističnu za domen primene portala i uspostavljaju vezu između korisničkog interfejsa i sloja za pristup podacima.

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

### Pristup podacima

sloj

Sloj za pristup podacima se nalazi na dnu troslojne arhitekture i sadrži Angular 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 Angular i MySQL baza podataka.

### HTML

tehnologija

Tehnologija HTML-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.

### Angular

tehnologija

Tehnologija Angular -a obezbeđuje mehanizam za pisanje i izvršavanje skripti na strani servera. Ove skripte mogu da generišu HTML kod koji realizuje korisnički interfejs i 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 GETOut web aplikacije.

# Pogled na procese

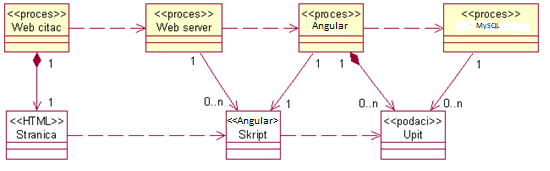
U ovom odeljku je sadržan pogled na procesnu arhitekturu sistema. Ovaj opis treba da sadrži specifikaciju različitih zadataka (procesa i niti) uključenih u rad sistema. Takođe je potrebno dati dijagrame koji pokazuju njihovu interakciju i konfiguraciju. Dodela objekata i klasa na određene zadatke takođe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

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

Ilustracije radi u nastavku je dat opis procesa uključenih u izvršenje GETOut kao Web aplikacije.

## Procesi

Na sledećem UML dijagramu klasa prikazani su procesi koji učestvuju u izvršenju PeNcIL portala. Dijagram je opšteg tipa i može se primeniti na bilo koju Web aplikaciju zasnovanu na PHP-u i MySQL bazi podataka.



### Web čitač

Web čitač je proces koji izvršava funkcionalnost 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 Angular skript, Web server inicira izvršenje Angular-CGI procesa koji obrađuje odgovarajući skript i generiše sadržaj koji se vraća čitaču. Web server može paralelno da inicira veći broj Angular-CGI procesa.

### Angular-CGI

Angular-CGI proces obavlja posao obrade zadatog Angular skripta i generiše odgovarajući tekstualni sadržaj koji Web server šalje Web čitaču. Za izvršenje Angular skripta ovaj proces može da zahteva usluge MySQL servera. Komunikacija između Angular-CGI procesa i MySQL servera se obavlja preko prosleđivanja upita i vraćanja rezultat.

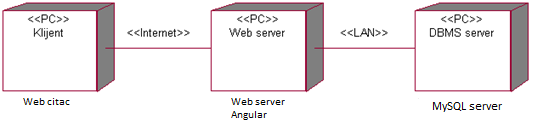
### MySQL Server

MySQL Server je proces koji izvršava funkcionalnost MySQL sistema za upravljanje bazama podataka. Ovaj proces može konkurentno da prihvati 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 GETOut web aplikacije.



## Klijent

Pristup GETOut web aplikaciji se obavlja preko klijentskih računara na kojima se izvršava 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,. 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 izvršava 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 GETOut web aplikacije ovaj odeljak sadrži model domena, šemu baze podataka i prikaz komponenti sistema razvrstanih u ranije identifikovane pakete.

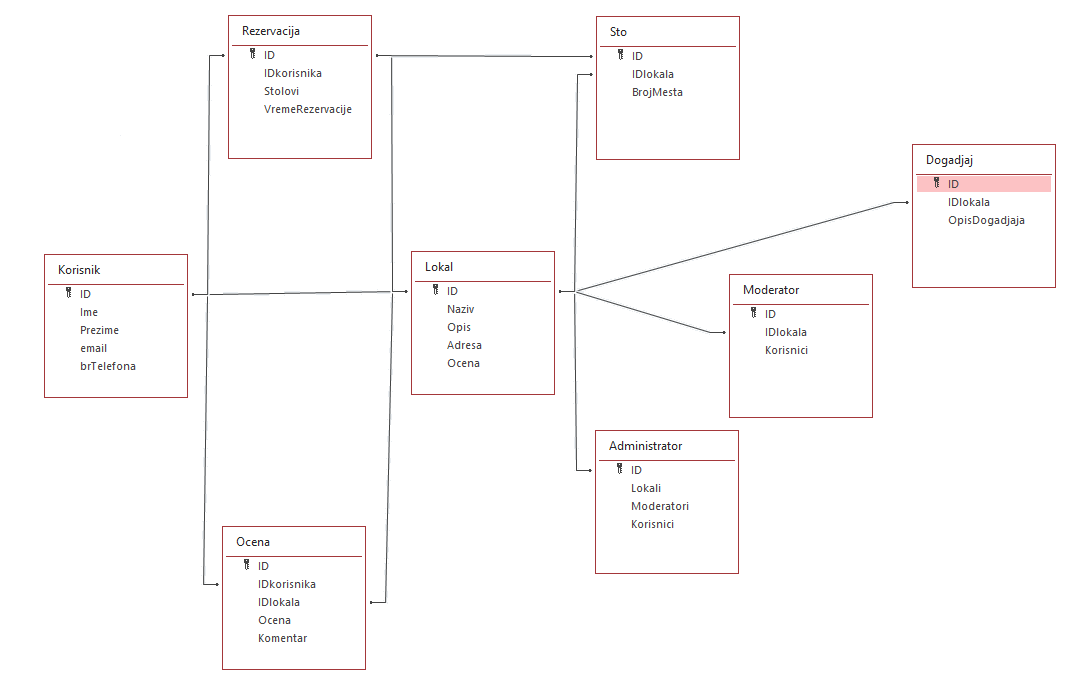
## Model domena

Model domena za koji se GETOut web 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.



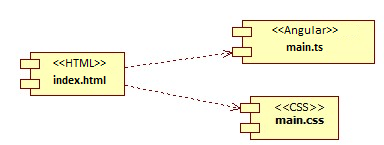
## Komponente sistema

*NAPOMENA: GETOut web aplikacija nije kompletno implementiran, tj. implementiran je na nivou arhitekturnog prototipa. Zbog ovoga sledeću sekciju trebate shvatiti kao ilustraciju kako bi dokument trebao da izgleda. Opis komponenti sistema za vaše projekte mora biti kompletan! Procena složenosti vašeg rešenja će zavisiti od opisa sistema koji ovde date.*

Komponente sistema GETOut su Angular skripte čiji će pregled biti dat po arhitekturnim slojevima. Za ilustraciju će biti korišćeni UML dijagrami komponenti, ali i dijagrami klasa. U slučajevima gde je PHP skript prikazan kao klasa atributi predstavljaju ulazne podatke koji se uzimaju iz GET ili POST dela HTTP poruke, dok metodi predstavljaju funkcije definisane u okviru skripta.

### Komponente korisničkog interfejsa

Dizajn korisničkog interfejsa je obuhvaćen komponentama sadržanim na Bootstrap stranici, povezanoj sa aplikacijom.

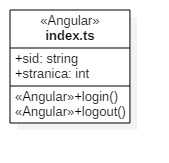


Komponenta **index.html** je implementira stranicu portala čiji sadržaj može da varira od parametra koji joj se proslede u zahtevu.

Komponenta **main.css** predstavlja opis stilova za pojedine HTML elemente koji se javljaju na različitim stranicama.

Komponenta **main.ts** predstavlja funkcionalnost portala i vezu sa slojem aplikacione logike.

Parametri koji utiču na izbor i jezik za prikaz stranice ilustrovani su sledećim dijagramom klasa:

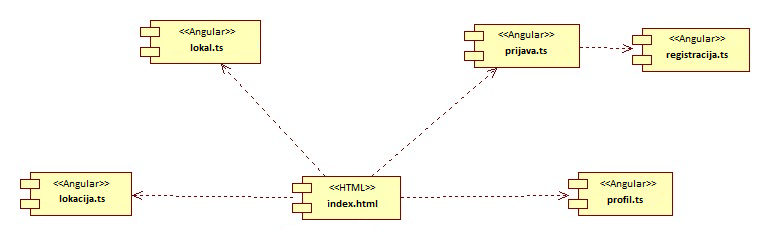


Značenje atributa je sledeće:

* stranica – interni identifikator stranice (pocetna, detaljniji prikaz lokala)
* sid – identifikator sesije kada je korisnik ulogovan

### Komponente aplikacione logike

Komponente koje realizuju domen problema se uključuju isključivo preko **index.html** 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:



**Registracija:** Angular skripta koja preko JSON-a dobija podatke potrebne za registraciju korisnika i prosleđuje ih bazi.

**Prijava:** Angular skripta koja preko JSON-a dobija podatke neophodne za prijavu korisnika, a potom ih prosleđuje bazi dobijene podatke, da bi se u bazi proverilo da li takav korisnik uopšte postoji u bazi. U ovoj skripti treba da se vrši i proveri da li se username i password korisnika koji se prijavljuje poklapaju sa podacima korisnika u bazi. Angular skripta za prijavu stranici treba da vrati jedinstveni identifikator prijavljenog korisnika.

**Profil:** Angular skripta koja iz baze preuzima podatke o korisniku na osnovu jedinstvenog identifikatora prijavljenog korisnika i prikazuje podatke o njemu.

**Lokal:** Angular skripta koja iz baze preuzima podatke o lokalu na osnovu jedinstvenog identifikatora lokala i prikazuje podatke o njemu.

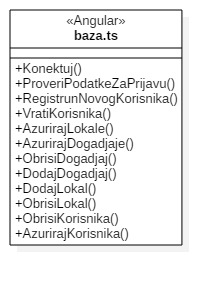
**Lokacija:** Angular skripta koja iz baze preuzima podatke o lokaciji na osnovu jedinstvenog identifikatora lokala i prikazuje je.

### Komponente za pristup podacima



Pristup bazi podataka je u potpunosti zatvoren u funkcije koje su definisane u okviru Angular skripta **app.ts**. Pomenuti skript se uključuje na početku index.php-a, tako da su sve funkcije za pristup podacima dostupne svim komponentama.

Na sledećem UML dijagramu klasa pobrojane su funkcije za pristup podacima iz baze:



Navedene funkcije obavljaju sledeće zadatke:

* **Konektuj():** povezivanje na bazu podataka, poziva se u samom skriptu
* **ProveriPodatkeZaprijavu():** funkcija koja proverava da li se korisnik koji se prijavljuje nalazi u bazi podataka
* **RegitrujNovogKorisnika():** funkcija kojom se dodaje novi korisnik u bazu podataka
* **VratiKorisnika():** funkcija koja vraća podatke o konkretnom korisniku
* **AzurirajLokal():** funkcija kojom se izmenjuju podaci o konkretnom lokalu i azuriraju se ti podaci u bazi podataka
* **AzurirajDogadjaj():** funkcija kojom se izmenjuju podaci o konkretnom dogadjaju unutar lokala i azuriraju se ti podaci u bazi podataka
* **ObrisiLokal():** funkcija kojom se konkretan lokala uklanja iz baze podataka
* **ObrisiDogadjaj():** funkcija kojom se konkretan dogadjaj unutar lokala uklanja iz baze podataka
* **DodajLokal():** funkcija kojom se dodaje novi lokal u bazi podataka
* **DodajDogadjaj():** funkcija kojom se dodaje novi dogadjaj unutar lokala u bazi podataka
* **AzurirajKorisnika():** funkcija kojom se izmenjuju podaci o konkretnom korisniku i azuriraju se ti podaci u bazi podataka
* **ObrisiKorisnika():** funkcija kojom uklanja konkretan korisnik iz baze podataka

# 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 1000 simultanih pristupa korisnika portalu.
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. GETOut web aplikacija će biti dostupna 24 časa dnevno, 7 dana u nedelji. Vreme kada portal nije dostupan ne sme da pređe 10%.
2. Srednje vreme između dva sukcesivna otkaza ne sme da padne ispod 120 sati.