BioTS

BioTracking Serbia

Arhitekturni projekat

Verzija 1.0

Pregled izmena

|  |  |  |  |
| --- | --- | --- | --- |
| **Datum** | **Verzija** | **Opis** | **Autor** |
| 22.4.2024. | 1.0 | Inicijalna verzija | Taškov Vukan |
| 25.4.2024. | 2.0 | Finalna verzija | Taškov Vukan |
| 28.4.2024. | 3.0 | Korekcije | Taškov Vukan |
|  |  |  |  |

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 Registracija 8

6.2.2 Prijavljivanje 8

6.2.3 Ažuriranje podataka o sebi 8

6.2.4 Pregled osnovnih podataka o biljci 8

6.2.5 Pretraga biljaka 8

6.2.6 Iscrtavanje područja biljke na karti 8

6.2.7 Promena područja biljke 8

6.2.8 Brisanje biljke 9

6.2.9 Pregled svih predstojećih obilazaka 9

6.2.10 Pregled osnovnih informacija o obilasku 9

6.2.11 Filtriranje podataka po datumu 9

6.2.12 Filtriranje obilazaka po lokaciji 9

6.2.13 Pregled obilazaka za koje je korisnik prijavljen 9

6.2.14 Pregled obilazaka na kojima je određeni korisnik učestvovao 9

6.2.15 Pregled podataka o organizatoru obilaska 9

6.2.16 Dodavanje novog obilaska 9

6.2.17 Otkazivanje obilaska 9

6.2.18 Promena rute obilaska 9

6.2.19 Promena datuma obilaska 9

6.2.20 Prijavljivanje korisnika za obilazak 10

6.2.21 Komentarisanje obilaska 10

6.2.22 Pregled liste korisnika sistema 10

6.2.23 Brisanje postojećeg korisnika 10

7. Pogled na logičku arhitekturu sistema 10

7.1 Pregled arhitekture – organizacija paketa i podsistema u slojeve 11

7.1.1 Korisnički interfejs 11

7.1.2 Aplikaciona logika 11

7.1.3 Pristup podacima 11

7.1.4 React.js 11

7.1.5 .NET Core 11

7.1.6 MySQL 11

8. Pogled na procese 12

8.1 Procesi 12

8.1.1 Web čitač 12

8.1.2 Web server 12

8.1.3 .NET Core 12

8.1.4 MySQL Server 12

9. Pogled na raspoređivanje sistema 13

9.1 Klijent 13

9.2 Web server 13

9.3 DBMS server 13

10. Pogled na implementaciju sistema 14

10.1 Model domena 14

10.2 Šema baze podataka 15

10.3 Komponente sistema 16

10.3.1 Komponente korisničkog interfejsa 16

10.3.2 Komponente aplikacione logike 18

10.3.3 Komponente za pristup podacima 22

11. Performanse 22

12. Kvalitet 22

Arhitekturni projekat

# Cilj dokumenta

Cilj ovog dokumenta je detaljni opis arhitekture BioTS web aplikacije.

# Opseg dokumenta

Dokument se odnosi na BioTS web aplikaciju koja će biti razvijena od strane VADevelopment tima. BioTS predstavlja skraćenicu za BioTracking Serbia. Namena sistema je efikasno upoznavanje sa prirodom Srbije, organizovanje i posećivanje obilazaka na teritoriji Srbije..

# Reference

Spisak korišćene literature:

1. BioTS – Predlog projekta, SWE-BioTS-01, V3.0, 2024, VADevelopment.
2. BioTS – Planirani raspored aktivnosti na projektu, V1.0, 2024, VADevelopment.
3. BioTS – Plan realizacije projekta, V2.0, 2024, VADevelopment.
4. BioTS – Vizija sistema, V3.1, 2024, VADevelopment.
5. BioTS – Specifikacija zahteva, 2.1, 2024, VADevelopment.

# 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. BioTS portal će biti implementiran kao Web aplikacija zasnovana na ASP NET framework, Reac biblioteci i MySQL bazi podataka [4].
2. Klijentski deo BioTS web aplikacije će biti optimizovan za sve Web čitače.
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 BioTS web aplikacije su:

* Registracija
* Prijavljivanje
* Ažuriranje podataka o sebi
* *Pregled informacija:*
  + Pregled osnovnih podataka o biljci
  + Pregled svih predstojećih obilazaka
  + Pregled osnovnih informacija o obilasku
  + Pregled obilazaka za koje je korisnik prijavljen
  + Pregled obilazaka na kojima je određeni korisnik učestvovao
  + Pregled podataka o organizatoru obilaska
  + Pregled liste korisnika sistema
* *Pretraga informacija:*
  + Pretraga biljaka
  + Filtriranje obilazaka po datumu
  + Filtriranje obilazaka po lokaciji
* *Ažuriranje podataka:*
  + Promena područja biljke
  + Brisanje biljke
  + Promena rute obilaska
  + Promena datuma obilaska
  + Brisanje postojećeg korisnika
* *Interakcija sa sistemom:*
  + Iscrtavanje područja biljke na karti
  + Dodavanje novog obilaska
  + Otkazivanje obilaska
  + Prijavljivanje korisnika za obilazak
  + Komentarisanje obilaska

Ove slučajevi korišćenja mogu da iniciraju posetilac sistema, običan korisnik, turistički vodič i administrator.

## Dijagrami slučajeva korišćenja

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

A diagram of a diagram

Description automatically generated

Slučajevi korišćenja *pregled biljaka* i *ažuriranje podataka o biljkama* i *pregled obilazaka i ažuriranje podataka o obilascima* obuhvataju složenije radnje koje se mogu razložiti dalje razložiti na pojedinačne slučajeve korišćenja.

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

A diagram of a company

Description automatically generated

Detaljni UML dijagram za slučaj korišćenja *ažuriranje podataka o biljkama* je prikazan na sledećoj slici:

A diagram of a diagram

Description automatically generated

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

A diagram of a company

Description automatically generated

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

A diagram of a company

Description automatically generated

## Kratak opis slučajeva korišćenja

### Registracija

Kratak opis: Registracija novog korisnika na sistem.

Akteri koji iniciraju slučaj korišćenja: Posetilac sistema

### Prijavljivanje

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

Akteri koji iniciraju slučaj korišćenja: Posetilac sistema, običan korisnik, turistički vodič i administrator.

### Ažuriranje podataka o sebi

Kratak opis: Ažuriranje podataka o sebi od strane prijavljenog korisnika.

Akteri koji iniciraju slučaj korišćenja: Običan korisnik, turistički vodič i administrator.

### Pregled osnovnih podataka o biljci

Kratak opis: Prikaz forme sa osnovnim podacima o biljci.

Akteri koji iniciraju slučaj korišćenja: Posetilac sistema, običan korisnik, turistički vodič i administrator.

### Pretraga biljaka

Kratak opis: Korisnik ima opciju da pretraži željene biljke na osnovu parametara koji ga zanimaju, odnosno da li je biljka lekovita, da li je trenutno njena sezona, i na osnovu cveta i stabla, ili naziva.

Akteri koji iniciraju slučaj korišćenja: Posetilac sistema, običan korisnik, turistički vodič i administrator.

### Iscrtavanje područja biljke na karti

Kratak opis: Grafički prikaz područja pronađenih biljaka nakon pretrage.

Akteri koji iniciraju slučaj korišćenja: Posetilac sistema, običan korisnik, turistički vodič i administrator.

### Promena područja biljke

Kratak opis: Izmena trenutnog područja na kojem se pronalazi određena biljka.

Akteri koji iniciraju slučaj korišćenja: Turistički vodič i administrator.

### Brisanje biljke

Kratak opis: Brisanje biljke iz sistema ukoliko nestane sa teritorije Srbije.

Akteri koji iniciraju slučaj korišćenja: Turistički vodič i administrator.

### Pregled svih predstojećih obilazaka

Kratak opis: Prikaz stranice sa svim predstojećim organizovanim obilascima.

Akteri koji iniciraju slučaj korišćenja: Običan korisnik, turistički vodič i administrator.

### Pregled osnovnih informacija o obilasku

Kratak opis: Prikaz forme sa svim informacijama o izabranom obilasku.

Akteri koji iniciraju slučaj korišćenja: Običan korisnik, turistički vodič i administrator.

### Filtriranje podataka po datumu

Kratak opis: Prikaz stranice sa obilascima razvrstanim po datumu.

Akteri koji iniciraju slučaj korišćenja: Običan korisnik, turistički vodič i administrator.

### Filtriranje obilazaka po lokaciji

Kratak opis: Prikaz stranice sa obilascima razvrstanim po lokaciji.

Akteri koji iniciraju slučaj korišćenja: Običan korisnik, turistički vodič i administrator.

### Pregled obilazaka za koje je korisnik prijavljen

Kratak opis: Prikaz svih obilazaka za koje je određeni korisnik prijavljen.

Akteri koji iniciraju slučaj korišćenja: Običan korisnik, turistički vodič i administrator.

### Pregled obilazaka na kojima je određeni korisnik učestvovao

Kratak opis: Prikaz svih obilazaka na kojima je učestvovao izabrani korisnik.

Akteri koji iniciraju slučaj korišćenja: Običan korisnik, turistički vodič i administrator.

### Pregled podataka o organizatoru obilaska

Kratak opis: Prikaz profila organizatora izabranog obilaska.

Akteri koji iniciraju slučaj korišćenja: Običan korisnik, turistički vodič i administrator.

### Dodavanje novog obilaska

Kratak opis: Dodavanje novog obilaska koji organizuje turistički vodič, od strane tog istog turističkog vodiča.

Akteri koji iniciraju slučaj korišćenja: Turistički vodič.

### Otkazivanje obilaska

Kratak opis: Brisanje zakazanog obilaska od strane turističkog vodiča koji je taj obilazak organizovao i dodao, ili od strane administratora.

Akteri koji iniciraju slučaj korišćenja: Turistički vodič, administrator.

### Promena rute obilaska

Kratak opis: Izmena osnovnih podataka o obilasku koji je izabran, odnosno rute.

Akteri koji iniciraju slučaj korišćenja: Turistički vodič, Administrator.

### Promena datuma obilaska

Kratak opis: Izmena osnovnih podataka o obilasku koji je izabran, odnosno datuma.

Akteri koji iniciraju slučaj korišćenja: Turistički vodič, Administrator.

### Prijavljivanje korisnika za obilazak

Kratak opis: Korisnik se prijavljuje za izabrani obilazak.

Akteri koji iniciraju slučaj korišćenja: Običan korisnik, turistički vodič, administrator.

### Komentarisanje obilaska

Kratak opis: Korisnik postavlja svoje komentare o obilasku na kojem je učestvovao.

Akteri koji iniciraju slučaj korišćenja: Običan korisnik, turistički vodič, administrator.

### Pregled liste korisnika sistema

Kratak opis: Prikazivanje liste svih korisnika registrovanih na sistem.

Akteri koji iniciraju slučaj korišćenja: Administrator.

### Brisanje postojećeg korisnika

Kratak opis: Brisanje korisničkog naloga i podataka za postojećeg korisnika sistema.

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.

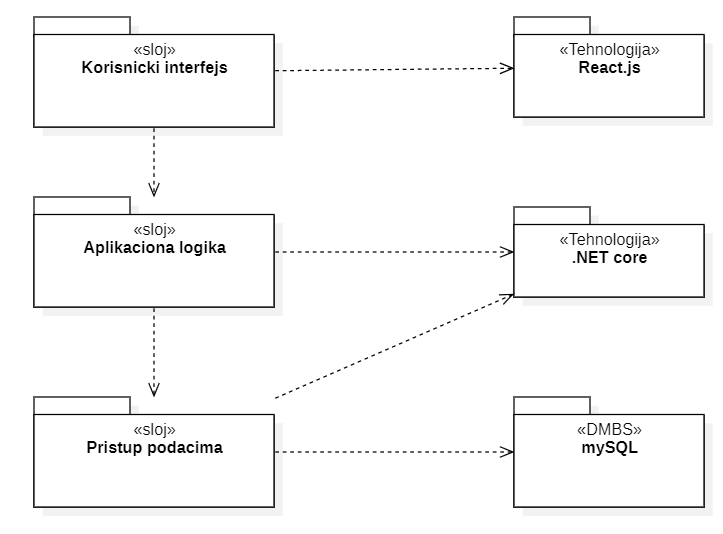
Logički pogled na BioTS web aplikaciju obuhvata 3 glavna paketa: Korisnički interfejs, Aplikaciona logika, Pristup podacima.

Paket *Korisnički* sadrži osnovnu HTML stranicu, React komponente, JS skripte i multimedijalni sadržaj kojim se realizuje grafički dizajn i forme preko kojih korisnici interaguju sa sistemom.

Paket *Aplikaciona logika* predstavlja srednji sloj sistema koji sadrži C# .NET fajlove u kojima je implementirana poslovna logika domena primene i preko kojih se vrši komunikacija sa bazom podataka.

Paket *Pristup podacima* sadrži C# .NET fajlove kojima je implementiran model domena primene, uključujući i odgovarajuća EF mapiranja.

## Pregled arhitekture – organizacija paketa i podsistema u slojeve



### Korisnički interfejs

sloj

Ovaj sloj realizuje korisnički interfejs portala. U njemu su sadržane sve HTML i CSS skripte, multimedijalni sadržaji i React.js komponente, preko kojih se generišu HTML stranice koje omogućavaju korisnicima interakciju sa sistemom.

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

### Aplikaciona logika

sloj

Sloj aplikacione logike je srednji sloj u troslojnoj arhitekturi BioTS web aplikacije. S adrži .NET Core kontrolere koji 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 .NET Core framework-a.

### Pristup podacima

sloj

Sloj za pristup podacima se nalazi na dnu troslojne arhitekture i sadrži C# fajlove koji koriste .NET Core za impelemntaciju logike potrebne za pristup podacima preko kontrolera pomenutih u sloju iznad.

Ovaj sloj ne zavisi od drugih slojeva, ali zavisi od .NET Core framework-a i mySQL baze podataka.

### React.js

tehnologija

Tehnologija React omogućava dinamičko kreiranje HTMl elemenata i upravljanje događajima tih elemenata. Izvršava se na klijentskoj strani pomoću implementiranih skripti.

### .NET Core

tehnologija

Tehnologija .NET Core se izvršava na serverskoj strani, obezbeđuje mehanizam za pisanje modela i kontrolera. Ovi kontroleri implementiraju samu aplikacionu logiku i pristupaju bazi podataka u cilju pribavljanja, unosa i ažuriranja podataka.

### MySQL

DBMS

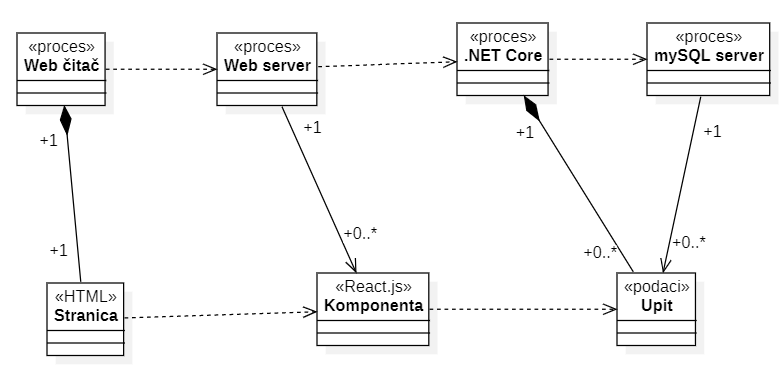
MySQL predstavlja sistem za upravljanje bazama podataka koji će se koristiti za realizaciju bioTS web aplikacije.

# Pogled na procese

U ovom odeljku je sadržan pogled na procesnu arhitekturu sistema. Ovaj opis sadrži specifikaciju različitih zadataka (procesa i niti) uključenih u rad sistema. Takođe, dati su dijagrami koji pokazuju njihovu interakciju i konfiguraciju.

## Procesi

Na sledećem UML dijagramu klasa prikazani su procesi koji učestvuju u izvršenju BioTS web aplikacije.



### Web čitač

Web čitač je proces koji služi za prikaz HTML stranice dobijene od nekog Web servera. U najopštijem slučaju Web pretraživač može da prikazuje samo jednu HTML stranicu u jednom pokrenutom procesu. U slučaju SPA aplikacije, više “stranica” se dostavlja Web čitaču, ali se komponente koje su prikazane u jednom trenutku određuju dinamički.

### Web server

Web server je proces koji izvršava funkcionalnost opsluživanja klijentskih zahteva. U slučaju SPA aplikacija pisanih u React.js, Web server kroz prvi zahtev vraća HTML stranicu, kao i sve prateće JS i CSS fajlove. Zahtevi koji nastaju kao posledica interakcije korisnika sa Web stranicom se prosleđuju .NET Core kontrolerima.

### .NET Core

.NET core je proces koji radi zajedno sa Web serverom. Osnanovna svrha web servera je da prosledi HTTP zahtev .NET core procesu. .NET Core procestaj zahtev obrađuje i vraća podatke klijentu u JSON formatu.

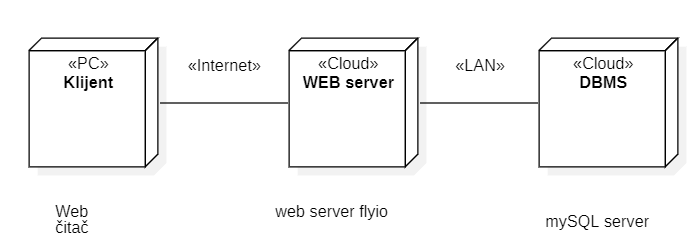
### MySQL Server

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 BioTS web aplikacije.



## Klijent

Pristup BioTS 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

Na Cloud-u se nalazi web server koji služi više klijenata preko interneta. Osim osnovnog procesa koji omogućava funkcionalnost web servera (flyio), na istom računaru se izvršavaju i kontroleri ASP.NET Core-a koji obrađuju pristigle zahteve.

## 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).

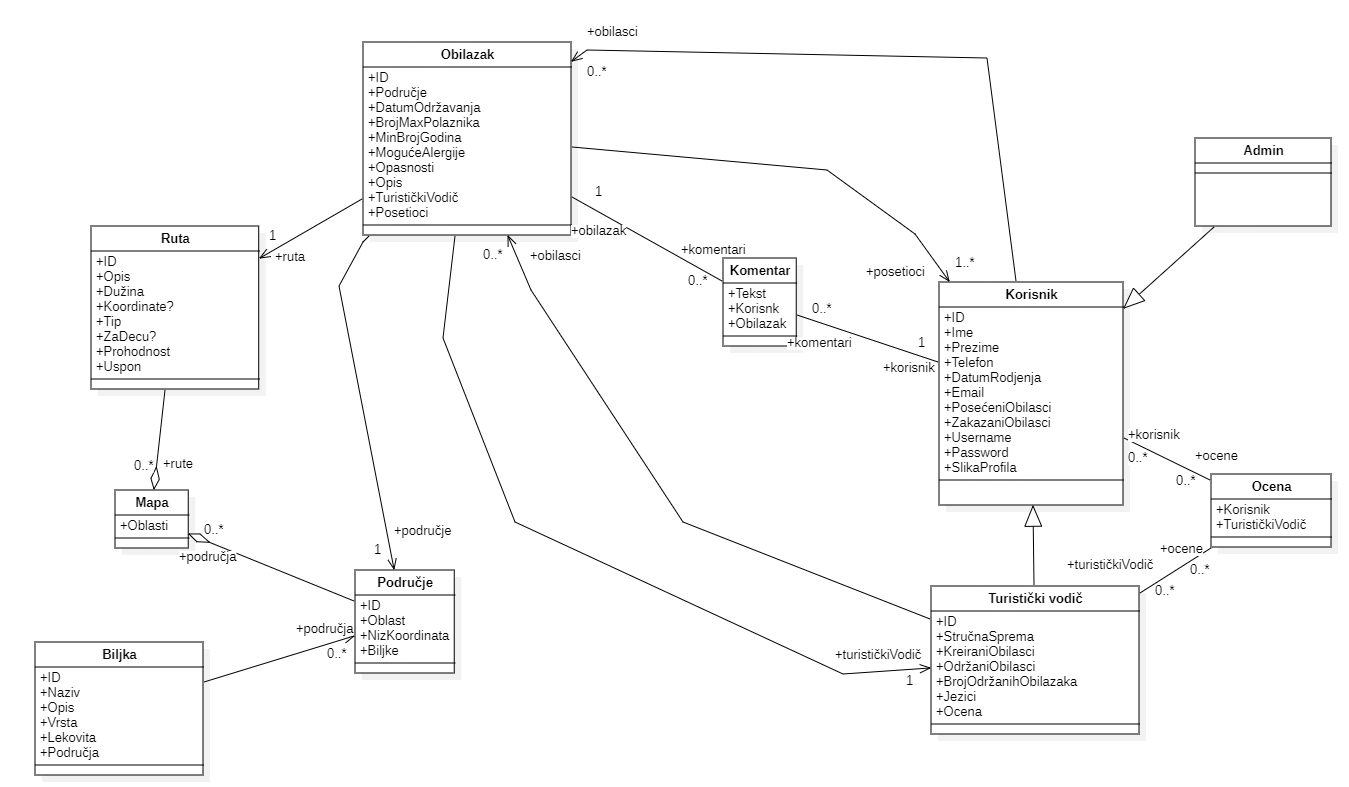
# Pogled na implementaciju sistema

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

## Model domena

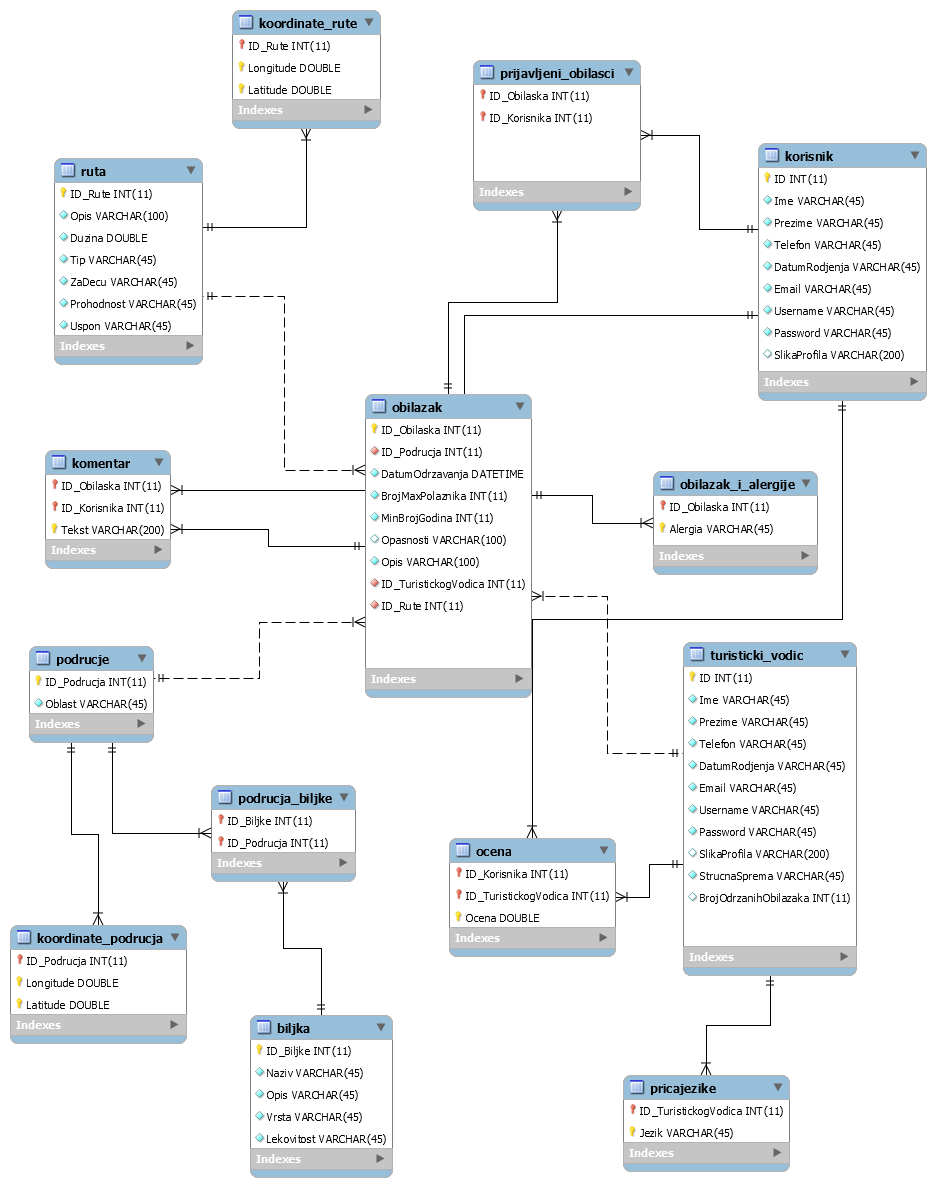
Model domena za koji se BioTS web aplikacije projektuje je ilustrovan UML dijagramom klasa. U njemu su prikazane domenske klase, neki od njihovih atributa, kao i veze koje se mogu identifikovati između njih.

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



## Šema baze podataka

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

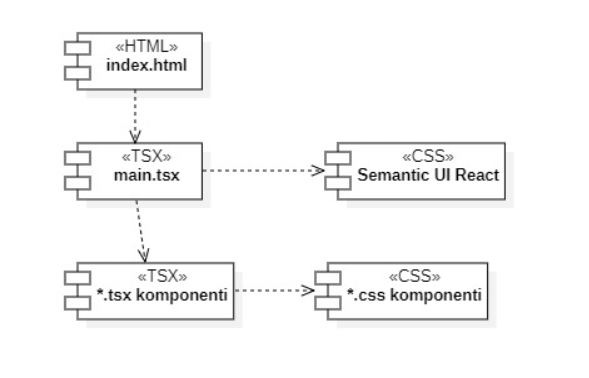


## Komponente sistema

Komponente sistema Eventopia platforme su React.js skripte i .NET fajlovi čiji će pregled biti dat po arhitekturnim slojevima. Za ilustraciju će biti korišćeni UML dijagrami komponenti i dijagrami klasa.

### Komponente korisničkog interfejsa

Dizajn interfejsa za neprijavljenog korisnika web aplikacije BioTS može se predstaviti komponentama sa slike:



Komponenta **index.html** je osnovna stranica koja se prikazuje korisniku nakon pristupanja sistemu. Ona se učitava zajedno sa prikazanim CSS i TS fajlovima.

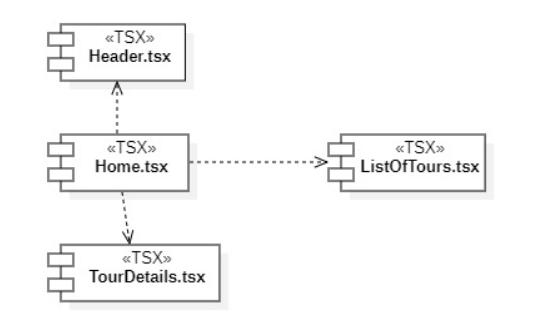
Komponenta **main.tsx** pruža provider usluge svim ugnježdenim komponentama, učitava sve globalne css stilove i predstavlja ulaznu tačku za sve TS fajlove.

**CSS fajl** predstavlja stil određene react komponente.

**TSX fajl** implementira određenu react komponentu.

**Semantic UI React** je biblioteka korisničkog interfejsa za razvoj web aplikacija koja kombinuje React sa stilizovanim komponentama koje pruža Semantic UI.

Dizajn korisničkog interfejsa za običnog korisnika BioTS web aplikacije obuhvata sledeće komponente:



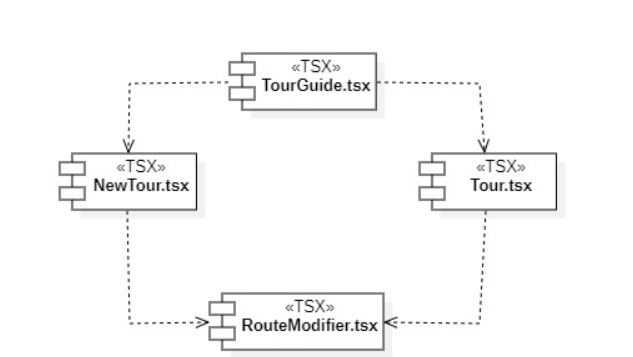
**HomePage.txt** predstavlja osnovnu stranicu, koja se prikazuje bilo kom tipu korisnika kada pristupi BioTS web aplikaciji. Korisnik može da pregleda kartu Srbije i pregleda informacije o biljkama sa ove komponente, Takođe može pristupiti prijavljivanju i registrovanju na platformu.

**Header.tsx** predstavlja navigacioni panel preko kojeg se može pristupiti prijavljivanju i registrovanju na platformu i pristup drugim sekcijama.

**ListOfTours.txt** je komponenta koja prikazuje rezultate filtriranja sa svim obilascima koji odgovaraju kriterijumima.

**TourDetails.tsx** je komponenta koja prikazuje detaljne informacije o izabranom obilasku. Takođe nudi i prijavljivanje za taj obilazak korisnicima sistema.

Dizajn korisničkog interfejsa za turističkog vodiča obuhvata sledeće komponente:



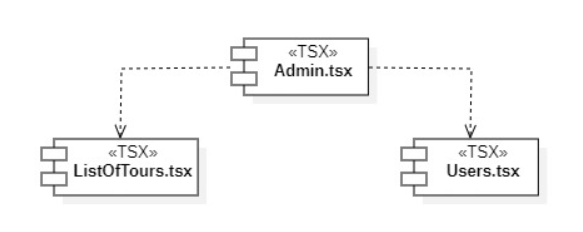
**TourGuide.tsx** je komponenta koja je prikazana turističkom vodiču kada je on prijavljen na sistem. Obuhvata osnovne informacije koje se prikazuju.

**NewTour.tsx** sadrži formu koju turistički vodič popunjava prilikom organizovanja novog obilaska.

**RouteModifier.tsx** je editor koji omogućava turističkom vodiču da kreira rutu po kojoj će se obilazak odvijati.

**Tour.tsx** komponenta pruža turističkom vodiču pregled svih obilazaka koje je održao ili koje tek treba da održi.

Dizajn korisničkog interfejsa koji se prikazuje kada je administrator prijavljen na sistem:

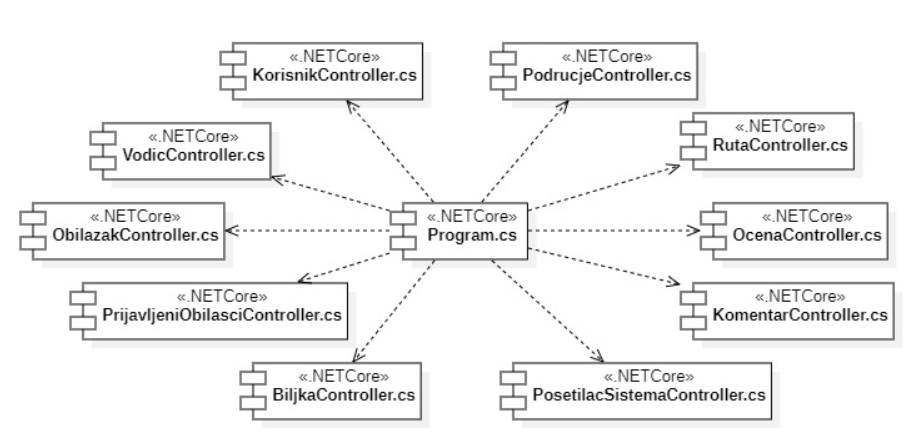


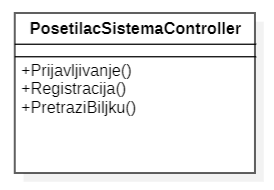
Admin.tsx komponenta predstavlja komponentu koja je prikazana administratoru prilikom prijavljivanja na sistem.

Users.tsx predstavlja komponentu koja obuhvata listu svih korisnika i mogućnosti i akcije koje administrator sistema može da izvrži nad korisnicima sistema.

ListOfTours.tsx administratoru pruža uvid u listu svih obilazaka i odgovarajuće akcije koje može da izvrši nad njima (brisanje, ažuriranje).

### Komponente aplikacione logike

Preko Program.cs fajla se uključuju u aplikaciju komponente koje realizuju domen problema, odnosno Controller klase u .NET Core framework-u. Na sledećem dijagramu su prikazane komponente ovog sloja i njihove zavisnosti:



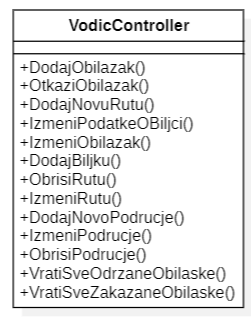
Navedene metode iz kontrolera obavljaju sledeće zadatke:

* Prijavljivanje() - proverava unete podatke, i prijavljuje korisnika na platformu
* Registracija() - vrsi novu validaciju i registruje novog korisnika odredjenog tipa
* PretraziBiljku() - proverava da li biljka postoji u bazi i prikazuje podrucje u kome se nalazi na mapi



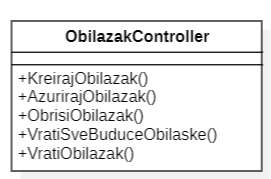
Navedene metode iz kontrolera obavljaju sledeće zadatke:

* Prijavljivanje() - proverava unete podatke, i prijavljuje korisnika na platformu
* PretraziBiljku() - proverava da li biljka postoji u bazi i prikazuje podrucje u kome se nalazi na mapi
* PrijaviObilazak() - prijavljivanje korisnika za postojeci obilazak
* OceniVodica() - dodeljuje ocenu vodicu i azurira novu prosecnu ocenu
* OtkaziObilazak() - otkazuje na prijavljani obilazak
* IzmeniPodatkeOSebi() - promena licnih podataka
* VratiSvePrethodneObilaske() - vraca obilaske na kojima je korisnik bio
* VratiSveBuduceObilaske() - vraca sve obilaske na kojima tek treba da prisustvuje



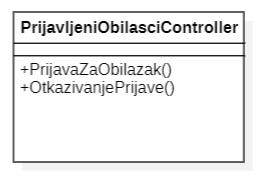
Navedene metode iz kontrolera obavljaju sledeće zadatke:

* DodajObilazak() - kreira novi obilazak, ako vec ne postoji
* OtkaziObilazak() - otkzuje postojeci obliazak
* DodajNovuRutu() - dodaje novu rutu, ako vec ne postoji
* IzmeniObilazak() - menja podatke o obilasku
* DodajBiljku() - dodaje biljku, ako vec ne postoji u bazi
* IzmeniPodatkeOBiljci() - menja podatke o biljci
* ObrisiRutu() - brisanje postojece rute
* IzmeniRutu() - promena podataka o postojecoj ruti
* DodajNovoPodrucje() - dodavanje novog podrucja, ako vec ne postoji
* ObrisiPodrucje() - brisanje postojeceg podrucja podrucja
* IzmeniPodrucje() - promena podataka o podrucju
* VratiSveZakazaneObilaske() - vraca sve obilaske koje je on kreirao i tek treba da se dese



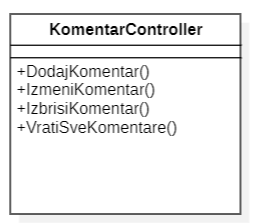
Navedene metode iz kontrolera obavljaju sledeće zadatke:

* KreirajObilazak() - kreira novi obilazak
* OtkaziObilazak() - otkazuje postojeci obilazak
* IzmeniObilazak() - menja podatke o postojecem obilasku
* VratiSveBuduceObilaske() - vraca sve buduce obilaske
* VratiObilazak() - vraca konkretan obilazak



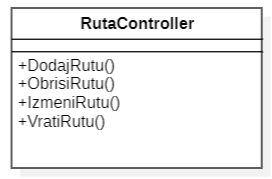
Navedene metode iz kontrolera obavljaju sledeće zadatke:

* PrijavaZaObilazak() - prijavljiuje korisnika za postojeci obilazak
* OtkazivanjePrijave() - otkazuje postojecu prijavu



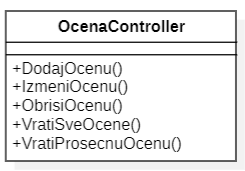
Navedene metode iz kontrolera obavljaju sledeće zadatke:

* DodajKomentar() - dodaje novi komentar
* IzbrisiKomentar() - brise postojeci komentar
* IzmeniKomentar() - menja sadrzaj postojeceg komentara
* VratiSveKomentare() - vraca sve komentare koji su vezani za obilazak



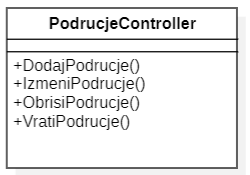
Navedene metode iz kontrolera obavljaju sledeće zadatke:

* DodajRutu() - dodaje rutu, ako vec ne postoji
* IzbrisiRutu() - brisanje vec postojece rute
* IzmeniRutu() - izmena podataka o ruti
* VratiRutu() - vraca postojecu rutu za konktretan obilazak



Navedene metode iz kontrolera obavljaju sledeće zadatke:

* DodajOcenu() - dodaje ocenu vodicu i azurira njegovu prosecnu ocenu
* IzbrisiOcenu() - brisanje vec dodate ocene
* PromeniOcenu() - izmene vec dodate ocene
* VratiProsecnuOcenu() - vraca prosecnu ocenu vodicu
* VratiSveOcene() - vraca sve ocene za konkretnog vodica



Navedene metode iz kontrolera obavljaju sledeće zadatke:

* DodajPodrucje() - dodaje novo podrucje, ako vec ne postoji
* IzmeniPodrucje() - izmena podataka o postojecem podrucju
* ObrisiPodrucje() - brisanje postojeceg podrucja
* VratiPodrucje() - vraca podrucje za konkretnu biljku

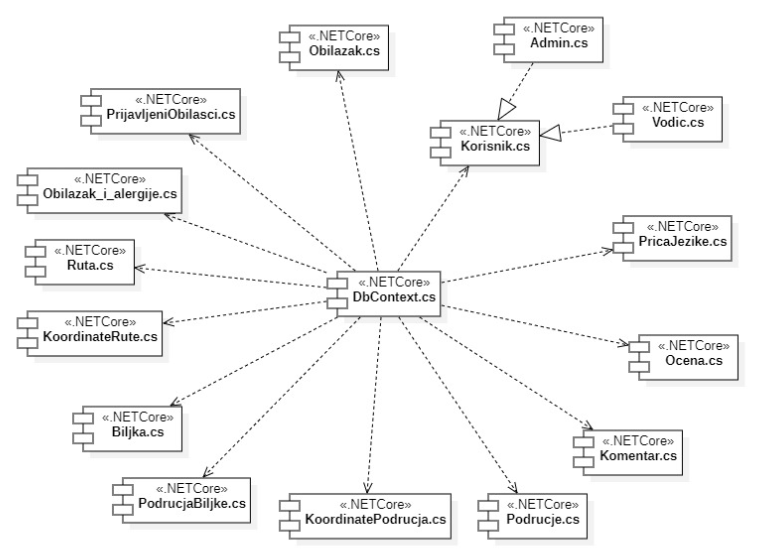


Navedene metode iz kontrolera obavljaju sledeće zadatke:

* SortirajBiljke() - sortira biljke na osnovu atributa
* DodajBiljku() - dodaje novu biljku, ako vec ne postoji u bazi
* IzmeniBiljku() - brise postojecu biljku
* VratiSveBiljke() - vraca sve biljke

### Komponente za pristup podacima

S obzirom da se pristup podacima ostvaruje kroz kontrolere koji su kreirani na osnovu objektno relacionih mapera Entity Framework-a, nije potrebno da se eksplicitno definišu funkcije za pristup i pretraživanje podataka jer je to omogućeno kroz kontrolere. Na sledećem dijagramu prikazane su komponente koje čine model i koje se preslikavaju na bazu podataka pomoću Entity Framework-a.



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