PetPlanet

# Platforma namenjena pet salonima

Arhitekturni projekat

Verzija 1.0

Pregled izmena

|  |  |  |  |
| --- | --- | --- | --- |
| **Datum** | **Verzija** | **Opis** | **Autor** |
| 15.4.2023. | 1.0 | Inicijalna verzija | Aleksa, Danilo, Jovana |

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 10

6.2.1 Registracija 10

6.2.2 Prijavljivanje 10

6.2.3 Pregled osnovnih informacija o salonima 10

6.2.4 Pregled proizvoda 10

6.2.5 Pregled informacija o uslugama salona 10

6.2.6 Pregled ocena i recenzija 10

6.2.7 Pregled odeljka za pitanja u salonima 10

6.2.8 Izmena lozinke 10

6.2.9 Izmena informacija na profilu 10

6.2.10 Dodaj proizvod 10

6.2.11 Obriši proizvod 10

6.2.12 Izmeni proizvod 10

6.2.13 Dodaj uslugu 10

6.2.14 Obriši uslugu 11

6.2.15 Izmeni uslugu 11

6.2.16 Obrada zahteva za uslugom 11

6.2.17 Obrada narudžbine 11

6.2.18 Odgovaranje na pitanja 11

6.2.19 Dodaj proizvod u korpu 11

6.2.20 Izbaci proizvod iz korpe 11

6.2.21 Online poručivanje 11

6.2.22 Pošalji zahtev 11

6.2.23 Obriši zahtev 11

6.2.24 Postavljanje pitanja 11

6.2.25 Ocenjivanje salona 11

6.2.26 Obriši klijenta 12

6.2.27 Obriši salon 12

6.2.28 Dodaj kategoriju 12

6.2.29 Obriši kategoriju 12

7. Pogled na logičku arhitekturu sistema 12

7.1 Pregled arhitekture – organizacija paketa i podsistema u slojeve 13

7.1.1 Korisnički interfejs 13

7.1.2 Aplikaciona logika 13

7.1.3 Pristup podacima 13

7.1.4 HTML 13

7.1.5 .NET 13

7.1.6 CSS 14

7.1.7 Bootstrap 14

7.1.8 ReactJS 14

7.1.9 Microsoft SQL 14

8. Pogled na procese 14

8.1 Procesi 14

8.1.1 Web čitač 15

8.1.2 Web server 15

8.1.3 ASP.NET Core 15

8.1.4 Microsoft SQL 15

9. Pogled na raspoređivanje sistema 15

9.1 Klijent 15

9.2 Web server 16

9.3 DBMS server 16

10. Pogled na implementaciju sistema 16

10.1 Model domena 16

10.2 Šema baze podataka 18

10.3 Komponente sistema 18

10.3.1 Komponente korisničkog interfejsa 18

10.3.2 Komponente aplikacione logike 19

10.3.3 Komponente za pristup podacima 19

11. Performanse 20

12. Kvalitet 20

Arhitekturni projekat

# Cilj dokumenta

Cilj ovog dokumenta je detaljni opis arhitekture Web aplikacije **PetPlanet**.

# Opseg dokumenta

Dokument se odnosi na Web aplikaciju **PetPlanet**, koja će biti razvijena od strane **AJDTeam-a**. **PetPlanet** je zapravo platforma koja će okupiti vlasnike salona kućnih ljubimaca na jednom mestu i omogućiti im da podele svoje podatke, proizvode i usluge koje nude, dok će sa druge strane vlasnicima ljubamca omogućiti da na jednom mestu istraže sve dostupne opcije za svoje ljubimce, kao i da zakažu neku od usluga koje su dostupne u salonima.

# Reference

Spisak korišćene literature:

1. PetPlanet – Predlog projekta, AJD-PetPlanet-01, V1.0, 2023, AJDTeam.
2. PetPlanet – Planirani raspored aktivnosti na projektu, V1.0, 2023, AJDTeam.
3. PetPlanet – Plan realizacije projekta, V1.0, 2023, AJDTeam.
4. PetPlanet – Vizija sistema, V1.0, 2023, AJDTeam.
5. PetPlanet – Specifikacija zahteva, V1.0, 2023, AJDTeam.

# 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. **PetPlanet** platforma će biti implementirana kao Web aplikacija zasnovana na .NET, ReactJS i Microsoft SQL Server bazi podataka.
2. Klijentski deo aplikacije **PetPlanet** će biti optimizovan za sledeće Web čitače: Internet Explorer, Opera, FireFox (Mozzila) i Google Chrome.
3. Svi zahtevi u pogledu performansi dati u [5] moraju biti uzeti u obzir pri izboru arhitekture i razvoju sistema.

# Pogled na slučajeve korišćenja

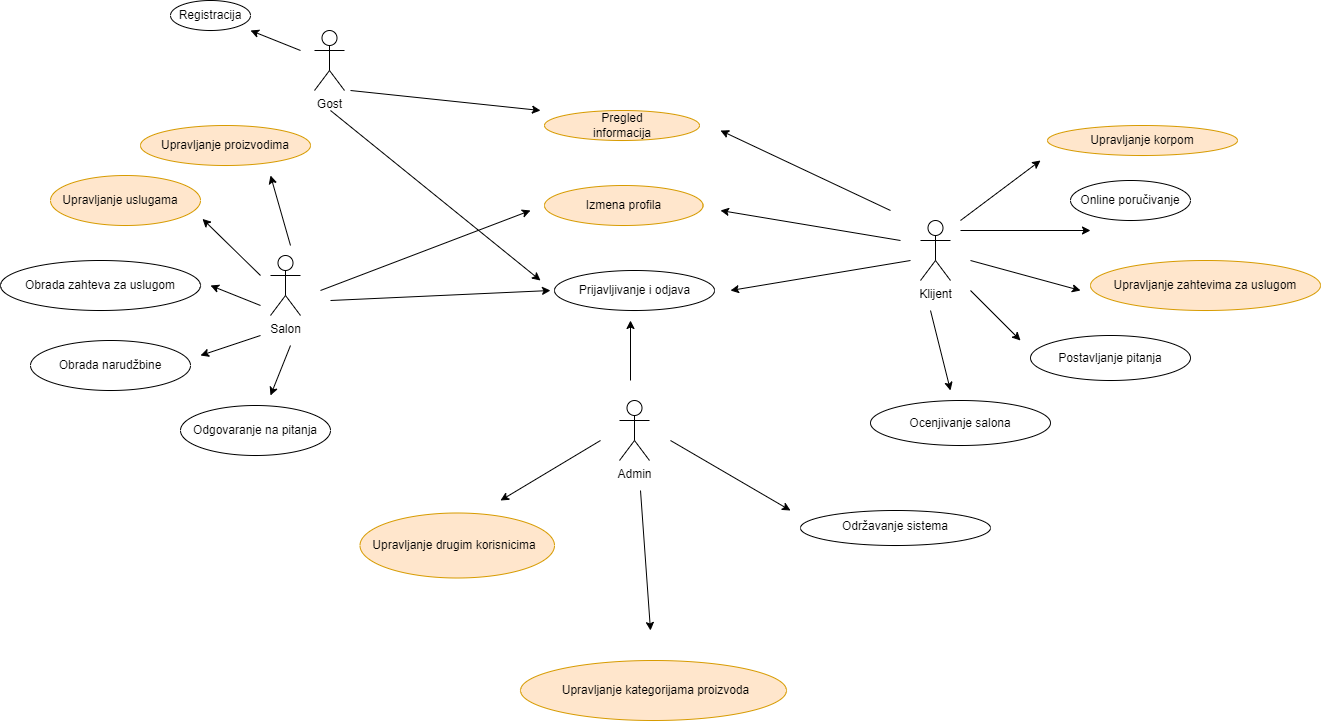
U ovom odeljku je dat pogled na slučajeve korišćenja definisane u specifikaciji zahteva [5].

Slučajevi korišćenja aplikacije **PetPlanet** portala su:

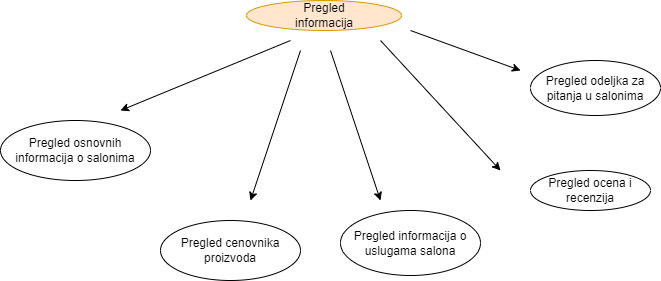
* Registracija
* Prijavljivanje
* *Pregled informacija*
  + Pregled osnovnih informacija o salonima
  + Pregled proizvoda
  + Pregled usluga
  + Pregled ocena i recenzija
  + Pregled odeljka za pitanja u salonima
* *Izmena profila*
  + Izmena lozinke
  + Izmena informacija na profilu
* *Upravljanje proizvodima*
  + Dodaj proizvod
  + Obriši proizvod
  + Izmeni proizvod
* *Upravljanje uslugama*
  + Dodaj uslugu
  + Obriši uslugu
  + Izmeni uslugu
* Obrada zahteva za uslugom
* Obrada narudžbine
* Odgovaranje na pitanja
* *Upravljanje korpom*
  + Dodaj proizvod u korpu
  + Izbaci proizvod iz korpe
* Online poručivanje
* *Upravljanje zahtevima za uslugom*
  + Pošalji zahtev
  + Obriši zahtev
* Postavljanje pitanja
* Ocenjivanje salona
* *Upravljanje drugim korisnicima*
  + Obriši klijenta
  + Obriši salon
* *Upravljanje kategorijama proizvoda*
  + Dodaj kategoriju
  + Obriši kategoriju

Ove slučajevi korišćenja mogu da iniciraju gost, klijent, salon ili administrator.

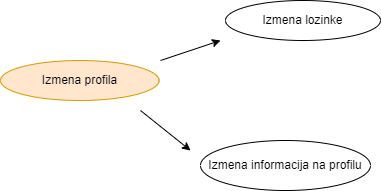
## Dijagrami slučajeva korišćenja

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

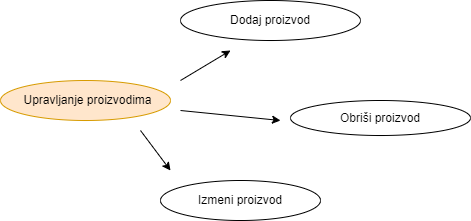
Slučajevi korišćenja *pregled informacija*, *izmena profila, upravljanje proizvodima, upravljanje uslugama*, *upravljanje korpom, upravljanje zahtevima za uslugom,* *upravljanje drugim korisnicima* i *upravljanje kategorijama proizvoda*, obuhvataju složenije radnje koje se mogu razložiti dalje razložiti na pojedinačne slučajeve korišćenja.

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

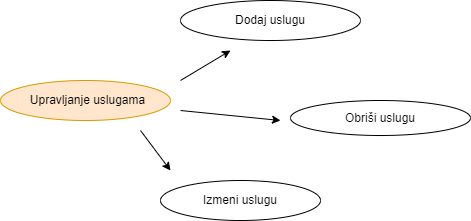
Detaljni UML dijagram za slučaj korišćenja *izmena profila* je prikazan na sledećoj slici:



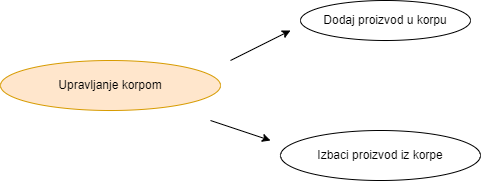
Detaljni UML dijagram za slučaj korišćenja *upravljanje proizvodima* je prikazan na sledećoj slici:



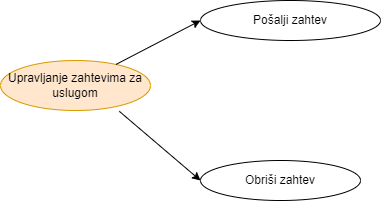
Detaljni UML dijagram za slučaj korišćenja *upravljanje uslugama* je prikazan na sledećoj slici:



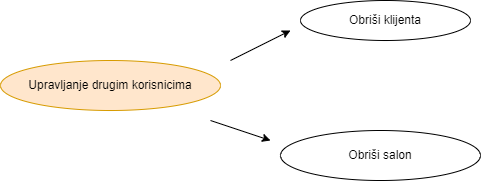
Detaljni UML dijagram za slučaj korišćenja *upravljanje korpom* je prikazan na sledećoj slici:



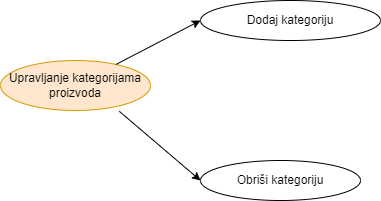
Detaljni UML dijagram za slučaj korišćenja *upravljanje zahtevima za uslugom* je prikazan na sledećoj slici:



Detaljni UML dijagram za slučaj korišćenja *upravljanje drugim korisnicima* je prikazan na sledećoj slici:



Detaljni UML dijagram za slučaj korišćenja *upravljanje kategorijama proizvoda* je prikazan na sledećoj slici:



## Kratak opis slučajeva korišćenja

### Registracija

**Kratak opis:** Kreiranje naloga novom korisniku kako bi kasnije pristupao ostalim funkcionalnostima aplikacije.

**Akteri:** Gost.

### Prijavljivanje

**Kratak opis:** Prijavljivanje korisnika na aplikaciju radi pristupanja ostalim funkcionalnostima aplikacije.

**Akteri:** Klijent, Salon, Administrator.

### Pregled osnovnih informacija o salonima

**Kratak opis:** Prikaz stranice sa informacijama o konkretnom salonu.

**Akteri:** Gost, Klijent, Salon, Administrator.

### Pregled proizvoda

**Kratak opis:** Prikaz stranice sa proizvodima koje konkretni salon nudi.

**Akteri:** Gost, Klijent.

### Pregled informacija o uslugama salona

**Kratak opis:** Prikaz stranice sa uslugama koje konkretni salon nudi.

**Akteri:** Gost, Klijent.

### Pregled ocena i recenzija

**Kratak opis:** Prikaz stranice na kojoj se nalazi ocena salona i recenzije klijenata o salonu.

**Akteri:** Gost, Klijent.

### Pregled odeljka za pitanja u salonima

**Kratak opis:** Prikaz stranice na kojoj se nalaze pitanja koja su postavljena konkretnom salonu.

**Akteri:** Gost, Klijent.

### Izmena lozinke

**Kratak opis:** Korsnik može da izmeni svoju lozinku.

**Akteri:** Klijent, Salon, Administrator.

### Izmena informacija na profilu

**Kratak opis:** Izmena ličnih podataka korisnika.

**Akteri:** Klijent, Salon, Administrator.

### Dodaj proizvod

**Kratak opis:** Dodavanje novog proizvoda koje salon nudi.

**Akteri:** Salon.

### Obriši proizvod

**Kratak opis:** Brisanje nekog od postojećih proizvoda koje salon nudi.

**Akteri:** Salon.

### Izmeni proizvod

**Kratak opis:** Izmena podataka nekog od postojećih proizvoda koje salon nudi.

**Akteri:** Salon.

### Dodaj uslugu

**Kratak opis:** Dodavanje nove usluge koje salon nudi.

**Akteri:** Salon.

### Obriši uslugu

**Kratak opis:** Brisanje neke od postojećih usluga koje salon nudi.

**Akteri:** Salon.

### Izmeni uslugu

**Kratak opis:** Izmena podataka neke od postojećih usluga koje salon nudi.

**Akteri:** Salon.

### Obrada zahteva za uslugom

**Kratak opis:** Salon bira zahtev koji želi da obradi iz liste neobrađenih zahteva.

**Akteri:** Salon.

### Obrada narudžbine

**Kratak opis:** Salon obrađuje narudžbine koje je primio od klijenata.

**Akteri:** Salon.

### Odgovaranje na pitanja

**Kratak opis:** Salon odgovara na pitanja koja su klijenti postavili.

**Akteri:** Salon.

### Dodaj proizvod u korpu

**Kratak opis:** Klijent može da doda proizvod u korpu.

**Akteri:** Klijent.

### Izbaci proizvod iz korpe

**Kratak opis:** Klijent može izmeniti sadržaj svoje korpe pre izvršenog zahteva.

**Akteri:** Klijent.

### Online poručivanje

**Kratak opis:** Klijent nakon ubacivanja proizvoda u korpu može da poruči iste.

**Akteri:** Klijent.

### Pošalji zahtev

**Kratak opis:** Klijent može da pošalje zahtev za rezervaciju odredjene usluge.

**Akteri:** Klijent.

### Obriši zahtev

**Kratak opis:** Klijent ima opciju da obriše zahtev.

**Akteri:** Klijent.

### Postavljanje pitanja

**Kratak opis:** Klijent ima opciju da postavi pitanje salonima.

**Akteri:** Klijent.

### Ocenjivanje salona

**Kratak opis:** Klijent može da oceni salon.

**Akteri:** Klijent.

### Obriši klijenta

**Kratak opis:** Administrator može da briše profile klijenata.

**Akteri:** Administrator.

### Obriši salon

**Kratak opis:** Administrator može da briše profile salona.

**Akteri:** Administrator.

### Dodaj kategoriju

**Kratak opis:** Administrator može da dodaje nove kategorije proizvoda koje saloni mogu da biraju.

**Akteri:** Administrator.

### Obriši kategoriju

**Kratak opis:** Administrator može da obriše neku od postojećih kategorija proizvoda koje saloni mogu da biraju.

**Akteri:** 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.

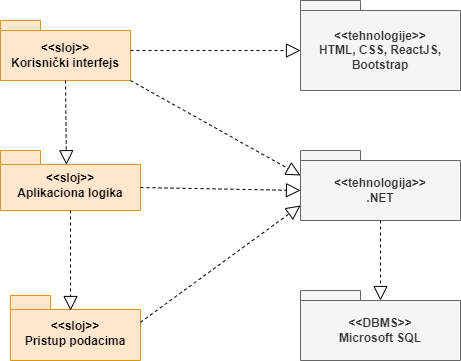
Logički pogled na **PetPlanet** aplikaciju obuhvata 3 glavna paketa: Korisnički interfejs, Aplikaciona logika, Pristup podacima.

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

Paket *Pristup podacima* pomoću ASP.NET Core framework-a za C#???? predstavlja 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 **PetPlanet** Web aplikacije. U njemu su sadržane sve HTML stranice, CSS stilove i ReactJS skripte koje generišu HTML stranice preko kojih korisnici komuniciraju sa sistemom.

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

### Aplikaciona logika

sloj

Sloj aplikacione logike je srednji sloj u troslojnoj arhitekturi **PetPlanet** Web aplikacije. Sadrži ReactJS skripte koje 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 ReactJS paketa.

### Pristup podacima

sloj

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

Ovaj sloj zavisi od aplikacionog sloja u kome se kreiraju upiti upućeni bazi 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.

### .NET

ASP.NET stranice se izvršavaju na serverskoj strani i generišu HTML, WML ili XML koji se šalje desktop ili mobil pretraživačima.ASP.NET koristi „event-driven“ model programiranja koji poboljšava performanse i omogućava separaciju korisničkog interfejsa od logike aplikacije.ASP.NET radi na vrhu HTTP protokola koristeći HTTP komande i pravila kako bi omogućio obostranu komunikaciju izmedju klijenta i servera.Kod je moguće pisati u C#, VisualBasic, Jscript ili J# jezicima.

ASP.NET omogućava 3 metode razvoja :

• Web Forms (tradicionalni ASP.NET)

• Web Pages

• MVC (Model View Controller)

### CSS

tehnologija

Dok HTML definiše gradivne elemente stranica, CSS je jezik formatiranja pomoću kog se definiše izgled elemenata Web-stranice.

### Bootstrap

tehnologija

Bootstrap je open-source JavaScript framework, odnosno kombinacija JavaScript-a, HTML-a i CSS-a razvijen sa ciljem da omogući i olakša razvoj web formi kao i razvoj naprednih web komponenti.Modularnog je tipa što omogućava njegovu lakšu nadogradnju i upotrebu sa različitim modulima koje razvijaju različiti developeri. Odlično sarađuje sa JavaScriptom i jQuery bibliotekom.

### ReactJS

Tehnologija

ReactJS je frontend framework. U pitanju je JavaScript biblioteka koja služi za kreiranje korisničkih interfejsa. Omogućava modularan pristup kreiranju komponenti, što omogućava laku organizaciju i ponovno korišćenje koda. ReactJS koristi JSX, koji omogućava pisanje koda sličnog HTML-u unutar JavaScript-a. ReactJS ne podržava ugrađenu dependency injection funkcionalnost, ali se dependency injection može postići korišćenjem drugim biblioteka i alata. ReactJS može biti fobar izbor za aplikacije koje zahtevaju brzo ažuriranje podataka i interakciju sa korisnikom, kao što su društvene mreže, e-commerce sajtovi i slično.

### Microsoft SQL

DBMS

Microsoft SQL predstavlja sistem za upravljanje bazama podataka koji će se koristiti za realizaciju **PetPlanet** aplikacije. Microsoft SQL čuva podatke kao XML *(Extensible Markup Language)* dokumente sa dinamičkim šemama. Microsoft SQL koristi koncept SQL baza podataka, koje koriste SQL za povezivanje podataka.

# Pogled na procese

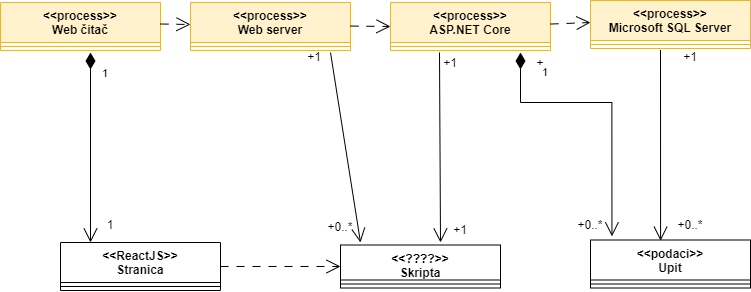
U ovom odeljku je sadržan pogled na procesnu arhitekturu sistema. Ovaj opis treba da sadrži specifikaciju različitih zadataka (procesa i niti) uključenih u rad sistema. Takođe je potrebno dati dijagrame koji pokazuju njihovu interakciju i konfiguraciju. Dodela objekata i klasa na određene zadatke takođe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

Web aplikacije zasnovane na PHP-u imaju relativno jednostavan procesni model koji je u potpunosti pod kontrolom Web servera. Sa stanovišta projektanta PHP 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 PeNcIL portala kao Web aplikacije.

## Procesi

Na sledećem UML dijagramu klasa prikazani su procesi koji učestvuju u izvršenju **PetPlanet** aplikacije.



### 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. Obradu tih zahteva vrši ASP.NET Core proces i rezultat obrade je je C# skript. Web server na osnovu sadržaja C# skripte formira prikaz i prikazuje ga klijentu u Web čitaču. Web server može paralelno da inicira veći broj ASP.NET Core procesa.

### ASP.NET Core

ASP.NET Core obavlja proces obrade C# skripti. Kao rezultat se generišse odgovarajući tekstualni sadržaj koji Web čitač može da prikaže. Za izvršenje C# skripti ovaj proces može da zahteva usluge Microsoft SQL Server servera. Komunikacija između ASP.NET Core procesa i SQL servera se obavlja preko prosleđivanja upita i vraćanja rezultata.

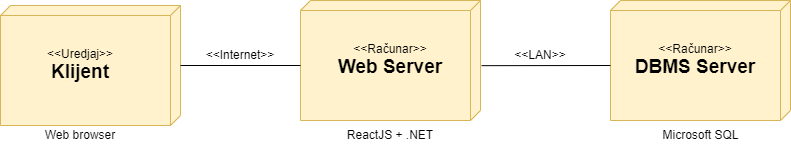
### Microsoft SQL

Microsoft SQL je proces koji izvršava funkcionalnost čuvanja i skladištenja podataka koji su neophodni za rad aplikacije.Ukoliko se javila potreba za pribavljanjem podataka onda se podaci pribavljaju posredstvom .NET procesa, takođe moguće je dodavanje novih podataka takođe kroz posredstvo .NET procesa.

# 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 **PetPlanet** aplikacije.



## Klijent

Pristup **PetPlanet** 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, na ovom računaru mogu da se izvršavaju i procesi .NET procesi koji vrše obradu zadatih ReactJS skripti. U najopštijoj 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 Microsoft 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).

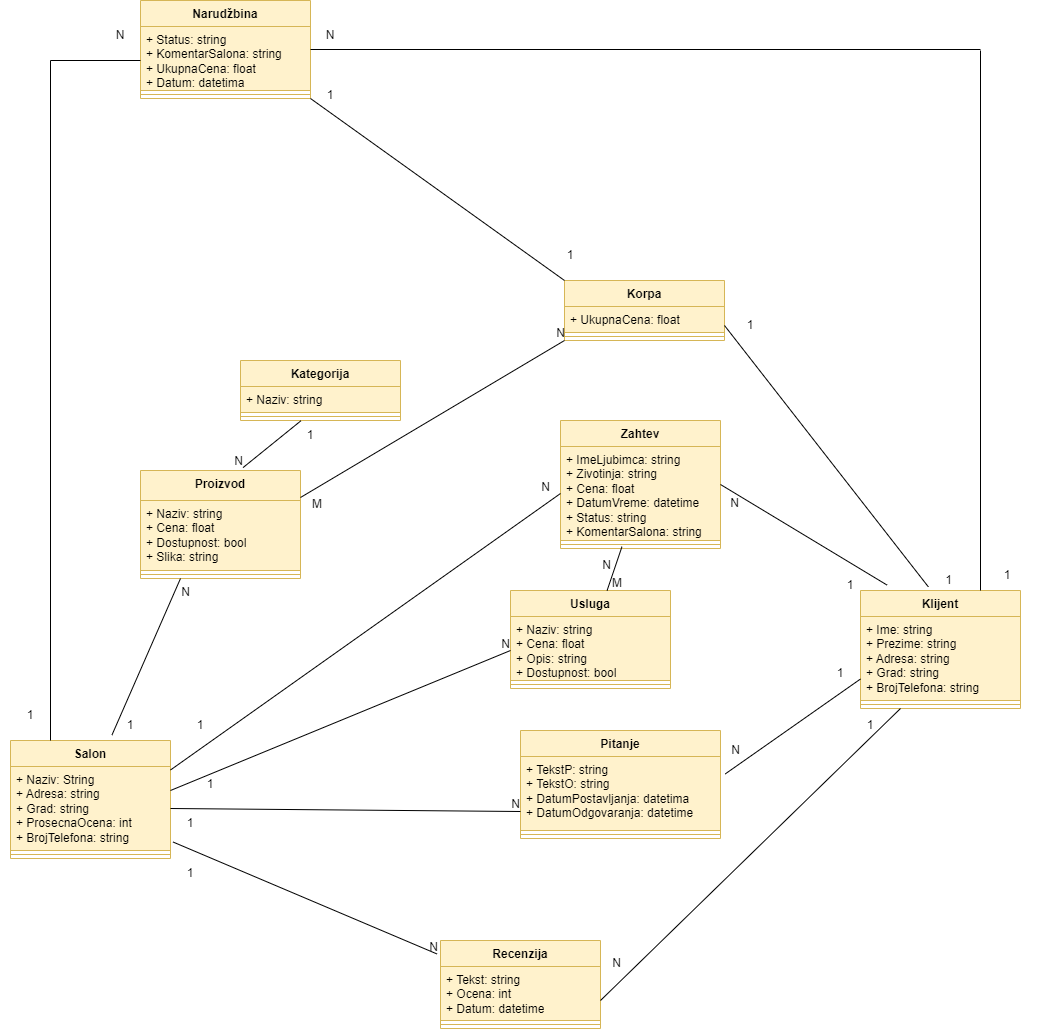
# Pogled na implementaciju sistema

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

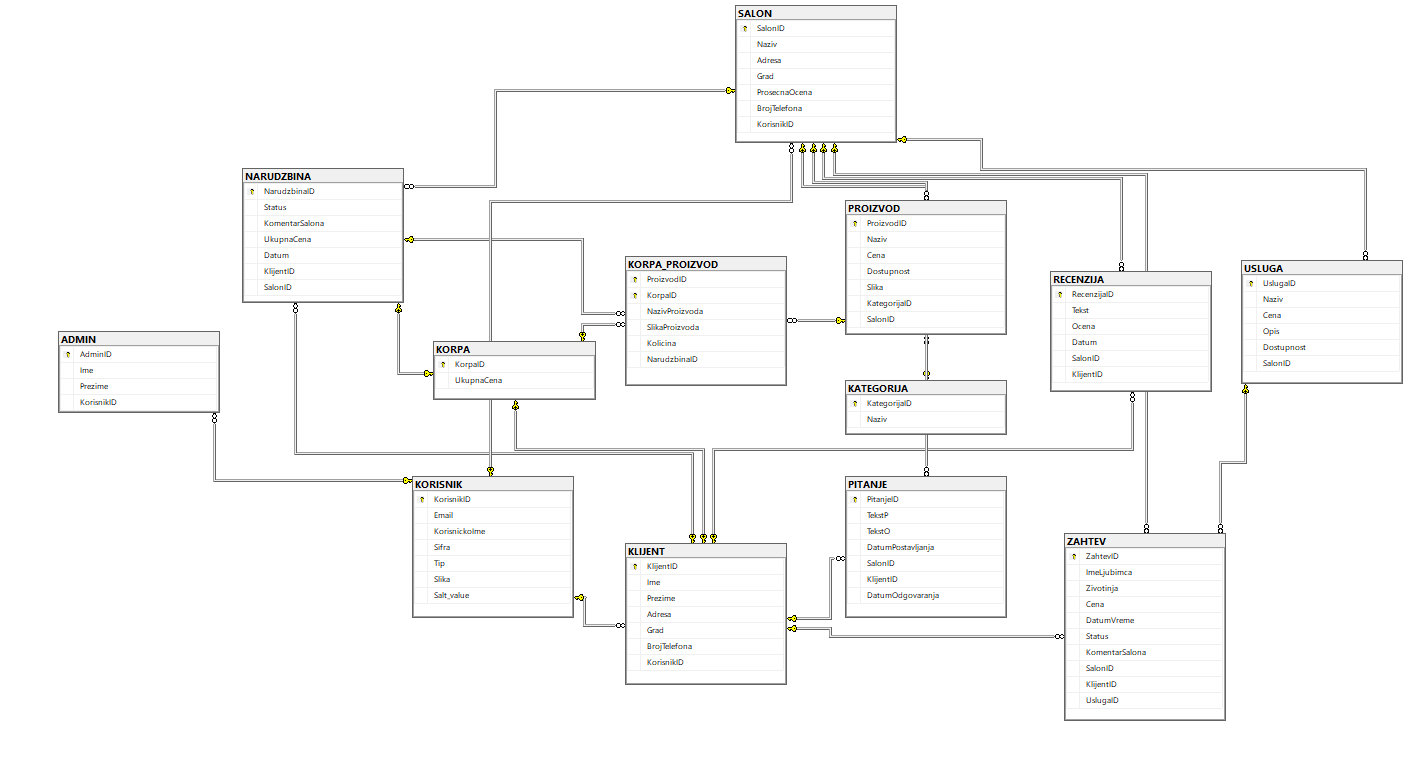
## Model domena

Model domena za koji se **PetPlanet** 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 su u planu za implementaciju.



## Šema baze podataka

Detaljna šema baze podataka je prikazana na sledećem dijagramu. Baza podataka je kreirana korišćenjem Azure Data Studio-a, a dijagram korišćenjem Microsoft SQL Management Studio-a.

## Komponente sistema

*NAPOMENA: PeNcIL 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 sistema PeNcIL portala su PHP 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 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

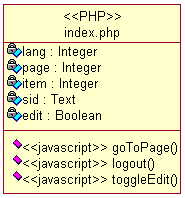
Dizajn korisničkog interfejsa je obuhvaćen dvema komponentama:



Komponenta **index.php** 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.

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

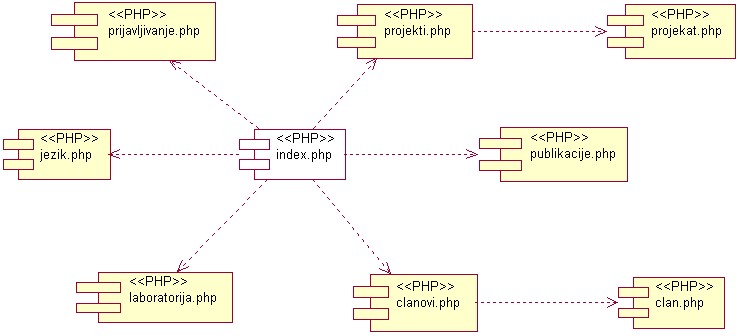


Značenje atributa je sledeće:

* lang – identifikator jezika (JID iz tabele Jezik)
* page – interni identifikator stranice (o laboratoriji, članovi, publikacije, projekti)
* item – identifikator stavke koja se detaljno prikazuje na stranici (određeni član ili projekat)
* sid – identifikator sesije kada je korisnik ulogovan
* edit – definiše da li je uključeno uređivanje

### Komponente aplikacione logike

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



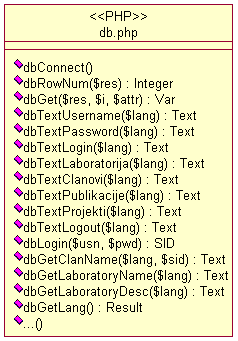
... opis svih komponenti sa dijagrama sa detaljnim dijagramima po potrebi.

### Komponente za pristup podacima

Pristup bazi podataka je u potpunosti zatvoren u funkcije koje su definisane u okviru PHP skripta **db.php**. 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:

* dbConnect – povezivanje na bazu podataka, poziva se u samom skriptu
* dbRowNum – vraća broj slogova sadržanih u rezultatu upita
* dbGet – vraća vrednost polja iz rezultata upita
* dbText... – vraća vrednost pojedinih konstanti koje se definišu uz jezik
* dbLogin – loguje korisnika i vraća SID
* dbGetClanName – vraća ime člana po jeziku i SID-u
* dbGetLaboratoryName – vraća naziv laboratorije za zadati jezik
* dbGetLaboratoryDesc – vraća opis laboratorije za zadati jezik
* dbGetLang – vraća rezultat upita nad jezicima definisanim u bazi podataka
* ... – 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. **PetPlanet** aplikacija će biti dostupna 24 časa dnevno, 7 dana u nedelji. Vreme kada portal nije dostupan ne sme da pređe 15%.
2. Srednje vreme između dva sukcesivna otkaza ne sme da padne ispod 120 sati.