CShark

Arhitekturni projekat

Verzija 1.0

Pregled izmena

|  |  |  |  |
| --- | --- | --- | --- |
| **Datum** | **Verzija** | **Opis** | **Autor** |
| 03.05.2021. | 1.0 | Inicijalna verzija | Marko, Nevena, Anita |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

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 Pregled svih zadataka 8

6.2.2 Kreiranje, ažuriranje i brisanje zadataka 8

6.2.3 Registrovanje 8

6.2.4 Pregled rang liste 8

6.2.5 Prijavljivanje 8

6.2.6 Rešavanje zadataka 8

6.2.7 Osvajanje poena 8

6.2.8 Plasiranje na rang listi 8

6.2.9 Ocenjivanje zadataka 8

6.2.10 Ažuriranje podataka 8

6.2.11 Prijavljivanje greške 8

6.2.12 Uvid u druge korisnike 8

6.2.13 Predlaganje novih zadataka 9

6.2.14 Održavanje sistema 9

6.2.15 Dodavanje i brisanje korisnika 9

6.2.16 Dodavanje privilegija 9

6.2.17 Odobravanje zadataka 9

6.2.18 Uvid u komentare i ispravljanje grešaka 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 tehnologija 10

7.1.5 CSS tehnologija 10

7.1.6 JavaScript tehnologija 10

7.1.7 C# tehnologija 11

7.1.8 SQL Server 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 C#-CGI 12

8.1.4 SQL Server 12

9. Pogled na raspoređivanje sistema 12

9.1 Klijent 12

9.2 Web server i DBMS 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 za pristup podacima 15

11. Performanse 18

12. Kvalitet 18

Arhitekturni projekat

# Cilj dokumenta

Cilj ovog dokumenta je detaljni opis arhitekture CShark Web aplikacije.

# Opseg dokumenta

Dokument se odnosi na CShark Web aplikaciju koji će biti razvijen od strane tima The Sharks. Namena aplikacije je da omogući interaktivno učenje programiranja kao i da podstakne sticanje novog znanja.

# Reference

Spisak korišćene literature:

1. CShark – Predlog projekta, D01\_Predlog\_Projekta, V1.1, 2021, The Sharks.
2. CShark – Raspored aktivnosti, D03\_Raspored\_Aktivnosti, 2021, The Sharks.
3. CShark – Plan realizacije projekta, D03\_Plan\_Realizacije,V1.1, 2021, The Sharks.
4. CShark – Vizija sistema, D02\_Vizija\_Sistema, V1.1, 2021, The Sharks.
5. CShark – Specifikacija zahteva, D04\_Spec\_Zahteva, V1.1, 2021, The Sharks.

# 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. CShark sistem će biti implementiran kao Web aplikacija zasnovana na JavaScript-u, C# framework-u .NET 5.0, i SQL
2. Klijentski deo CShark sistema će biti optimizovan za sledeće Web čitače: Edge 11, Chrome 65 i noviji, Opera 50 i noviji, kao i Firefox(Mozilla) 59 i noviji.
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].

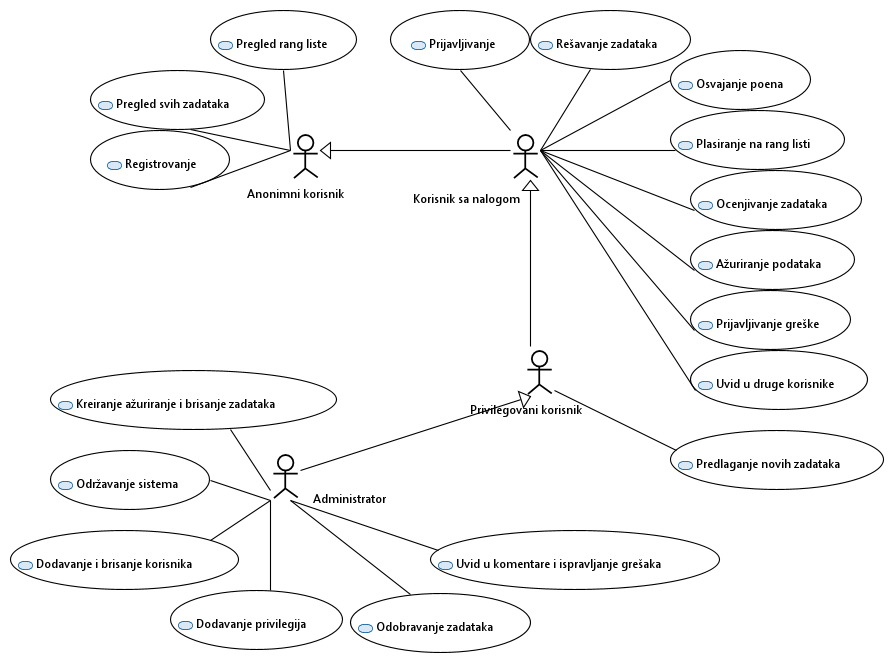
Slučajevi korišćenja CShark sistema su:

* *Pregled svih zadataka*
  + Pregled zadataka po kategorijama
  + Pregled zadataka po težini
  + Pregled zadataka po oceni
* *Kreiranje, ažuriranje i brisanje zadataka*
  + Kreiranje zadataka
    - Postavljanje tačnog rešenja
    - Dodeljivanje poena na osnovu težine
  + Uočavanje greške
    - Dobijanje prijave o grešci
  + Ažuriranje zadataka
    - Izmena u tekstu
    - Izmena rešenja
    - Izmena vrednosti poena
  + Brisanje zadataka
* Registrovanje
* Pregled rang liste
* Prijavljivanje
* Rešavanje zadataka
* Osvajanje poena
* Plasiranje na rang listi
* Ocenjivanje zadataka
* Ažuriranje podataka
* Prijavljivanje greške
* Uvid u druge korisnike
* Predlaganje novih zadataka
* Održavanje sistema
* Dodavanje i brisanje korisnika
* Dodavanje privilegija
* Odobravanje zadataka
* Uvid u komentare i ispravljanje grešaka

Ove slučajevi korišćenja mogu da iniciraju anonimni korisnik, korisnik sa nalogom, privilegovani korisnik i administrator.

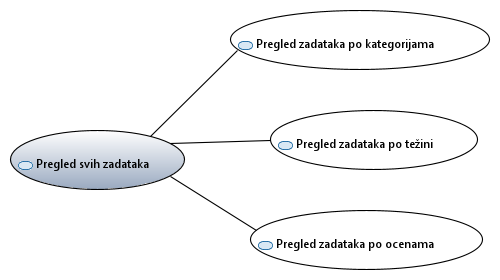
## Dijagrami slučajeva korišćenja

Osnovni UML dijagram koji prikazuje korisnike i slučajeve korišćenja CShark Web aplikacije prikazan je na sledećoj slici:

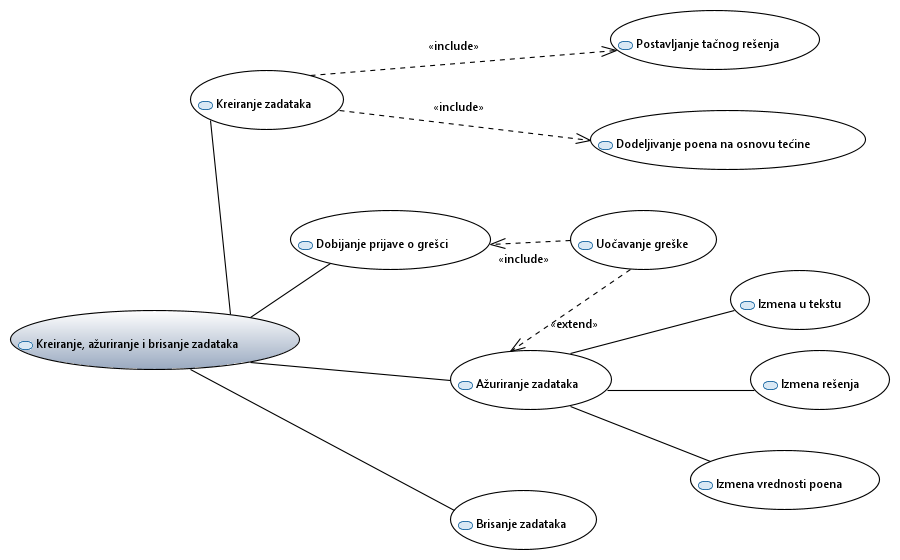


Slučajevi korišćenja *pregled svih zadataka*  i *kreiranje, ažuriranje i brisanje zadataka* obuhvataju složenije radnje koje se mogu razložiti dalje razložiti na pojedinačne slučajeve korišćenja.

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



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



## Kratak opis slučajeva korišćenja

### Pregled svih zadataka

Kratak opis: Korisnik ima uvid u sve objavljene zadatke, na osnovu kategorija, težine i poena..

Akteri koji iniciraju slučaj korišćenja: Korisnik sa nalogom, korisnik sa privilegijama, administrator.

### Kreiranje, ažuriranje i brisanje zadataka

Kratak opis: Administrator kreira nov zadatak koji mogu rešavati korisnici, izvršava izmenu zadataka i briše kreirane zadatke.

Akteri koji iniciraju slučaj korišćenja: Administrator.

### Registrovanje

Kratak opis: Registrovanje na bazu podataka.

Akteri koji iniciraju slučaj korišćenja: Anonimni korinsik.

### Pregled rang liste

Kratak opis: Korisnik ima uvid u mesečnu rang listu, kao i kompletnu rang listu koja se redovno ažurira.

Akteri koji iniciraju slučaj korišćenja: Anonimni korisnik, korisnik sa nalogom, korisnik sa privilegijama, administrator.

### Prijavljivanje

Kratak opis: Korisnik se prijavljuje na sistem.

Akteri koji iniciraju slučaj korišćenja: Korisnik sa nalogom, korisnik sa privilegijama, administrator.

### Rešavanje zadataka

Kratak opis: Korisnik rešava zadatke u kategoriji po izboru ili nekategorizovane.

Akteri koji iniciraju slučaj korišćenja: Korisnik sa nalogom, korisnik sa privilegijama.

### Osvajanje poena

Kratak opis: Korisnik osvaja poene na osnovu izabranog rešenja prilikom rešavanja zadatka.

Akteri koji iniciraju slučaj korišćenja: Korisnik sa nalogom, korisnik sa privilegijama.

### Plasiranje na rang listi

Kratak opis: Korisnik ima mogućnost ocenjivanja zadataka na osnovu težine, tj. davanja procene o odnosu između težine zadatka i broja poena koje isti nosi.

Akteri koji iniciraju slučaj korišćenja: Korisnik sa nalogom, korisnik sa privilegijama.

### Ocenjivanje zadataka

Kratak opis: Prikaz stranice sa informacijama o određenom projektu.

Akteri koji iniciraju slučaj korišćenja: Korisnik sa nalogom, korisnik sa privilegijama.

### Ažuriranje podataka

Kratak opis: Korisnik može menjati podatke na svom nalogu.

Akteri koji iniciraju slučaj korišćenja: Korisnik sa nalogom, korisnik sa privilegijama, administrator.

### Prijavljivanje greške

Kratak opis: Korisnik prijavljuje grešku u okviru nekog zadatka ako je uoči.

Akteri koji iniciraju slučaj korišćenja: Korisnik sa nalogom, korisnik sa privilegijama, administrator.

### Uvid u druge korisnike

Kratak opis: Korisnik ima uvid u informaciju drugih korisnika.

Akteri koji iniciraju slučaj korišćenja: Korisnik sa nalogom, korisnik sa privilegijama, administrator.

### Predlaganje novih zadataka

Kratak opis: Nakon sticanja određenog broja poena korisnik dobija privilegiju predlaganja novih zadataka.

Akteri koji iniciraju slučaj korišćenja: Korisnik sa privilegijama, administrator.

### Održavanje sistema

Kratak opis: Administrator održava sistem i ispravlja sve bug-ove koji su se javili prilikom korišćenja sistema.

Akteri koji iniciraju slučaj korišćenja: Administrator.

### Dodavanje i brisanje korisnika

Kratak opis: Administrator dodaje i briše postojeće korisnike ako je neophodno.

Akteri koji iniciraju slučaj korišćenja: Administrator.

### Dodavanje privilegija

Kratak opis: Administrator dodaje privilegije korisnicima ako su osvojili dovoljan broj poena

Akteri koji iniciraju slučaj korišćenja: Administrator.

### Odobravanje zadataka

Kratak opis: Na osnovu pristiglih predloga zadataka, administrator procenjuje i odobrava određene predloge..

Akteri koji iniciraju slučaj korišćenja: Administrator.

### Uvid u komentare i ispravljanje grešaka

Kratak opis: Na osnovu pristiglih prijava grešaka kod zadataka, administrator procenjuje i ispravlja greške ukoliko postoje.

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.

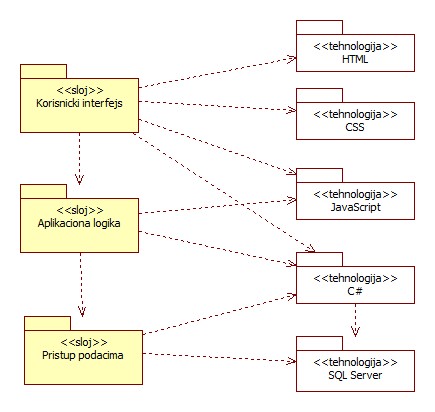
Logički pogled na CShark sistem obuhvata 3 glavna paketa: Korisnički interfejs, Aplikaciona logika, Pristup podacima.

Paket *Korisnički interfejs* sadrži HTML, CSS, JS skripte, C# 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 JS i C# skripte zadužene za realizaciju funkcionalnosti specifičnih za domen sistema koji se razvija.

Paket *Pristup podacima* sadrži C# 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, CSS, JS i C# skripte koje generišu HTML stranice preko kojih korisnici komuniciraju sa sistemom.

Sloj korisničkog interfejsa zavisi od sloja aplikacione logike, pa i paketa HTML, CSS, JS i C#.

### Aplikaciona logika sloj

Sloj aplikacione logike je srednji sloj u troslojnoj arhitekturi CShark sistema. Sadrži C# 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 JS i C# paketa.

### Pristup podacima sloj

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

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

### CSS tehnologija

Tehnologija CSS definiše izgled stranice, tj. kako su podaci prikazani u Web čitaču. CSS tehnologijom se definiše izgled gradivnih elementata HTML-a.

### JavaScript tehnologija

Tehnologija JS definiše ponašanje stranice prilikom interakcije sa korisnikom. JS skripte se izvršavaju u Web čitaču. JS skripte mogu da generišu HTML kod koji realizuje korisnički interfejs.

### C# tehnologija

Tehnologija C#-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.

### SQL Server DBMS

SQL Server predstavlja sistem za upravljanje bazama podataka koji će se koristiti za realizaciju CShark sistema.

# Pogled na procese

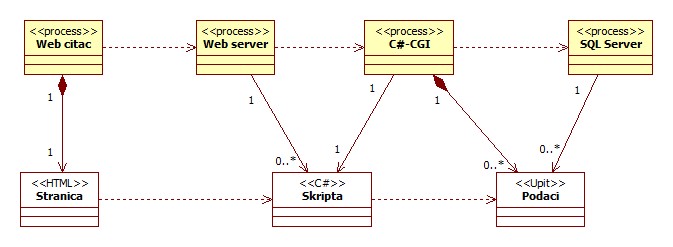
U ovom odeljku je sadržan pogled na procesnu arhitekturu sistema. Ovaj opis treba da sadrži specifikaciju različitih zadataka (procesa i niti) uključenih u rad sistema. Takođe je potrebno dati dijagrame koji pokazuju njihovu interakciju i konfiguraciju. Dodela objekata i klasa na određene zadatke takođe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

Web aplikacije zasnovane na C#-u imaju relativno jednostavan procesni model koji je u potpunosti pod kontrolom Web servera. Sa stanovišta projektanta C# 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 CShark sistema kao Web aplikacije.

## Procesi

Na sledećem UML dijagramu klasa prikazani su procesi koji učestvuju u izvršenju CShark sistema. Dijagram je opšteg tipa i može se primeniti na bilo koju Web aplikaciju zasnovanu na C#-u i SQL Server 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 C# skript, Web server inicira izvršenje C#-CGI procesa koji obrađuje odgovarajući skript i generiše sadržaj koji se vraća čitaču. Web server može paralelno da inicira veći broj C#-CGI procesa.

### C#-CGI

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

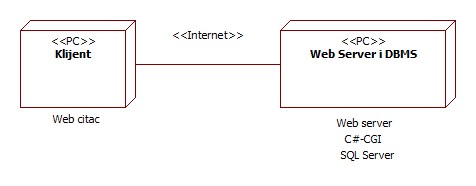
### SQL Server

SQL Server je proces koji izvršava funkcionalnost SQL Server 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 CShark Web aplikacije.



## Klijent

Pristup CShark Web aplikaciji preko klijentskih računara na kojima se izvršava Web čitač. Za povezivanje između klijenta i Web servera i DBMS-a koristi se Internet infrastruktura tako da nema ograničenja u pogledu lokacije klijenata.

## Web server i DBMS

Računar na kome se izvršava Web server i DBMS opslužuje više klijenata koji pristupaju preko Interneta. Pored osnovnog procesa koji realizuje funkcionalnost Web servera, na ovom računaru mogu da se izvršavaju i procesi C#-CGI koji vrše obradu zadatih C# skripti. DBMS se ne izvršava na posebnoj mašini, već na onoj na kojoj se izvršava Web server. Na DBMS-u se izvršava SQL Server proces koji realizuje funkcionalnost sistema za upravljanje bazama podataka.

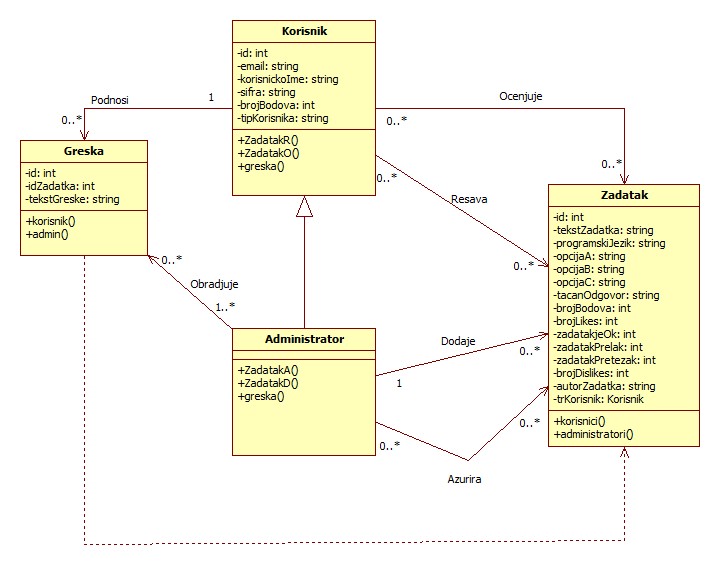
# Pogled na implementaciju sistema

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

## Model domena

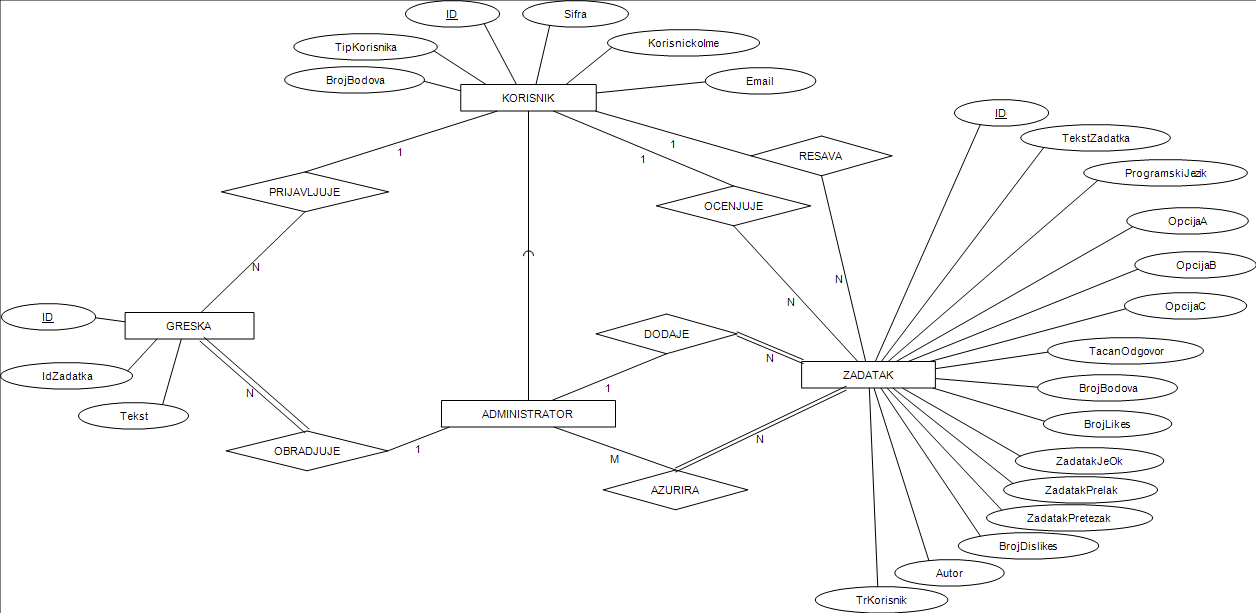
Model domena za koji se CShark portal projektuje je ilustrovan UML dijagramom klasa. U njemu su prikazane domenske klase, neki od njihovih atributa, kao i veze koje se mogu identifikovati između njih.

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 je kreirana korišćenjem *SQL Server-a* korišćenjem migracija i kontrolera u .NET, dok je dijagram kreiran korišćenjem #EER.

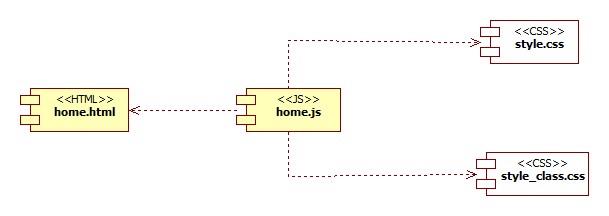


## Komponente sistema

Komponente sistema CShark sistema su C# 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 C# 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:



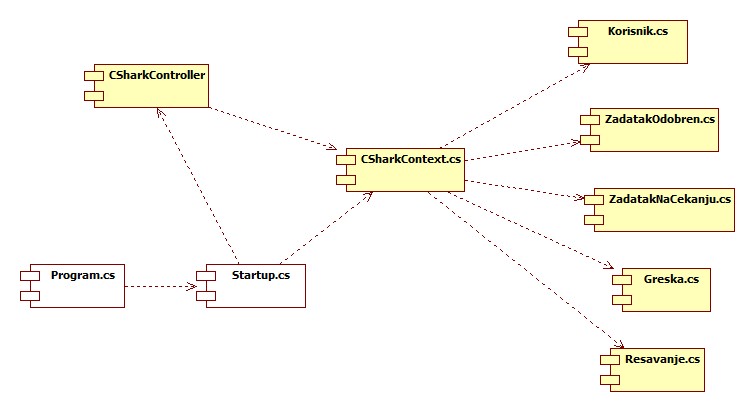
Komponenta **home.html** implementira stranicu sistema čiji sadržaj predstavlja kompletni korisnički interfejs sistema CShark.

Komponenta **home.js** implementira kompletnu dinamičku funkcionalnost sistema.

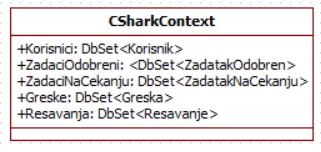
Komponenta **style.css** predstavlja opis stilova za pojedine HTML elemente koji se javljaju na različitim stranicama.

Komponenta **style\_class.css** sadrži preklapanja pojedinih stilova u komponenti **style.css** koja utiču na način prikaza JavaScript elemenata.

### Komponente za pristup podacima

Pristup bazi podataka je u potpunosti zatvoren u funkcije koje su definisane u okviru PHP skripti. Pomenute skripte se nije potrebno uključivati u fajlove, već su dostupne svim komponentama preko unapred definisanih ruta. Za konekciju sa bazom podataka nije potrebno pisati funkciju, već se to obavlja promenom odgovarajućih parametara u konfiguracionom fajlu projekta.

Na sledećem UML dijagramu klasa pobrojani su atributi klase CSharkContext iz odgovarajućeg cs fajla za pristup podacima iz baze:



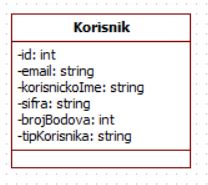
Na sledećem UML dijagramu klasa pobrojani su atributi klase ZadatakOdobren iz odgovarajućeg cs fajla za pristup podacima iz baze:



Na sledećem UML dijagramu klasa pobrojani su atributi klase ZadatakNaCekanju iz odgovarajućeg cs fajla za pristup podacima iz baze:



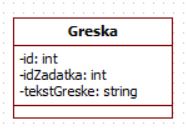
Na sledećem UML dijagramu klasa pobrojani su atributi klase Korisnik iz odgovarajućeg cs fajla za pristup podacima iz baze:



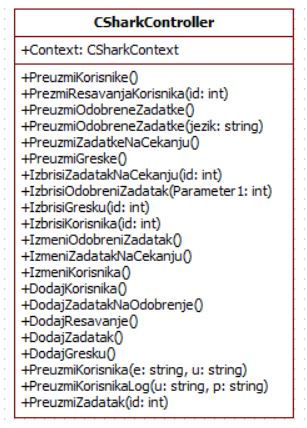
Na sledećem UML dijagramu klasa pobrojani su atributi klase Resavanje iz odgovarajućeg cs fajla za pristup podacima iz baze:



Na sledećem UML dijagramu klasa pobrojani su atributi klase Greska iz odgovarajućeg cs fajla za pristup podacima iz baze:



Na sledećem UML dijagramu klasa pobrojane su funkcije klase CSharkController iz odgovarajućeg cs fajla za pristup podacima iz baze:



# 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. CShark sistem će biti dostupan 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.