**SWE-HealthAppointment-01**

**Portal zdravstvene ustanove**

Arhitekturni projekat

Verzija 1.0

Pregled izmena

| **Datum** | **Verzija** | **Opis** | **Autor** |
| --- | --- | --- | --- |
| 03.05.2022. | 1.0 | Inicijalna verzija | Đorđe Mitrović, Nikola Stanković, Nikola Jovanović, Matija Milutinović |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

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 6

6.2.1 Pregled osnovnih podataka o klinici 6

6.2.2 Pregled podataka o doktorima 6

6.2.3 Pregled podataka o određenom doktoru klinike 6

6.2.4 Pregled doktora klinike filtriranih na osnovu specijalizacije 6

6.2.5 Pregled podataka iz ugla posetilaca 6

6.2.6 Pregled podataka iz ugla pacijenata 7

6.2.7 Pregled podataka iz ugla lekara opšte prakse i lekara specijalizanata 7

6.2.8 Kreiranje naloga 7

6.2.9 Prijavljivanje 7

6.2.10 Ažuriranje podataka o pacijentu 7

6.2.11 Organizovanje lekarskog pregleda 7

6.2.12 Unos podataka o pregledu 7

6.2.13 Ažuriranje podataka o pregledu 7

6.2.14 Brisanje postojećeg pacijenta 7

6.2.15 Arhiviranje postojećeg pacijenta 7

6.2.16 Recenzija 7

7. Pogled na logičku arhitekturu sistema 7

7.1 Pregled arhitekture – organizacija paketa i podsistema u slojeve 8

7.1.1 Korisnički interfejs 8

7.1.2 Aplikaciona logika 8

7.1.3 Logika pristupa podacima 8

7.1.4 HTML, CSS 8

7.1.5 Javascript [8](#_heading=h.3j2qqm3)

7.1.6 .NET Core 8

7.1.7 Firebase [8](#_heading=h.3j2qqm3)

8. Pogled na procese 8

8.1 Procesi 8

8.1.1 Client 8

8.1.2 Server [9](#_heading=h.1y810tw)

8.1.2 Firebase 9

9. Pogled na implementaciju sistema 10

9.1 Prijavljivanje u aplikaciju 10

9.2 Korišćenje sistema aplikacije 10

9.3 Šema baze podataka 11

10. Pogled na raspoređivanje sistema 12

10.1 Klijent 12

10.2 Web Server 12

10.3 DBMS Server 12

11. Performanse 12

12. Kvalitet 12

Arhitekturni projekat

# Cilj dokumenta

Cilj ovog dokumenta je detaljni opis arhitekture Health Appointment sistema.

# Opseg dokumenta

Dokument se odnosi na Health Appointment portal koji će biti razvijen od strane Доктори tima. Health Appointment predstavlja naziv za Web aplikacija za rad klinike. Namena sistema je efikasno prezentovanje, kreiranje i održavanje sadržaja vezanih za rad privatne zdravstvene klinike.

# Reference

Spisak korišćene literature:

1. HealthAppointment – Predlog projekta, SWE-HealthAppointment-01, V1.0, 2022, Доктори.
2. HealthAppointment – Vizija sistema, V1.0, 2022, Доктори.
3. HealthAppointment – Planirani raspored aktivnosti na projektu, V1.0, 2022, Доктори
4. HealthAppointment – Plan realizacije, V1.0, 2022, Доктори.
5. HealthAppointment – Specifikacija zahteva, V1.0, 2022, Доктори.

# 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. Health appointment će biti realizovana kao WEB aplikacija zasnovana na JavaScript biblioteci (ReactJS), .NET Core backend-u i usluge Firebase baza podataka [4].
2. Klijentski deo Health Appointment portala će biti optimizovan za: Internet Explorer 6.0 i noviji, Opera 8.0 i noviji, kao i Firefox (Mozilla), Google Chrome, Safari i ostale Web čitače [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].

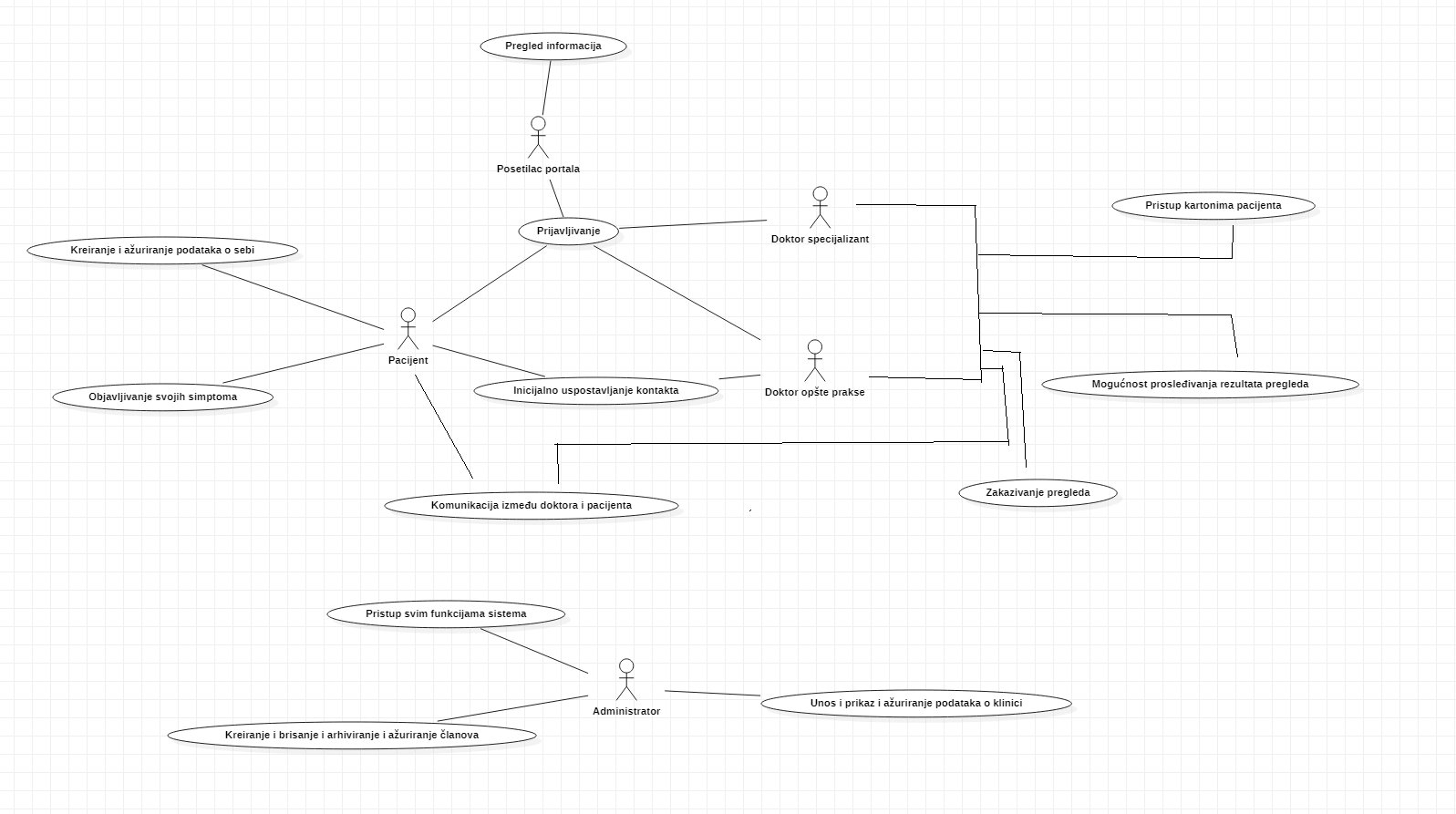
Slučajevi korišćenja Health Appointment sistema su:

* Pregled osnovnih podataka o klinici
* Pregled podataka o doktorima
* Pregled podataka o određenom doktoru klinike
* Pregled doktora klinike filtriranih na osnovu specijalizacije
* Pregled doktora klinike filtriranih na osnovu recenzije
* Pregled podataka iz ugla posetilaca
* Pregled podataka iz ugla pacijenata
* Pregled podataka iz ugla lekara opšte prakse i lekara specijalizanata
* Kreiranje naloga
* Prijavljivanje
* Administrator može it baze - Ažuriranje podataka o pacijentu
* Organizovanje lekarskog pregleda
* Unos podataka o pregledu
* Ažuriranje podataka o pregledu
* Brisanje postojećeg pacijenta
* Recenzija
* Četovanje

Ove slučajevi korišćenja mogu da iniciraju posetilac portala, pacijent, doktor opšte prakse, doktor specijalizant ili administrator.

## Dijagrami slučajeva korišćenja

UML dijagram koji prikazuje korisnike i slučajeve korišćenja Health Appointment sistema prikazan je na sledećoj slici:



## Kratak opis slučajeva korišćenja

### Pregled osnovnih podataka o klinici

Kratak opis: Prikaz stranice portala sa osnovnim podacima o klinici.

Akteri koji iniciraju slučaj korišćenja: Posetilac portala.

### Pregled podataka o doktorima

Kratak opis: Prikaz stranice portala sa spiskom doktora opšte prakse i doktora specijalizanata (skraćeno doktorima).

Akteri koji iniciraju slučaj korišćenja: Pacijent, Doktor opšte prakse, Doktor specijalizant, Administrator.

### Pregled podataka o određenom doktoru klinike

Kratak opis: Prikaz stranice sa informacijama o određenom doktoru klinike.

Akteri koji iniciraju slučaj korišćenja: Pacijent, Doktor opšte prakse, Doktor specijalizant, Administrator.

### Pregled doktora filtriranih na osnovu specijalizacije

Kratak opis: Prikaz stranice portala sa spiskom doktora klinike sortiran na osnovu specijalizacije.

Akteri koji iniciraju slučaj korišćenja: Pacijent, Doktor opšte prakse, Doktor specijalizant, Administrator.

### Pregled podataka iz ugla posetilaca

Kratak opis: Prikaz stranice sa svim svojim osnovnim podacima. Posetilac moze da vidi osnovne podatake na sajtu, ali nema mogucnost direktne komunikacije sa lekarima. Ima jedinu opciju kontakt.

Akteri koji iniciraju slučaj korišćenja: Posetilac portala, Pacijent, Doktor opšte prakse, Doktor specijalizant, Administrator.

### Pregled podataka iz ugla pacijenta

Kratak opis: Prikaz stranice sa forumom gde može da objavi svoje simptome.

Akteri koji iniciraju slučaj korišćenja: Pacijent, Administrator.

### Pregled podataka iz ugla lekara opšte prakse i ostalih lekara specijalizanata

Kratak opis: edina razlika između ove dve vrste lekara je uvid rekara opšte prakse u forum na kome pacijenti pišu simptome. Specijalizanti nemaju uvid u opšti forum pacijenata.

Akteri koji iniciraju slučaj korišćenja: Pacijent, Doktor opšte prakse, Doktor specijalizant, Administrator.

### Kreiranje naloga

Kratak opis: Samo pacijenti klinike i lekari mogu da naprave nalog - ljudi koju su slučajni posetioci sajta to ne mogu dok ne budu korisnici usluga klinike ili zaposleni u njoj.

Akteri koji iniciraju slučaj korišćenja: Pacijent, Doktor opšte prakse, Doktor specijalizant, Administrator.

### Prijavljivanje

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

Akteri koji iniciraju slučaj korišćenja: Pacijent, Doktor opšte prakse, Doktor specijalizant, Administrator.

### Ažuriranje podataka o pacijentu

Kratak opis: Ažuriranje podataka o pacijentu može admin iz baze podataka..

Akteri koji iniciraju slučaj korišćenja: Pacijent, Doktor opšte prakse, Doktor specijalizant, Administrator.

### Organizovanje lekarskog pregleda

Kratak opis: Zakazivanje novog pregleda od strane lekara.

Akteri koji iniciraju slučaj korišćenja: Pacijent, Doktor opšte prakse, Doktor specijalizant.

### Unos podataka o pregledu

Kratak opis: Unos podataka o odrađenom pregledu od strane lekara(opšte prakse ili lekara specijaliste).

Akteri koji iniciraju slučaj korišćenja: Pacijent, Doktor opšte prakse, Doktor specijalizant, Administrator.

### Ažuriranje podataka o pregledu

Kratak opis: Izmena podataka o obavljenom pregledu.

Akteri koji iniciraju slučaj korišćenja: Doktor opšte prakse, Doktor specijalizant, Administrator.

### Brisanje postojećeg pacijenta

Kratak opis: Brisanje korisničkog naloga i podataka za postojećeg pacijenta.

Akteri koji iniciraju slučaj korišćenja: Administrator.

### Četovanje

Kratak opis: Pacijeti sa doktorom može da se dopisuje kako bi imali što bolju saradnju za što brže izlečenje

### Recenzija

Kratak opis: Ocenjivanje doktora nakon odrađenih pregleda.

Akteri koji iniciraju slučaj korišćenja: Pacijent, Doktor opšte prakse, Doktor specijalizant, 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.

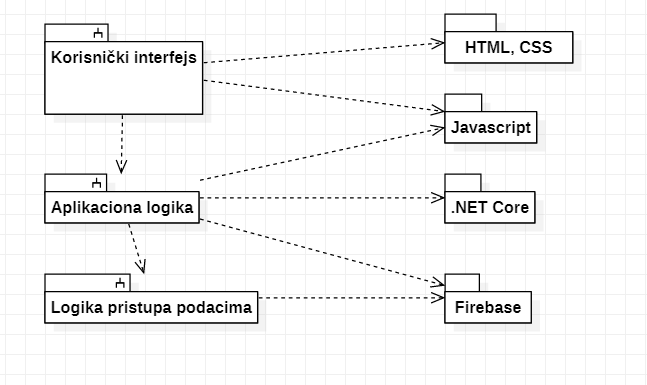
Logička arhitektura Health Appointment sistema obuhvata 3 glavna podsistema: Korisnički interfejs, aplikaciona logika i logika upravljanja podacima.

Korisnički interfejs je zadužen za grafički prikaz aplikacije, na kome se može naći različit multimedijalni sadržaj koji interaguje sa korisnikom. Realizovan je u HTML-u, CSS-u i JavaScriptu.

Aplikaciona logika predstavlja logiku funkcionisanja aplikacije, kakve probleme rešava aplikacija, kakva rešenja nudi i svoju namenu. Predstavlja medijum između korisničkog interfejsa i logike upravljanja podacima.

Logika upravljanja podacima ima ulogu u čuvanju, skladištenju, učitavanju podataka za njihovu buduću upotrebu, kao i odgovarajuću njihovu organizaciju. Podaci se čuvaju u vidu dokumenata preko usluga Firebase-a, dok se podaci iz njih vuku preko backend servera.

## Pregled arhitekture – organizacija paketa i podsistema u slojeve



### Korisnički interfejs

podsistem

U korisničkom interfejsu se realizuje grafički dizajn aplikacije. Svaki vidljiv detalj kod aplikacije rađen je u ovom podsistemu. Tehnologije u kojima se izrađuje ovaj podsistem su HTML, koji služi za skelet stranice, CSS koji služi za stilizovanje komponenti stranice, i u najvećem delu Javascript, koji uvodi dinamičnost i interaktivnost kod stranice. U ReactJS-u se 99% stvari dizajniraju u JavaScriptu koji kasnije Web čitači pretvaraju u HTML i CSS format.

UML dijalgram klasa koji prikazuje organizaciju klasu u okviru ovog projekat je dat u nastavku.

### Aplikaciona logika

podsistem

Aplikaciona logika je najzaslužnija za funkcionisanje Web aplikacije i predstavlja medijum između FrontEnd-a i BackEnd-a. Ona je zadužena za pravilno uzimanje upita sa klijentske strane prosleđeni serverskoj strani kao i za uzvraćanje odgovora sa dodatnim informacijama. Ovaj podsistem predstavlja najbitniju logiku kod arhitekture aplikacije.

### Logika pristupa podacima

podsistem

Logika pristupa podacima zasnovana je na pravilnoj organizaciji podataka kao i njihovom čuvanju i skladištenju. Zadužena je za sigurnost i transparentnost podataka, kao i na održavanju organizaciji podataka. Komunicira sa korisničkim interfejsom putem aplikacione logike. U ovom podsistemu se nalaze svi podaci aplikacije.

### HTML, CSS

jezici za markiranje

HTML (Hyper Text Markup Language) i CSS (Cascade Style Sheet) su jezici za formatiranje i služe kako bi aplikacija dobila svoj grafički izgled. Korisnički interfejs se u najvećoj meri zasniva na jezicima za markiranje.

### Javascript

skriptni jezik

Javascript u aplikaciju dovodi interaktivnost i dinamičnost, takođe i povezuje klijentski deo aplikacije sa serverskim delom. Cela logika izrade aplikacije se zasniva na ovom skriptnom jeziku.

### .NET Core

Backend framework

.NET Core predstavlja framework za izradu Backend servera i zadužen je za odgovaranje na upite klijentskog dela koji su usmereni serverskom delu. Predstavlja medijum između klijentskog i serverskog dela koji obavlja prenos podataka kao i određenu validaciju i autentikaciju.

### Firebase

platforma za upravljanje podacima

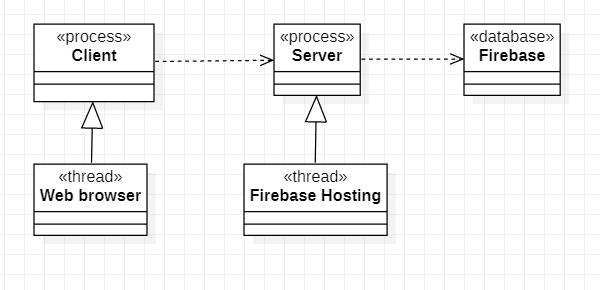
Firebase je platforma koje nudi različite usluge za upravljanje podacima, ali i celokupnom aplikacijom. Aplikacija se nalazi na hosting provajderu koji nudi Firebase. Firebase poseduje takozvanu Firestore bazu podataka koja se u ovoj aplikaciji koristi u najvećoj meri.

# 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.

## Procesi

Na sledećem UML dijagramu klasa prikazani su procesi koji učestvuju u izvršenju Health Appointment sistema.



### Client

Aplikacija na klijentu preko glavne niti koja se održava preko Web Browsera služi za preuzimanje sadržaja aplikacije sa servera preko HTTP protokola.

*Client* zavisi od procesa servera, sa koga se preuzimaju sadržaji.

### Server

Serverska aplikacija poseduje glavnu nit na hostingu koji nudi Firebase platforma i ona služi za održavanje dostupne veze svim klijentima koji pristupaju aplikaciji.

*Server* zavisi od platforme Firebase na kome se nalazi hosting.

### Firebase

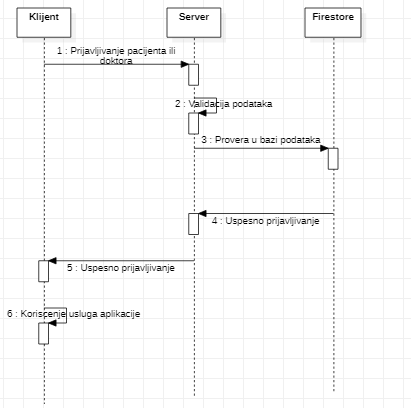
Firebase poseduje bazu podataka i hosting provajder koji nudi serveru globalni pristup aplikaciji kao i svim njegovim podacima.

# Pogled na implementaciju sistema

Pogled na implementaciju prikazuje različite aspekte bitne za implementaciju sistema. U slučaju Health Appointment-a ovaj odeljak sadrži opis prijavljivanja u aplikaciju i ispravno korišćenje sistema.

## Prijavljivanje u aplikaciju

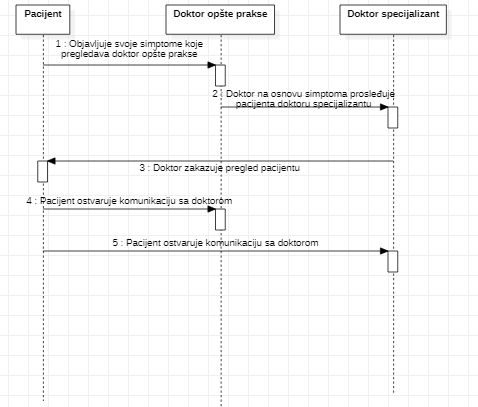
Sekvencijalni dijagram prijavljanja korisnika na aplikaciju je prikazan ispod.



Najpre klijent šalje zahtev serveru da želi da se prijavi. Nakon toga server validira ispravnost podataka, pa zatim proverava te podatke u Firestore bazi podataka. Nakon što je pronašao te podatke i podudaraju se, šalje se nazad poruka do klijenta da je prijava uspešna, i korisnik može da koristi usluge aplikacije.

## Korišćenje sistema aplikacije

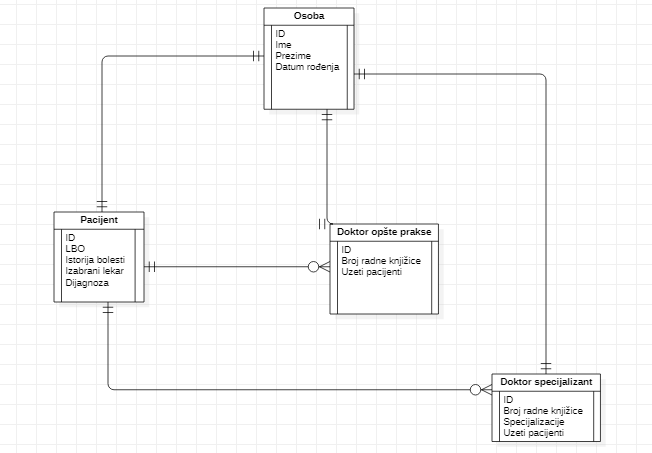
Korišćenje sistema aplikacije je prikazano na sekvencijalnom dijagramu ispod.



Korisnik koji je prijavljen kao pacijent, može da okači svoje simptome na aplikaciju. Doktori opšte prakse gledaju simptome koje su okačili pacijenti i zaključno sa tim mogu da zakažu pregled ili ga proslede nekom doktoru specijalizantu. Doktor specijalizant kod koga je prosleđen pacijent zakazuje pregled pacijentu. Pacijent u svakom trenutku ima pravo da ostvari direktnu komunikaciju sa doktorom koji mu je raspodeljen.

## Šema baze podataka

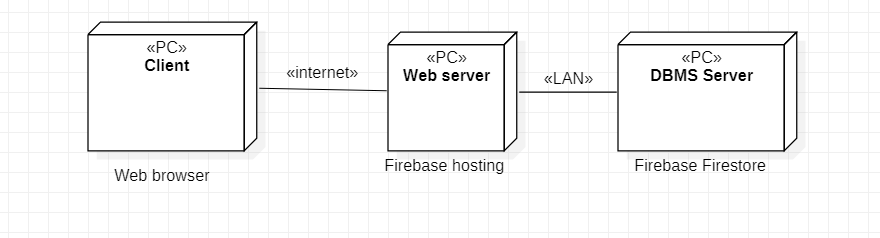
U bazi podataka nalaze se 4 glavne tabele: Osoba, Pacijent, Doktor opšte prakse i Doktor specijalizant. Postoje još mnoštvo dodatnih, helper i katalog tabela koje nemaju veliki uticaj na rad same aplikacije. Glavne tabele podataka prikazane su ispod na ER dijagramu.



# 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 komponenti.

Na sledećoj slici dat je UML dijagram raspoređivanja Health Appointment sistema.



## Klijent

Pristup Health Appointment portalu se obavlja preko Web browsera na klijentskim računarima gde se preuzima sadržaj aplikacije. Povezivanje klijenta i servera obavlja se preko internet mreže, samim tim nema ograničenja u pogledu lokacije klijenta.

## Web server

Web server se izvršava na jednom računaru koji komunicira sa više klijenta od jednom putem internet mreže. Nalazi se na Firebase Hosting provajderu. Na Web serveru se izvršavaju osnovni procesi koji omogućavaju osnovnu funkcionalnost Web aplikacija, ali nalaze se i različiti zahtevi, upiti i procesi koji dostižu putem Javascript skripti i upita putem .NET Core-a. Web server opslužuje klijente, dok je u konstantnoj komunikaciji sa DBMS serverom putem LAN mreže.

## DBMS server

DBMS server je realizovan na računaru koji omogućava svaku funkcionalnost po pitanju čuvanju, učitavanju i skladištenju podataka u bazi podataka. Zbog sigurnosti nema direktnu komunikaciju sa klijentom, već je komunikacija sa serverom moguća jedino putem Web Servera, koji je povezan lokalnom LAN mrežom.

# Performanse

Izabrana arhitektura softvera podržava zahteve u pogledu broja korisnika koji mogu simultano pristupati sistemu 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. Health Appointment sistem će biti dostupan 24 časa dnevno, 7 dana u nedelji. Vreme kada portal nije dostupan ne sme da pređe 16%.
2. Srednje vreme između dva sukcesivna otkaza ne sme da padne ispod 24 sata.