ZurkaNaKlik

Arhitekturni projekat

Verzija 1.0

Pregled izmena

|  |  |  |  |
| --- | --- | --- | --- |
| **Datum** | **Verzija** | **Opis** | **Autor** |
| 18.04.2024. | 1.0 | Inicijalna verzija | Jana, Teodora, Mihailo, Dusan |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

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 8

6.2.1 Izbor jezika 8

6.2.2 Pregled osnovnih podataka o laboratoriji 8

6.2.3 Pregled spiska članova 8

6.2.4 Pregled podataka o određenom članu laboratorije 8

6.2.5 Pregled publikacija po autoru 8

6.2.6 Pregled publikacija po tipu 8

6.2.7 Pregled publikacija po godini 8

6.2.8 Pregled spiska projekata 8

6.2.9 Pregled podataka o određenom projektu 8

6.2.10 Prijavljivanje 9

6.2.11 Ažuriranje podataka o članu 9

6.2.12 Dodavanje nove publikacije 9

6.2.13 Brisanje postojeće publikacije 9

6.2.14 Ažuriranje osnovnih podataka o laboratoriji 9

6.2.15 Kreiranje novog člana 9

6.2.16 Brisanje postojećeg člana 9

6.2.17 Arhiviranje postojećeg člana 9

6.2.18 Kreiranje projekta i postavljanje vođe 9

6.2.19 Ažuriranje podataka o projektu 9

7. Pogled na logičku arhitekturu sistema 9

7.1 Pregled arhitekture – organizacija paketa i podsistema u slojeve 10

7.1.1 Korisnički interfejs 10

7.1.2 Aplikaciona logika 10

7.1.3 Pristup podacima 10

7.1.4 HTML 10

7.1.5 PHP 11

7.1.6 React 11

7.1.7 MySQL 11

8. Pogled na procese 11

8.1 Procesi 11

8.1.1 Web čitač 11

8.1.2 Web server 11

8.1.3 PHP-CGI 11

8.1.4 MySQL Server 12

9. Pogled na raspoređivanje sistema 12

9.1 Klijent 12

9.2 Web server 12

9.3 DBMS server 12

10. Pogled na implementaciju sistema 12

10.1 Model domena 12

10.2 Šema baze podataka 13

10.3 Komponente sistema 14

10.3.1 Komponente korisničkog interfejsa 14

10.3.2 Komponente aplikacione logike 14

10.3.3 Komponente za pristup podacima 15

11. Performanse 16

12. Kvalitet 16

Arhitekturni projekat

# Cilj dokumenta

Cilj ovog dokumenta je detaljni opis arhitekture ZurkaNaKlik web aplikacije.

# Opseg dokumenta

Dokument se odnosi na ZurkaNaKlik web aplikaciju kojia će biti razvijena od strane LeVl-a.

# Reference

Spisak korišćene literature:

1. ZurkaNaKlik – Predlog projekta, LeVl- ZurkaNaKlik -01, V1.0, 2024, LeVl.
2. ZurkaNaKlik – Planirani raspored aktivnosti na projektu, V1.0, 2024, LeVl.
3. ZurkaNaKlik – Plan realizacije projekta, V1.0, 2024, LeVl..
4. ZurkaNaKlik – Vizija sistema, V1.0, 2024, LeVl.
5. ZurkaNaKlik – Specifikacija zahteva, V1.0, 2024, LeVl.

# 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. ZurkaNaKlik portal će biti implementiran kao Web aplikacija zasnovana na .NET jeziku, React framework-u i MySQL bazi podataka [4].
2. Klijentski deo ZurkaNaKlik web aplikacije će biti optimizovan za sledeće Web čitače: Internet Explorer 6.0 i noviji, 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.

# Pogled na slučajeve korišćenja

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

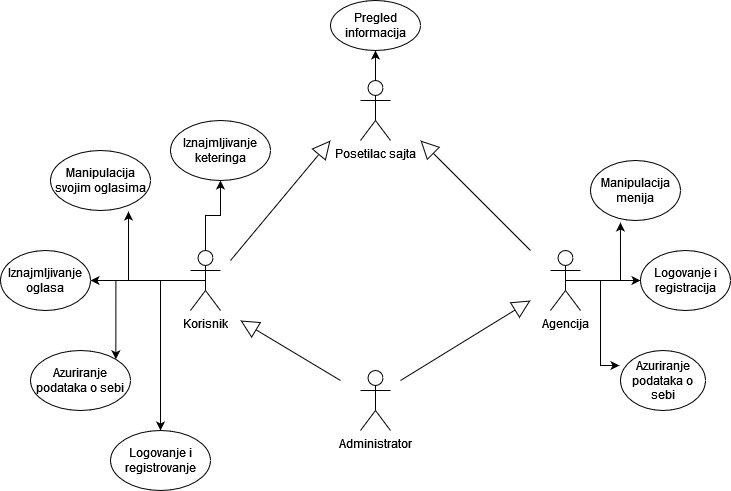
Slučajevi korišćenja ZurkaNaKlik web aplikacije su:

* *Pregled informacija*
  + Pregled osnovnih podataka o sajtu
  + *Dodavanje oglasa*
    - Dodavanje cene u svom oglasu
    - Dodavanje slika
    - Dodavanje dodatnih informacija
    - Izmena cena
    - Izmena slika i dodatih informacija oglasa
    - Brisanje oglasa
* Prijavljivanje
* Registrovanje
* Ažuriranje podataka o korisniku/ Agenciji za ketering
* *Pregled menija*
  + Dodavanje menija
  + Brisanje postojećeg menija
  + Azuriranje podataka o meniju

Ove slučajevi korišćenja mogu da iniciraju posetilac aplikacije, korisnik aplikacije, agencija za ketering i administrator.

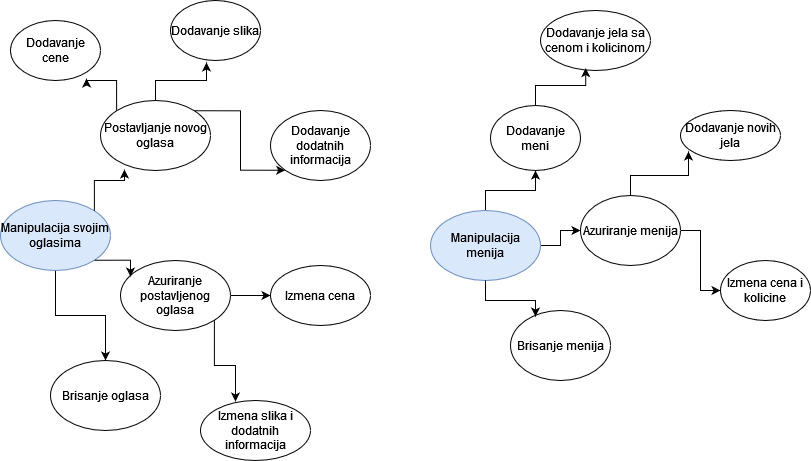
## Dijagrami slučajeva korišćenja

Osnovni UML dijagram koji prikazuje korisnike i slučajeve korišćenja ZurkaNaKlik web aplikacije prikazana je na sledećoj slici:

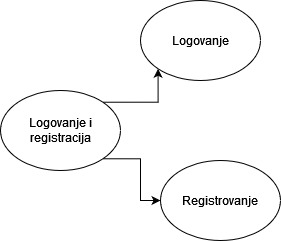


Slučajevi korišćenja pregled informacija i modifikacija oglasa i meija,logovanje i registracija obuhvataju složenije radnje koje se mogu razložiti dalje razložiti na pojedinačne slučajeve korišćenja.

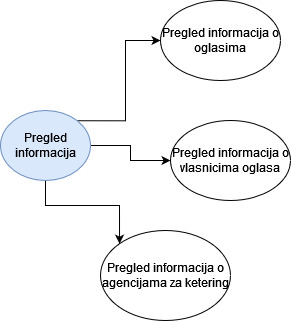
Detaljni UML dijagram za slučaj korišćenja modifikacija oglasa i menija je prikazan na sledećoj slici:



Detaljni UML dijagram za slučaj korišćenja logovanje i registrovanje je prikazan na sledećoj slici:



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



## Kratak opis slučajeva korišćenja

### Pregled osnovnih podataka o sajtu

**Kratak opis:**

Prikaz stranice sajta sa osnovnim podacima o postavljenim oglasima.

**Akteri:**

Posetilac sajta, Korisnik,Agencija za ketering, Administrator.

### Pregled osnovnih podataka o laboratoriji

**Kratak** **opis**: Prikaz stranice portala sa osnovnim podacima o laboratoriji.

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

### Dodavanje oglasa

**Kratak opis:**Dodavanje novog oglasa od strane Korisnika portala.

**Akteri**:Korisnik portala, Administrator.

### Dodavanje cene u svom oglasu

**Kratak opis:**

Korisnik portala moze dodati cenu objektu kojeg ce oglasiti.

**Akteri:**

Korisnik portala.

### Dodavanje slika

**Kratak opis:**

Korisnik portala dodaje sliku objektu koji oglasava.

**Akteri:**

Korisnik portala.

### Dodavanje dodatnih informacija

**Kratak opis:**

Korisnik moze prosiriti svoj oglas sa nekim dodatnim informacijama.

**Akteri:**

Korisnik portala.

### Izmena cena

**Kratak opis:**

Prikaz stranice sa oglasom kojeg treba azurirati.

**Akteri:**

Korisnik portala, Administrator.

### Izmena slika i dodatnih informacija oglasa

**Kratak opis:**

Prikaz stranice sa oglasom kojeg treba azurirati.

**Akteri:**

Korisnik portala, Administrator.

### Brisanje oglasa

**Kratak opis:**

Prikaz stranice sa svojim oglasima.

**Akteri:**

Korisnik portala, Administrator.

### Prijavljivanje

**Kratak opis:**

Logovanje korisnika na portal u cilju pristupa specifičnim funkcijama koje zahtevaju autorizaciju.

**Akteri:**

Korisnik portala,Agencija za ketering , Administrator.

### Ažuriranje podataka o Korisniku/Agenciji za ketering

**Kratak opis:**

Ažuriranje podataka o sebi od strane prijavljenog korisnika portala.

**Akteri:**

Korisnik portala, Agencija za ketering, Administrator.

### Dodavanje menija

**Kratak opis:**

Dodavanje novog menija od strane Agencije za ketering.

**Akteri:**

Agencija za ketering, Administrator.

### Brisanje postojećeg menija

**Kratak opis:**

Biranje postojećegmenija od strane agencije za ketering ili administratora.

**Akteri:**

Agencija za ketering, Administrator.

### Ažuriranje osnovnih podataka o meniju

**Kratak opis:**

Izmena podataka o meniju.

**Akteri:**

Agencija za ketering, Administrator.

### Registrovanje

**Kratak opis:**

Kreiranje korisničkog naloga za novog korisnika/agenciju.

**Akteri:**

Korisnik portala, Agencija za ketering, Administrator.

### Iznajmljivanje oglasa i keteringa

**Kratak opis:**

Iznajmljivanje oglasa a zatim i keteringa za tu lokaciju.

**Akteri:**

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.

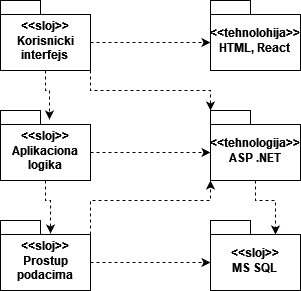
Logički pogled na ZurkaNaKlik web aplikacija obuhvata 3 glavna paketa: Korisnički interfejs, Aplikaciona logika, Pristup podacima.

Paket *Korisnički interfejs* obuhvata web stranice i multimedijalni sadržaj koji su realizovani korišćenjem React biblioteke za izradu korisničkih interfejsa u kombinaciji sa .NET tehnologijama. Ovaj paket omogućava kreiranje grafičkog dizajna i formi preko kojih korisnici sistema mogu da komuniciraju sa aplikacijom.

Paket *Aplikaciona logika* predstavlja srednji sloj sistema koji sadrži .NET C# skripte zadužene za implementaciju funkcionalnosti specifičnih za domen sistema koji se razvija. Ove skripte upravljaju poslovnim logikama i procesima aplikacije.

Paket *Pristup* *podacima* obuhvata .NET C# skripte koje predstavljaju interfejs za pristup, dodavanje i ažuriranje podataka koji se čuvaju u bazi podataka. Ovi skriptovi omogućavaju aplikaciji da komunicira sa bazom podataka i izvršava operacije nad podacima.

## Pregled arhitekture – organizacija paketa i podsistema u slojeve



### Korisnički interfejs sloj

### Ovaj sloj implementira korisnički interfejs portala. Sadrži sve React komponente za kreiranje korisničkog interfejsa, kao i multimedijalni sadržaj.

### Sloj korisničkog interfejsa zavisi od sloja aplikacione logike, kao i od paketa ASP .NET C#.

### Aplikaciona logika sloj

### Sloj aplikacione logike je srednji sloj u arhitekturi portala. Sadrži .NET C# skripte koje realizuju funkcionalnosti specifične za domen primene portala i uspostavljaju vezu između korisničkog interfejsa i sloja za pristup podacima.

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

### Pristup podacima

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

Ovaj sloj ne zavisi od drugih slojeva, ali je zavisan od paketa .NET C# i MS SQL 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.

### React

Tehnologija, koristi se za kreiranje reusable komponenta i za dinamičku promenu sadržaja veb stranice u zavisnosti od trenutnog stanja aplikacije. Pri renderovanju generiše HTML sadržaj za datu stranicu kojoj korisnik pristupa.

### .NET C# tehnologija

.NET C# tehnologija omogućava razvoj aplikacija na Microsoft platformi. Obezbeđuje mehanizam za pisanje i izvršavanje skripti na serverskoj strani, kao i pristupanje bazi podataka radi manipulacije podacima.

### MS SQL

DBMS

MS SQL predstavlja sistem za upravljanje bazama podataka koji će se koristiti za realizaciju ZurkaNaKlik portala.

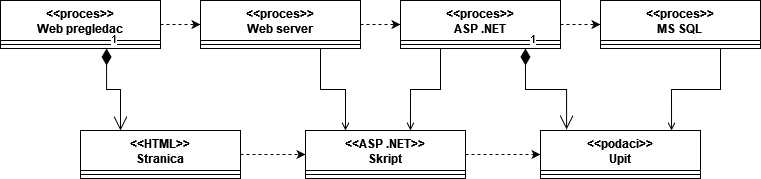
# Pogled na procese

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.

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

## Procesi

Na sledećem UML dijagramu klasa prikazani su procesi koji učestvuju u izvršenju web aplikacije ZurkaNaKlik. Dijagram je opšteg tipa i može se primeniti na bilo koju Web aplikaciju zasnovanu na ASP .NET i MS SQL bazi podataka.



### Web čitač

Web pregledac je proces koji izvršava funkcionalnost aplikacije za prikaz React komponenti i HTML stranica dobijenih od web servera. U najopštijem slučaju, Web čitač može u jednom trenutku prikazivati samo jednu stranicu ili komponentu. Web pregledac zavisi od web servera koji generiše i vraća odgovarajući sadržaj na zahtev.

### Web server

Web server je proces koji izvršava funkcionalnost opsluživanja zahteva prispelih od više Web čitača. Ukoliko je zahtevana stranica ili komponenta generisana korišćenjem .NET C# skripti, Web server ih obrađuje i vraća odgovarajući sadržaj čitačima. Web server može paralelno da obrađuje više zahteva.

### .NET C# backend proces

.NET C# Backend proces obavlja posao obrade zadatih zahteva i generiše odgovarajući sadržaj koji se šalje Web čitaču. Za obradu zahteva, ovaj proces može da zahteva usluge baze podataka. Komunikacija između .NET C# Backend procesa i baze podataka se obavlja putem slanja upita i vraćanja rezultata.

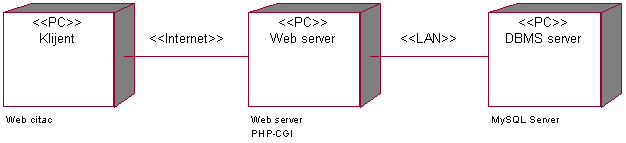
### MySQL Server

MS SQL Server je proces koji izvršava funkcionalnost MS SQL 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 ZurkaNaKlik portala.



## Klijent

Pristup web aplikaciji ZurkaNaKlik se obavlja preko klijentskih računara na kojima se pokrece Web pregledac. Za povezivanje između klijenta i Web servera koristi se Internet infrastruktura tako da nema ograničenja u pogledu lokacije klijenta.

## Web server

U ovoj opštoj konfiguraciji, računar na kojem se izvršava Web server opslužuje više klijenata koji pristupaju preko Interneta. Osim osnovnog procesa koji obavlja funkcionalnost Web servera, na istom računaru mogu se izvršavati i procesi za obradu C# komponenti. Baza podataka se izvršava na posebnoj mašini u lokalnoj mreži (LAN) i služi za upravljanje podacima. Komunikacija između Web servera i baze podataka se odvija putem mreže.

## DBMS server

DBMS server je računar na kome se izvršava MS SQL 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).

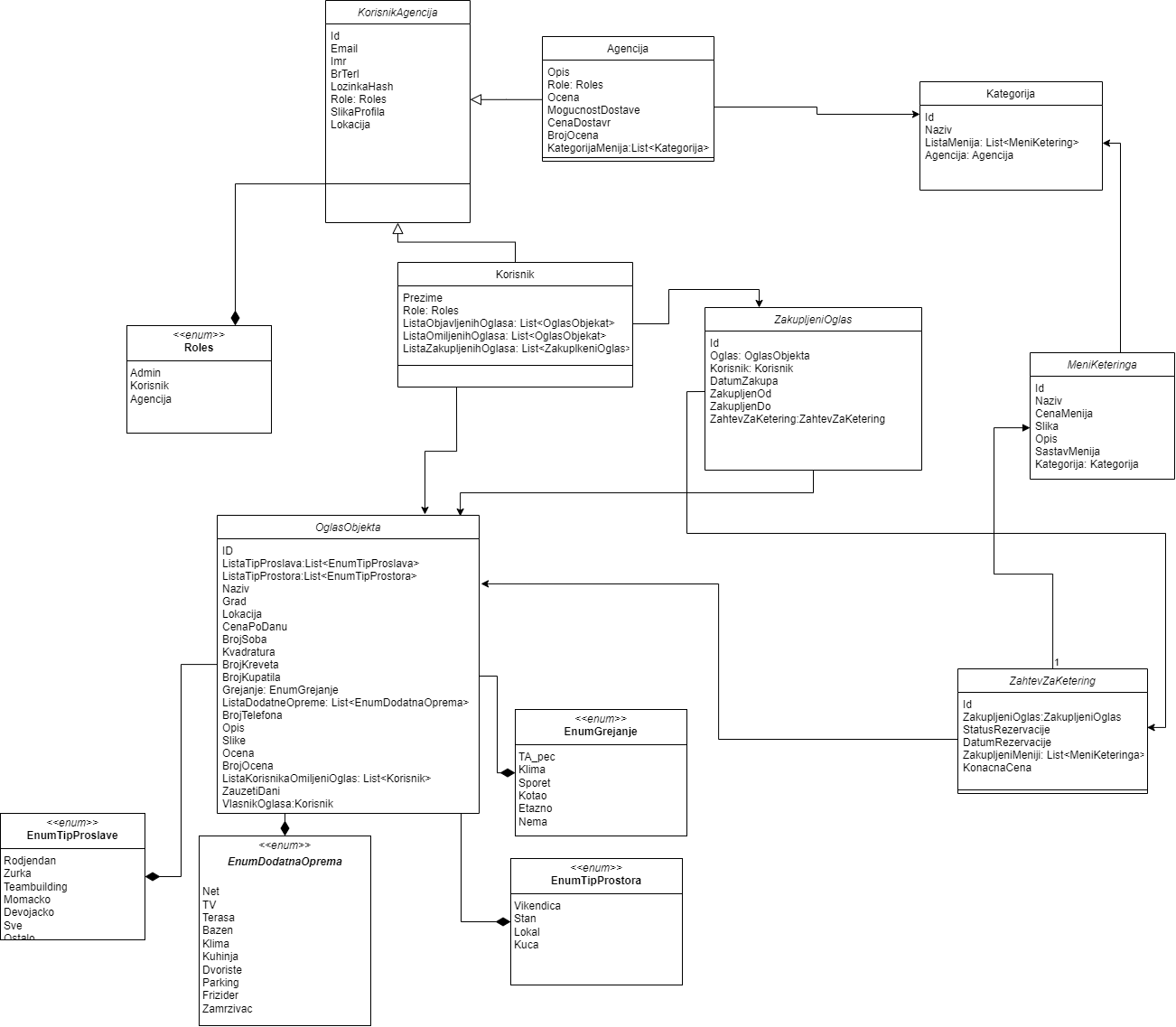
# Pogled na implementaciju sistema

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

## Model domena

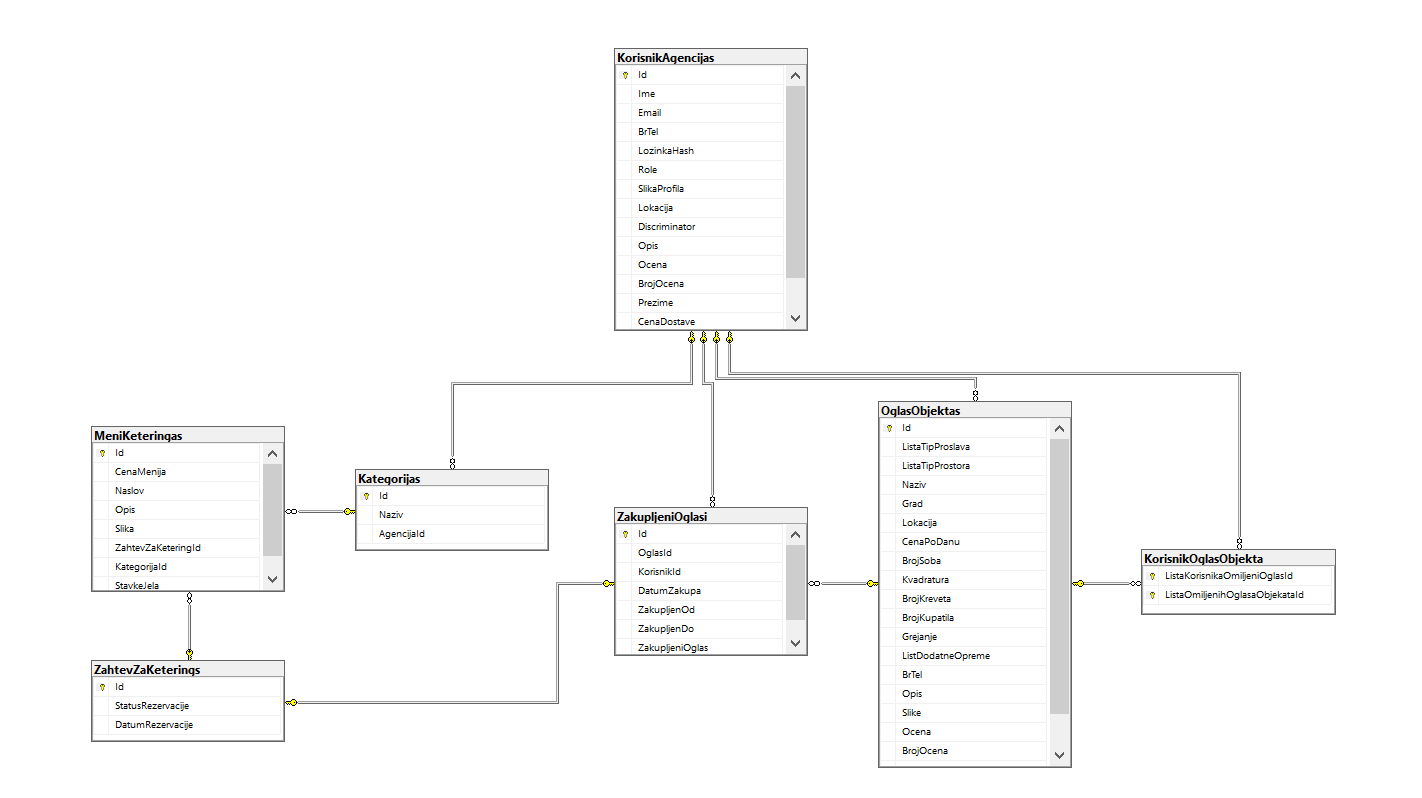
Model domena za koji se web aplikacija ZurkaNaKlik projektuje je ilustrovan UML dijagramom klasa. U njemu su prikazane domenske klase, neki od njihovih atributa, kao i veze koje se mogu identifikovati između njih.

Model domena predstavlja osnovu za projektovanje baze podataka, ali i identifikaciju nekih od komponenti (C# skripti) 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.



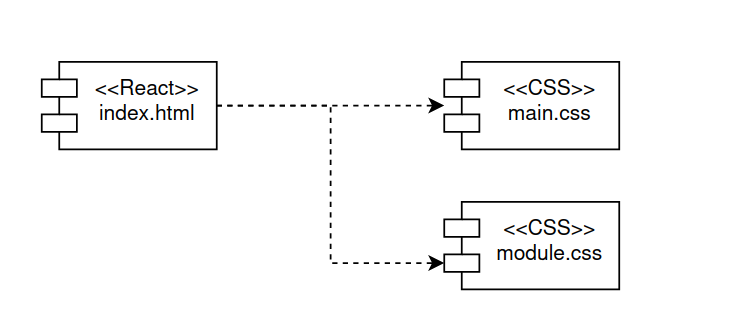
## Komponente sistema

*NAPOMENA: Web aplikacija ZurkaNaKlik projekat 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 web aplikacije ZurkaNaKlika su ASP .NET skripti čiji će pregled biti dat po arhitekturnim slojevima. Za ilustraciju će biti korišćeni UML dijagrami komponenti, ali i dijagrami klasa. U slučajevima gde je ASP .NET 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 dvema komponentama:

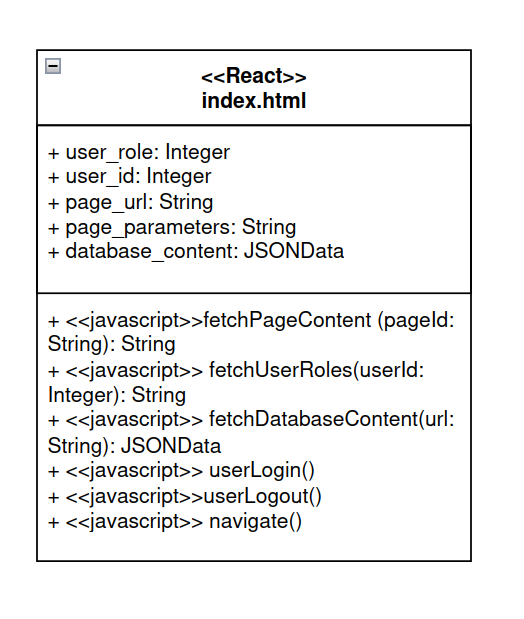


Komponenta **index.html** je implementira glavnu stranicu portala koja se prikazuje klijentu, sadržaj ove komponente varira od stranice na kojoj se nalazi klijent, kao i od privilegija korisnika, i parametara koji joj se proslede u zahtevu.

Komponenta **main.css** predstavlja opis stilova, i css promenjivih za HTML elemente koji se javljaju na različitim stranicama. Koristi se za elemente koji se nalaze na većem broju stranica.

Komponenta **modules.css** sadrži opis stilova za pojedine elemente, i sadrže samo stilove za dati element. Svaki od elemenata, ukoliko poseduje definiciju stilova, sadrži zaseban css modul koji važi samo za taj individualni element.

Parametri koji utiču na prikaz stranice ilustrovani su sledećim dijagramom klasa:

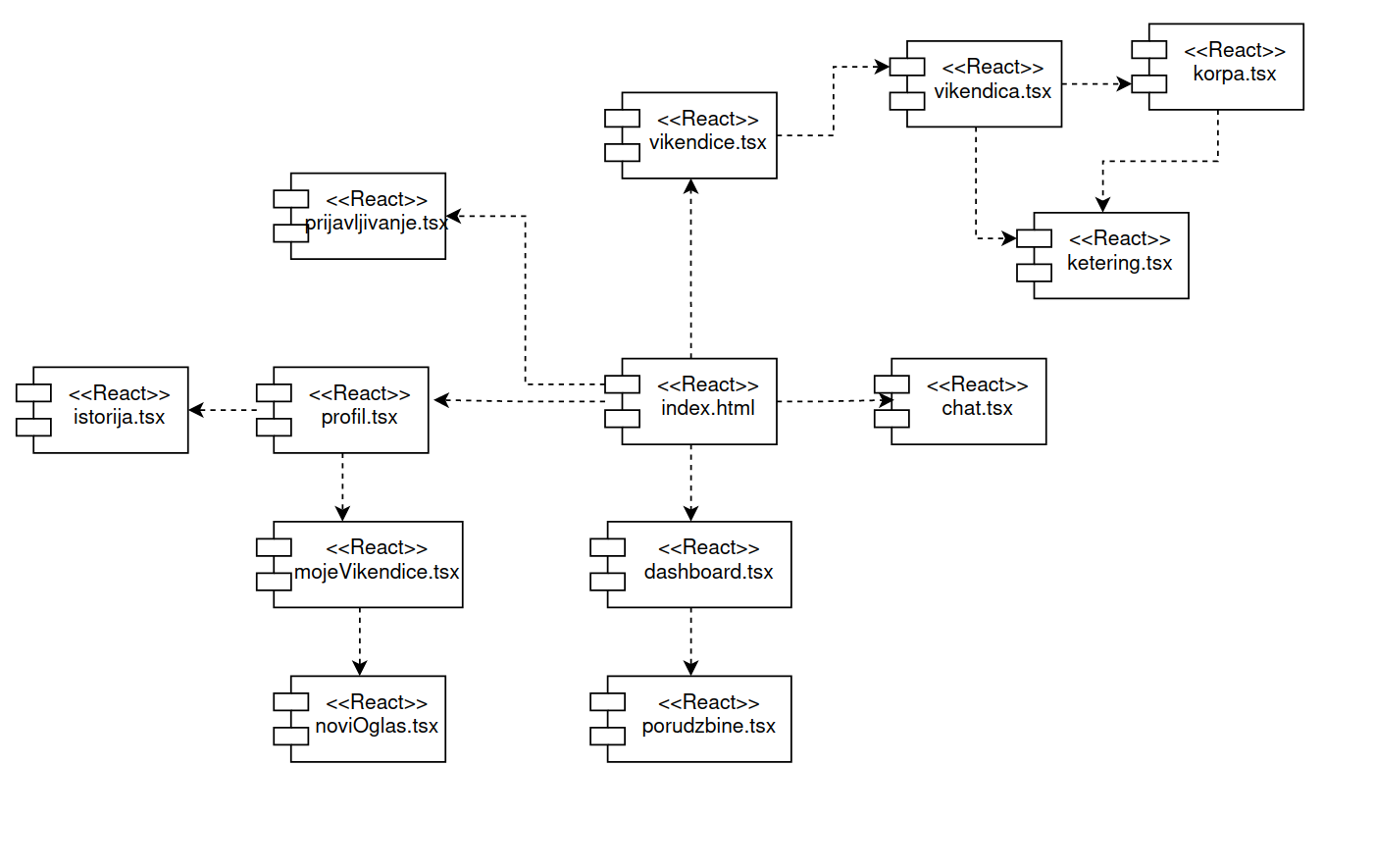


Značenje atributa je sledeće:

* user\_role – prava pristupa korisnika, za pristup nekim stranicama ili podacima je potrebno imati odgovarajuće privilegije
* user\_id – identifikator korisnika, pomoću njega znamo o kom korisniku je reč
* page\_url - interni identifikator stranice (o laboratoriji, članovi, publikacije, projekti)
* page\_parameters – dodatni parametri stranice
* database\_content – sadržaj baze podataka za elemente vidljive na datoj stranici

### 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, i dodaju svoje stilove definisane u datom css modulu definisanom za datu komponentu. Na sledećem dijagramu su prikazane glavne komponente ovog sloja i njihove međusobne zavisnosti:



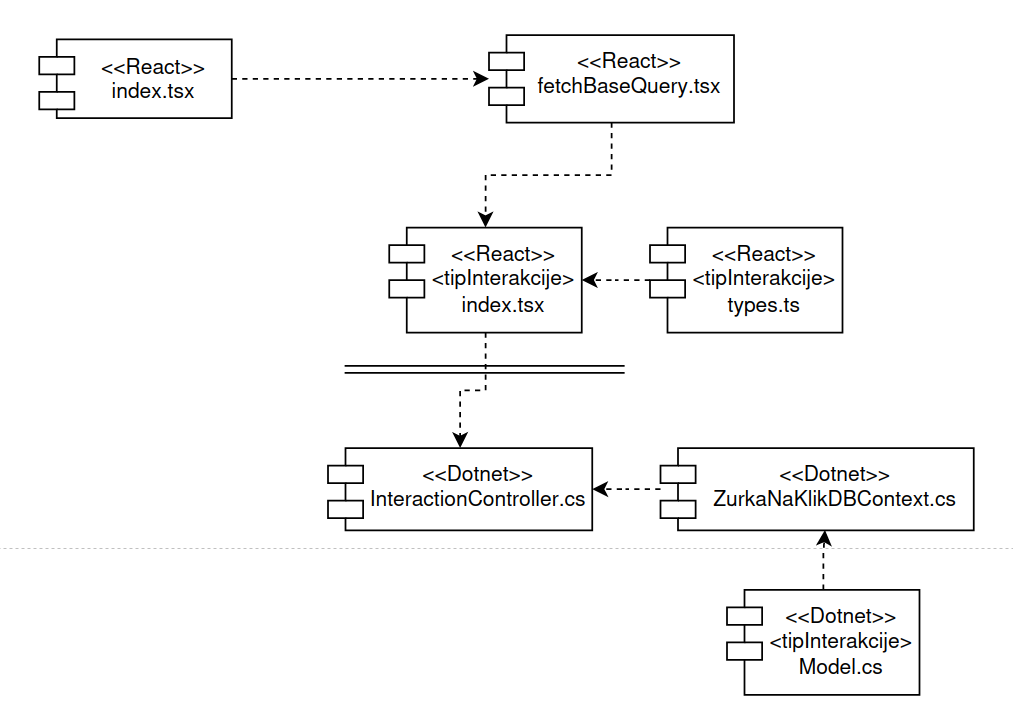
... opis svih komponenti sa dijagrama sa detaljnim dijagramima po potrebi.

### Komponente za pristup podacima

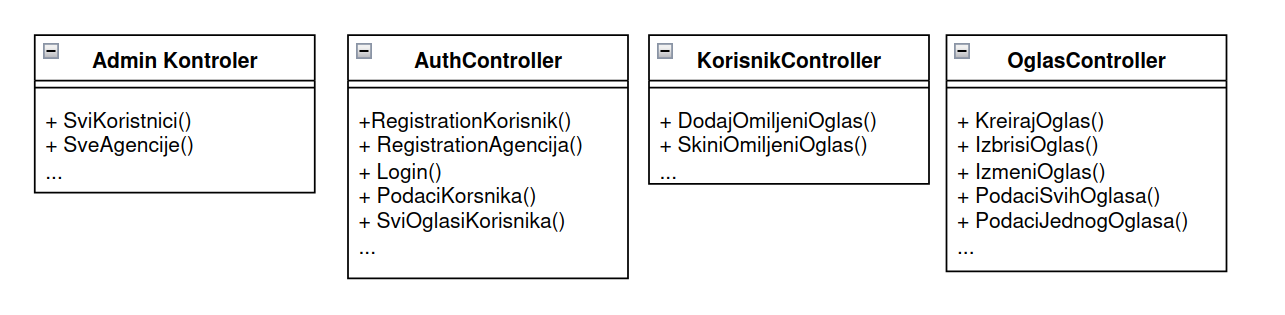
Za pristup bazi podataka je zaduzan API, koji ima direktan pristup bazi, i omogucuje da pregledač pristupi podacima koji su mu potrebni obraćanjem serveru za datu rutu. Na serveru se pristup određenim podacima

definiše unutar kontrolera za dati tip interakcije. Svaki tip interakcije ima svoj kontroler, i unutar tog kontrolera sadrži sve potrebne metode za pristup podacima. Kontroleri uzimaju svoj kontext iz glavnog konteksta, a koji sadrži modele podataka gde je svaki model vezan za dati tip interakcije. Model nam govori definiciju tabele unutar baze podataka.

Pristup podacima sa fronta je omogućen korišćenjem RTK Query biblioteke, sastoji se iz baseQuery definisane u index.tsx funkcije koja sadrži osnovne postavke za pristup serveru, a ostatak funkcionalnosti se nalazi u folderu api, koji sadrzi podfoldere za svaki tip interakcije, u svakom pod podfoldera imamo definisane tipove podataka koje se koriste u ovoj interakciji, i imamo index.tsx koji sadrži sve metode koje se koriste za uzimanje ili izmenu podataka u bazi podataka a koji su vezani za dati tip interakcije. Svi preuzeti podaci se čuvaju u store-u, keširani su sve dok ne budu invalidirani. U komponentama sistema je moguće importovati pomenuti index.tsx fajl za potrebni tip interakcije i pozvati potrebnu metodu za pristup baze koja je definisana u datom fajlu. Sve komponente imaju pristup pomenutom kešu, što smanjuje potrebu za ponovnom pristupu bazi, ukoliko to nije neophodno i podaci koje imamo su sveži.



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



Navedene funkcije obavljaju sledeće zadatke:

* SviKorisnici – Vraća listu svih korisnika
* SveAgencije – Vraća listu svih agencija
* RegistrationKorisnik – Registracija za korisnika
* RegistrationAgencija – Registracija za agenciju
* Login – Kombinovan login za agenciju i za korisnika
* PodaciKorisnika – Vraća sve podatke o datom korisniku
* SviOglasiKorisnika – Vraća sve oglase koje je dati korisnik postavio
* DodajOmiljeniOglas – Dodaje omiljeni oglas korisniku
* SkiniOmiljeniOglas – Briše omiljeni oglas korisniku
* KreirajOglas – Kreira novi oglas
* IzbrisiOglas – Briše dati oglas
* IzmeniOglas - Menja podatke o datom oglasu
* PodaciSvihOglasa – Vraća listu sa podacima o svakom oglasu iz baze (paginirano)
* PodaciJednogOglasa – Vraća sve podatke koje sadrži dati oglas
* ... – ostale funkcije

# 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. ZurkaNaKlik portal će biti dostupan 24 časa dnevno, 7 dana u nedelji. Vreme kada portal nije dostupan ne sme da pređe 1%.
2. Srednje vreme između dva sukcesivna otkaza ne sme da padne ispod 1000 sati.