UNIVERZITET „DŽEMAL BIJEDIĆ“
FAKULTET INFORMACIJSKIH TEHNOLOGIJA

PREDMET: SOFTVERSKE ARHITEKTURE

BetterLife – Sistem za upravljanje fitness centrom

Dokumentacija softverske arhitekture

Odobrila :
Prof. Dr. Sc. Dražena Gašpar

Autor:
Anisa Suljić, IB180102

Akademska 2021/2022.godina
Mostar, 12. Januar 2022. godina
Verzija: V1

Sadržaj

1 Uvod ... 3
1.1 Svrha ... 3
1.2 Obim ... 3
1.3 Definicije i skraćenice ... 3
1.4 Zainteresirane strane ... 4
2 Predstavljanje softverske arhitekture ... 5
2.1 Kontekst sistema ... 5
2.2 Interakcija korisnika ... 5
2.3 Ciljevi arhitekture ... 6
3 Karakteristike softverske arhitekture ... 7
3.1 Performanse ... 7
3.2 Upotrebljivost ... 7
3.3 Sigurnost ... 7
3.4 Sposobnost održavanja ... 7
4 Značajni use-case-ovi ... 8
4.1 Narudžbe ... 8
4.2 Funkcije uposlenika ... 9
4.3 Upravljanje terminima ... 11
5 Vrste (paterni) softverske arhitekture ... 12
5.1 Paterni i karakteristike SA ... 12
5.2 Ograničenja paterna ... 13
5.3 Odluke i obrazloženja ... 13
6 Logički pogled ... 14
7 Procesni pogled ... 15
7.1 Odlazak na trening ... 15
7.2 Registracija i prijava ... 16
7.3 Narudžbe ... 17
8 Implementacijski pogled ... 18
9 Veličina i performanse ... 19
10 Kvaliteta ... 20
11 Reference ... 21

1 Uvod

1.1 Svrha

Fitness centar je organizacija koja broji veliki broj članova i različitih kategorija uposlenika. Upravo iz ovog razloga projekt BetterLife ima za svrhu olakšati i unaprijediti poslovanje fitness centra. Olakšanje korištenja sistema se manifestuje kroz način na koji administrativni dio upravlja, te upotrebu od strane klijenata koja postaje prejednostavna. Naime uprava fitness centra ima mogućnost da svu dokumentaciju i rad sa dokumentacijom i pravnim aktima obavlja na sistemu, zatim kadrovski odjel koji čine osoblje na prijemnim pultovima, korištenjem sistema mogu da evidentiraju i upisuju članove, prate status članarine, prate sva dešavanja vezana za termine (evidencija, rezervacija, rasporedi, ažuriranja i sl.). Dok s druge strane imamo klijente koji su konzumenti svih tih opcija. Oni se mogu učlaniti, mijenjati tip članarine, produžiti istu ili obustaviti, zatim imaju uvid u sve termine, mogućnost zakazivanja termina kao i kupovinu suplemenata i sportske opreme.

Spomenuto je da pored toga što sistem olakšava poslovanje, on radi i na unapređenju. Unapređenje se ogleda kroz rad odjela za prodaju i marketing (prodaja opreme i suplemenata, loyalty kartice i bonusi), čime se privlače novi članovi i zadržavaju postojeći.

1.2 Obim

Dati sistem će biti u opsegu određenog fitness centra i organizacije centra.

Obzirom da se sistem kreira za tačno određenu organizaciju i krojen je po određenim zahtjevima, on ne predstavlja nikakvu krovnu ili korijensku instituciju ili organizaciju nego je sam sebi dovoljan.

U sklopu sistema postoji administrativna strana sistema koju sačinjavaju **uprava** sistema koja ima uvid u pravne dokumente, vođenje poslovanja i zapošljavanje uposlenika. **Uposlenici** su također dio administrativne strane sistema obzirom da oni upravljaju članovima, članarinama i terminima. U administrativni dio spadaju još **marketar** koji upravlja marketing kampanjama za privlačenje novih članova i zadržavanje postojećih, te **prodavač** koji imaju uvid u stanje robe koju prodaju i sam proces narudžbe.

Sa klijentske strane imamo **članove** sistema koji imaju mogućnost da pregledaju novosti na sistemu. Zatim imaju mogućnost učlanjivanja na sistem čime dobivaju svoj profil kojim mogu upravljati. Mogućosti članova su i da produžavaju ili otkazuju članarinu, kupuju opremu i suplemente i rezervišu termine.

1.3 Definicije i skraćenice

API	Application Programming Interface
DB	Database

TCP	Transmission Control Protocol
IP	Internet protocol
UI	User interface
SOA	Service oriented architecture

1.4 Zainteresirane strane

Ovaj sistem ima pet zainteresiranih strana i to klijente, marketare, prodavače, uposlenike i upravu fitness centra.

Uprava koja kao što je već navedeno radi sve vezano za dokumentaciju i pravna pitanja u svrhu dobrog vođenja posla.

Uposlenici klijentima pružaju usluge članstva, uvida u termine, rezervacije, rasporede, promjenu tipa članarine, produžavanje iste i sl.

Prodavač klijentima nudi širok asortiman suplemenata i opreme za trening.

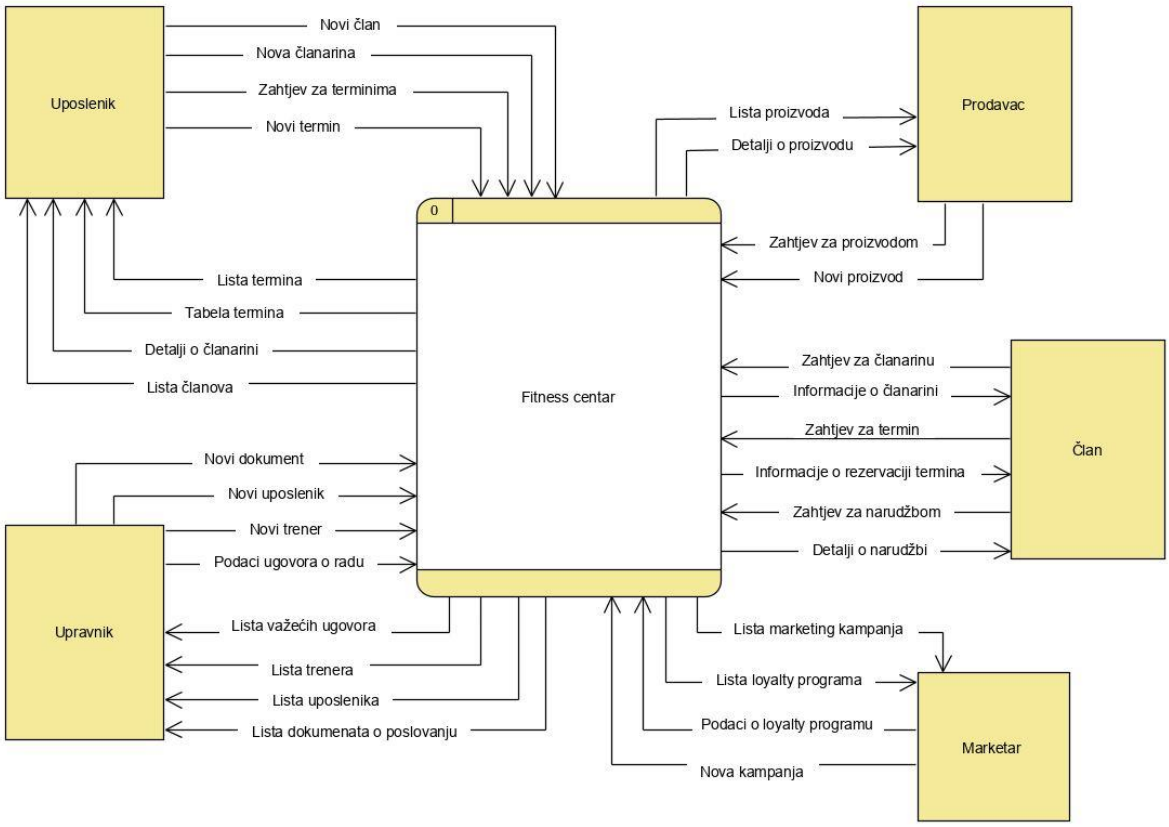
Marketar je tu za osmišljavanje marketinških planova koji će privući nove članove i kreiranje loyalty programa koji će zadržati postojeće.

Članovi / krajnji korisnici će imati svoje profile i moći će im pristupati kada požele. Obzirom da su konzumenti ovih usluga, oni mogu da traže bilo koju uslugu od uposlenika ili prodavača i s njima su u direktnoj vezi

2 Predstavljanje softverske arhitekture

2.1 Kontekst sistema

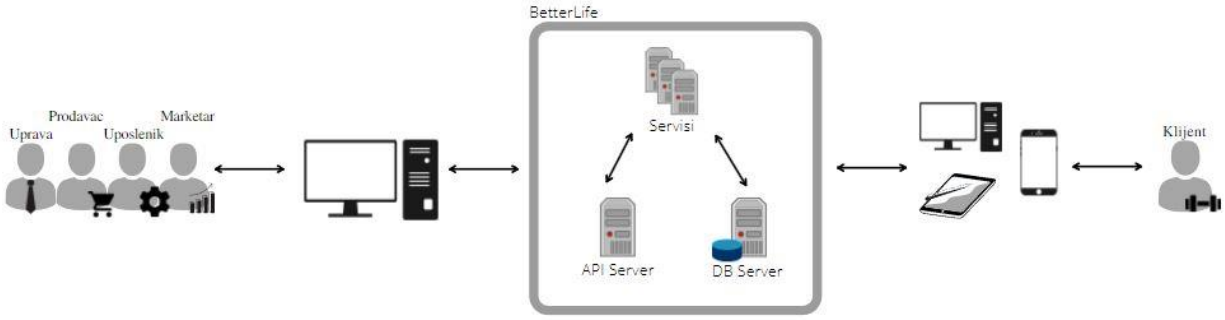
Kontekstualni dijagram predstavlja dijagram toka podataka najviše razine (0. razina) koji daje općenit prikaz procesa u njihovoj okolini. Na slici 1 je dijagram koji prikazuje odnos eksternih agenata koji su u interakciji sa sistemom, a tu skupinu čine : uposlenik, prodavač, upravnik, član i marketar.



Slika 1. Dijagram konteksta sistema

2.2 Interakcija korisnika

Interakcija sa sistemom je prikazana na slici 2 i osmišljena je tako da strana koja pripada organizaciji fitness centra (uprava, prodavač, marketar i uposlenici) koristi laptope ili personalne kompjutere, dok je planirano da članovi fitness centra imaju mogućnost korištenja bilo kojeg uređaja, odnosno nije bitno da li je to mobilni telefon, tablet ili laptop.



Slika 2. Dijagram interakcije korisnika sa sistemom

2.3 Ciljevi arhitekture

Problem sve više sistema danas jeste **sigurnost**. Upravo iz tog razloga je jako bitno da podaci koje klijenti ostavljaju u sistemu na korištenje budu **dobro zaštićeni** kako ne bi došlo do neovlaštenog pristupa ili promjene podataka na sistemu. Također je važno da se zaštite poslovni procesi i funkcionalnosti sistema kako bi njima mogle upravljati samo one osobe koje imaju permisije.

Naredni cilj se tiče **razumijevanja sistema** i njegovog korištenja. Naime potrebno je razviti sistem koji će biti **jednostavan** za shvatit i koristiti obzirom da je namijenjen **široj dobnoj skupini**, tako da sistem sigurno zahtjeva neku sekciju u kojoj će biti opisan cijeli proces korištenja funkcionalnosti sistema.

Kada smo se dotakli jednostavnosti, još jedan razlog zašto je važno da sistem bude jednostavan jeste taj da sama ta jednostavnost za sobom vuče i **dobre performanse**. Sistem sa dobrim performansama je danas „must have“. Treba imati na umu da na sistemu radi više skupina ljudi, nekada istovremeno, a nekada ne, ali je važno imati u vidu da će sistem doći u situacije kada će trebati uslužiti **veliki broj korisnika**, a sve informacije koje sistem pruža idalje moraju ostati **dostupne i ažurirane u real-time-u.**

I na kraju je važno napomenuti da ne postoji sistem koji je jednom isporučen i završen je rad na njemu, pa tako ni ovaj sistem. Sistem mora da bude **nadogradiv** i spreman za izmjene kada ga je potrebno unaprijediti.

3 Karakteristike softverske arhitekture

3.1 Performanse

Sistem koji se pravi mora imati dobre performanse kao što smo već spominjali, što zbog više skupina korisnika, što zbog situacija u kojima može doći do velikog broja korisnika na istom mjestu u isto vrijeme ali i zbog mogućnosti da u real time-u prikažu dostupne i zauzete termine, trenere, rezervacije, suplemente i sl.

3.2 Upotrebljivost

Jedan od faktora zbog kojih klijenti ostaju vjerni nečemu jeste lakoća korištenja i brzina funkcionisanja aplikacije, a to je ono što želimo postići kod kreiranja ovog sistema. Bitno je da sistem bude prilagođen za korištenje svim dobnim skupinama jer je i sama usluga u realnom svijetu takva, neograničena. Pored toga što će se UI pojednostaviti kako bi bio pristupačan i lako učljiv svima, korisnici sistema (uprava, menadžment, prodavač i uposlenici) će dobiti dokumentaciju kroz koju će moći pročitati sve o načinu funkcioniranja i korištenja sistema, dok bi se članovima sistema trebao omogućiti pristup uputama koje će biti u sklopu aplikacije. Na ovaj način bi izbjegli moguću diskriminaciju po dobnoj skupini i smanjili mogućnost grešaka i stalnih pitanja.

3.3 Sigurnost

Kako sistem ima dio koji je javno dostupan, koji se prikazuje krajnjim korisnicima i dio koji nije javno dostupan uvijek postoji bojazan da bi moglo doći do curenja informacija. Iako se na sistemu ne vrše nikakve transakcije, postoji mogućnost da dođe do ozbiljnog sigurnosnog propusta koji bi bio koban za funkcioniranje fitness centra. Iz ugla dijela sistema koji nije javno dostupan, moramo osigurati da će svi dokumenti i pravni akti ostati privatni i da ni u kojem slučaju ne dođu u situaciju da budu javno prikazani. Pored toga bitno je osigurati da svi podaci nisu svima dostupni odnosno da određeni dijelovi sistema su dozvoljeni samo osobama sa određenim ulogama odnosno permisijama. Iz ugla javno dostupnog dijela sistema trebamo voditi računa o podacima koje krajnji korisnici ostavljaju u sistemu. Takvi podaci se moraju osigurati i ne dozvoliti da bilo kada procure u javnost.

3.4 Sposobnost održavanja

Obzirom da svaka aplikacija, program ili softver nakon prve verzije koja je isporučena ima još mnogo verzija koje su poboljšanje one prethodne verzije, važno je da ova aplikacija ima dobru sposobnost održavanja. Bitno je da izmjene postojeće aplikacije i nadogradnja i implementacija novih funkcionalnosti budu aktivnosti koje je jednostavno za napraviti. Naravno da je uz to važno da se aplikacija može kvalitetno istestirati kako ne bi dolazilo do toga da izmjena na jednom dijelu aplikacije utiče na cijelu aplikaciju ili na neke druge dijelove aplikacije.

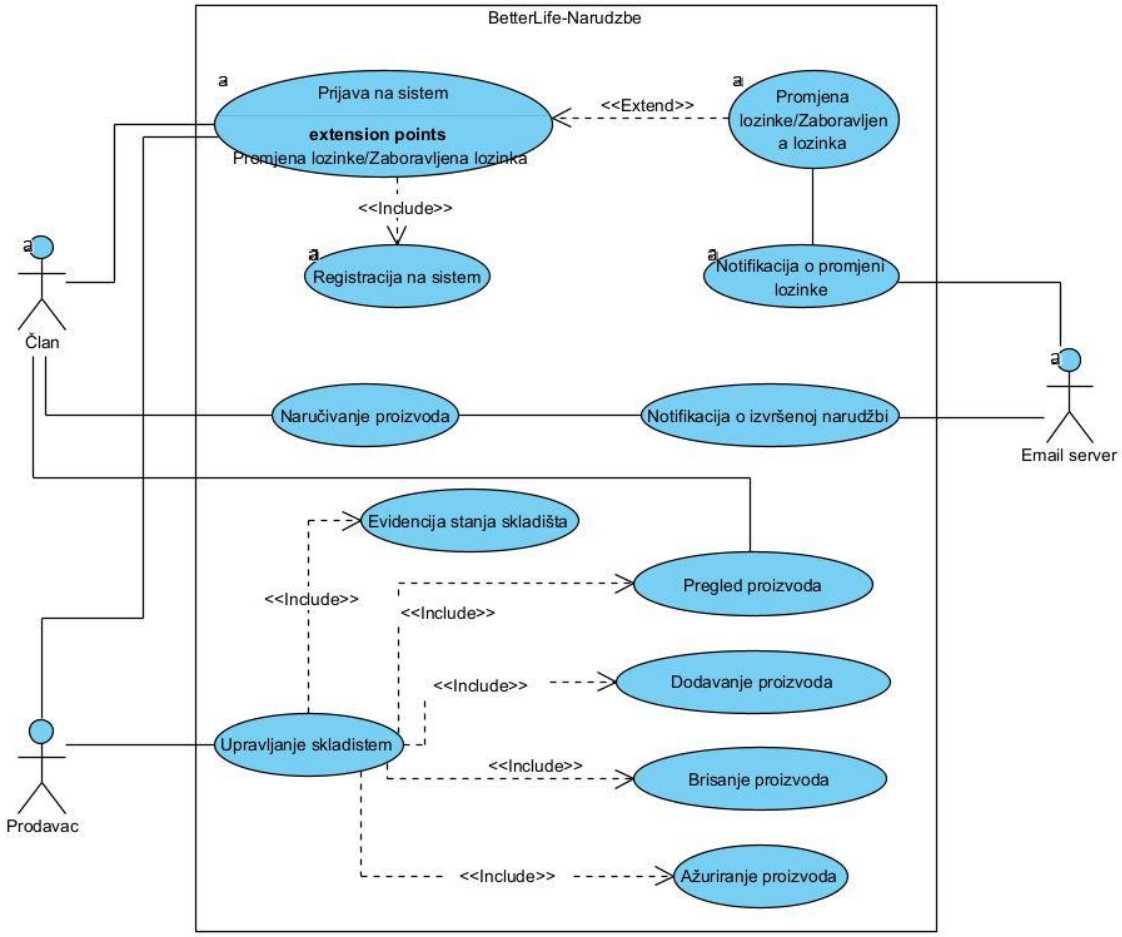
4 Značajni use-case-ovi

U svrhu boljeg razumijevanja pojedinih procesa sistema kreirana su tri use-case dijagrama i to:

1. Narudžbe
2. Funkcije uposlenika
3. Upravljanje terminima

4.1 Narudžbe

Članovi Fitness centra su glavni konzumenti na sistemu i pored ranije spomenutih opcija koje fitness centar nudi, članovi imaju opciju naručivanja artikala koje plaćaju po preuzimanju ili u fitness centru. Na slici 3 imamo dijagram koji prikazuje akcije člana i prodavač. Kao i ranije član i prodavač imaju opciju prijave na sistem i promjene lozinke, dok samo član ima mogućnost registracije. Prodavač nakon prijave dobija opcije upravljanja skladištem koje podrazumijevaju evidenciju stanja skladišta gdje može vidjeti koliko artikala ima na skladištu i kojih artikala, može dodati, brisati ili ažurirati proizvode na skladištu. Još jedna opcija koja je zajednička prodavaču i članu jeste pregled proizvoda. Član ima opciju naručivanja pri čemu se notifikacija potvrde o izvršenoj narudžbi dostavlja na obje strane, i prodavaču i članu.



Slika 3. Narudžbe – Use Case dijagram

4.2 Funkcije uposlenika

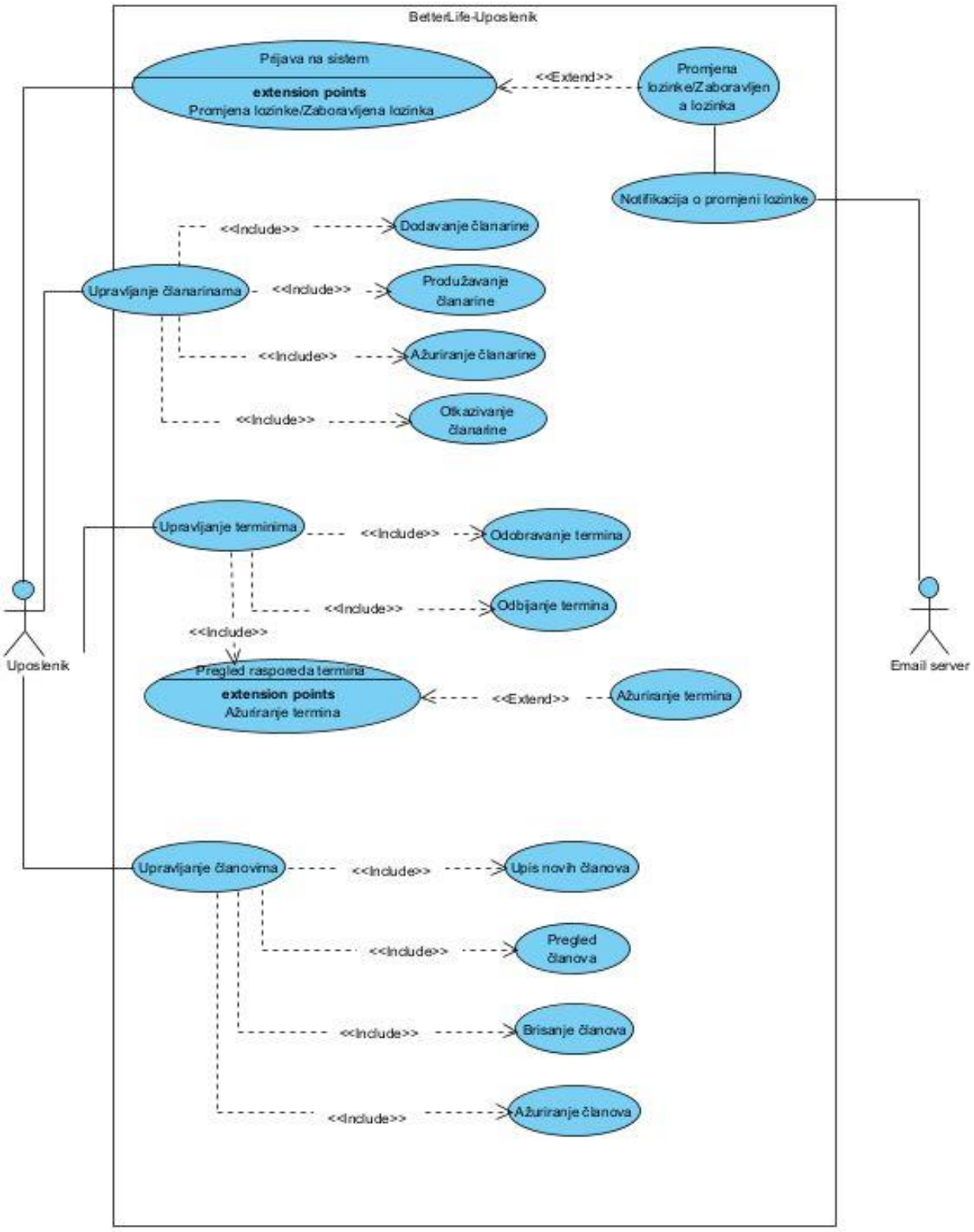
Drugi use case dijagram prikazan na slici 4 opisuje funkcije uposlenika fitness centra. Uposlenik ima najviše funkcija na sistemu pa ih je bitno istaknuti. Kao i svaki član administracijskog dijela fitness centra, tako i uposlenik nema mogućnost registracije jer se oni registriraju od strane upravnika, ali se mora prijaviti na sistem da bi imao sve opcije koje su u opisu njegovog posla. Uz prijavu ima mogućnost izmjene lozinke ukoliko je zaboravio ili samo želi promijeniti. Osnovne tri funkcionalnosti su mu :

- Upravljanje članovima
- Upravljanje članarinama
- Upravljanje terminima

Upravljanje članovima podrazumijeva da može upisivati nove članove u fitness centar, brisati članove iz sistema, ažurirati njihove informacije ili ih sve pregledati na jednom mjestu.

Upravljanje članarinama obuhvata akcije dodavanja članarina, produžavanje članarina, ažuriranje informacija članarina i otkazivanje.

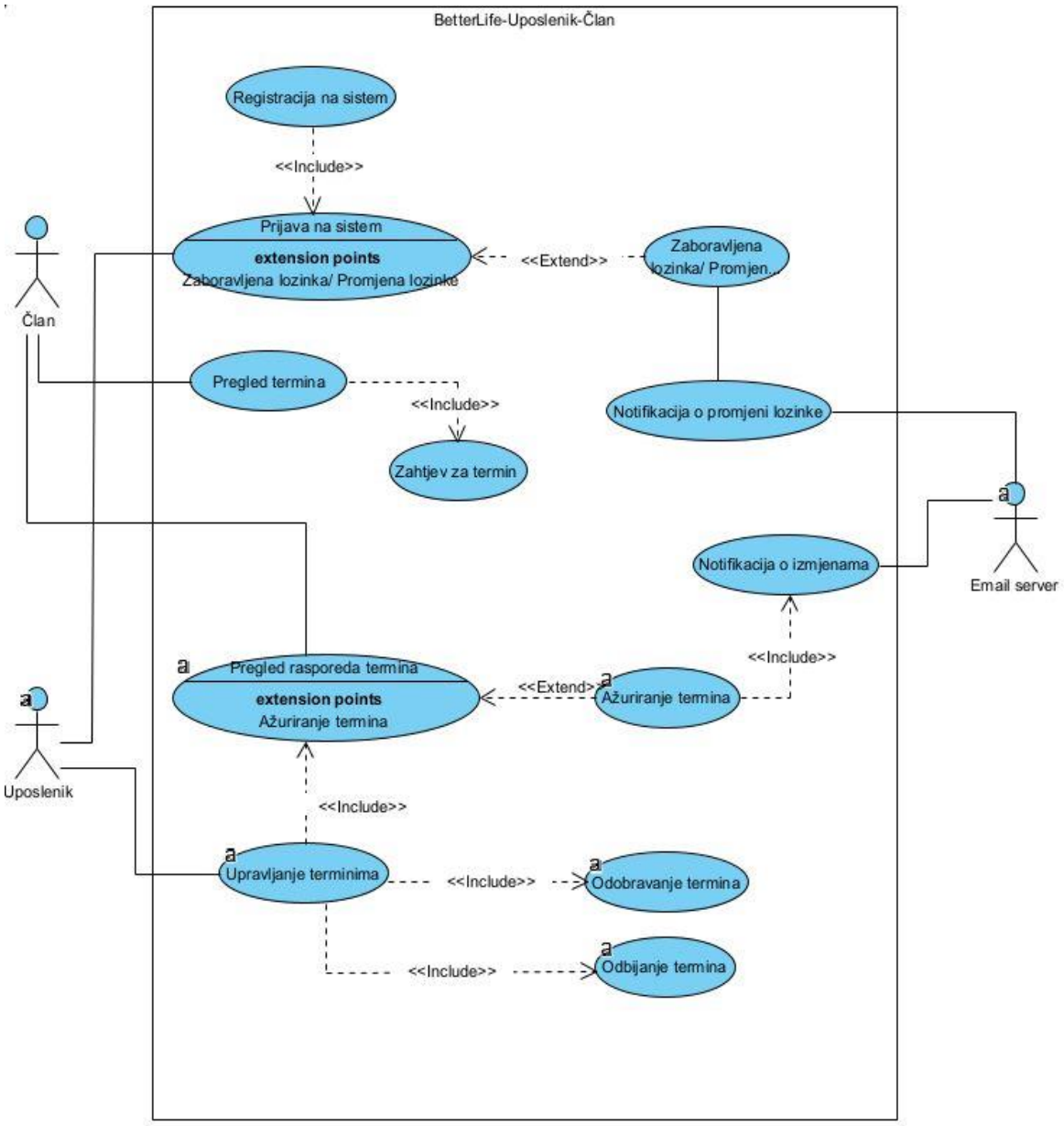
Upravljanje terminima ćemo dodatno objasniti u narednom dijagramu.



Slika 4. Funkcije uposlenika – Use Case dijagram

4.3 Upravljanje terminima

Use case dijagram sa slike 5 prikazuje odnos uposlenika fitness centra koji direktno upravlja terminima treninga i člana fitness centra koji je podnosilac zahtjeva za terminom. Na dijagramu možemo vidjeti da član ima mogućnost registracije ukoliko je novi na sistemu ili prijave na sistem. Također ima opciju promjene lozinke ukoliko je član zaboravio lozinku ili je samo želi promijeniti i pri svakoj izmjeni lozinke dobija notifikaciju putem email-a. Svaki član ima mogućnost pregleda svih dostupnih termina i slanja zahtjeva za određenim, odabranim terminom. Uposlenik s druge strane je dužan da se prijavi na sistem kako bi imao opcije za upravljanje terminima. Upravljanje terminima obuhvata pregled rasporeda već zakazanih termina, te ažuriranje istih pri čemu se šalju notifikacije članovima čiji su termini ažurirani. Druge akcije upravljanja terminima su odobravanje ili otkazivanje zahtjeva koje članovi šalju.



Slika 5. Upravljanje terminima – Use Case dijagram

5 Vrste (paterni) softverske arhitekture

U ovom poglavlju ćemo se pozabaviti **service-oriented**, **mikroservisnom** i **service-based arhitekturom**. U nastavku će svaka od njih biti podrobnije objašnjena i bit će izvučene njihove prednosti i nedostaci, a na kraju poglavlja ćemo odabrati onu koja najvišim postotkom ispunjava zahtjeva fitness centra.

5.1 Paterni i karakteristike SA

Service-oriented arhitektura, poznatija kao SOA zasniva se na reorganizaciji aplikacija u grupe funkcionalnosti koje nazivamo servisima. Osnovni koncept je da poslovne funkcionalnosti ne budu „zarobljene“ u zasebnim aplikacijama već da se stvori jedinstveni, centralni servisni sloj koji predstavlja katalog poslovnih funkcija dostupan svim aplikacijama, što omogućava ponovnu iskoristivost. U tehničkom smislu dobro skalira i ima slabo povezane servise. To znači da omogućava smanjenje za potrebe manjih sistema ili povećavanje za potrebe velikih sistema, što je dobra strana **sposobnosti održavanja** međutim ocjena iz testiranja je loša jer su to ciljevi koji su bili slabo podržani i nisu bili važni (ili čak aspirativni) ciljevi tokom razdoblja razvoja SOA arhitekture i to je minus za održavanje. **Sigurnost** je bila glavni problem još i u ranim implementacijama SOA-e. Ova arhitektura je podržala ciljeve kao što su elastičnost i skalabilnost, uprkos poteškoćama u implementaciji tih ponašanja, što je značajan doprinos na polju **upotrebljivosti**. Međutim **performanse** ove arhitekture su izuzetno loše jer se svaki poslovni zahtjev dijeli kroz ovu veliku arhitekturu.

Mikroservisna arhitektura sastoji se od skupa malih, autonomnih servisa. Ovo je distribuirana arhitektura najsitnije granularnosti u odnosu na druge dvije spomenute. Svaki servis je samostalan, ima vlastitu bazu podataka i trebao bi implementirati jednu poslovnu sposobnost. Servisi su mali i neovisni jedni o drugima, pa razvojni timovi mogu neovisno raditi na pojedinim servisima, implementirati i isporučivati ih, tako da je **sposobnost odražavanja** samih servisa na visokom nivou. Također zbog takvih osobina lakše i brže se testiraju i to je plus u održavanju**.** Ova arhitektura ima najvišu ocjenu iz elastičnosti što znači da se lako prilagođava radnom opterećenju u stvarnom vremenu i time dokazuje da joj je **upotrebljivost** bitna karakteristika koju ispunjava. Zbog svoje sitne granularnosti trpi na polju **performansi** i ovo im je najčešći problem. Da bi se odvijala komunikacija između servisa moraju se obaviti mnogi mrežni pozivi i pri svakom pristupu servisima se zahtjeva sigurnosna provjera, te na taj način se performanse arhitekture umanjuju. Obzirom da se pri svakom pozivu zahtjeva provjera **sigurnosti** u smislu autentifikacije i autorizacije za svaki servis, sigurnost je idalje izazov te zbog velikog broja mogućih servisa postoji i veći broj mjesta proboja.

Service-based arhitektura je arhitektura koja se temelji na servisima kao primarnim komponentama. Ova arhitektura je hibrid *service-oriented* i *mikroservisne* arhitekture. Neke od značajnih odlika jesu: domenski je bazirana arhitektura, ne koristi međuservisnu komunikaciju zbog dijeljenja baze podataka i koda, te to da posjeduje samodovoljne servise koji su odvojeno implementirani što omogućava odličnu ocjenu iz sfere **održavanja,** a to potvrđuju i dobre ocjene iz testabilnosti i skalabilnosti. Omogućava brze izmjene i implementacije ukoliko se promijene zahtjevi aplikacije, te testiranje kako bi se lako pronašle i izolirale greške. Elastičnost ima slabu ocjenu kod ove arhitekture, međutim ona je ostvariva iako ne u tako dobroj mjeri kao što je kod mikroservisne arhitekture, te zbog takve elastičnosti nije učinkovita ni toliko isplativa što se odražava na nešto lošijim performansama koje opet za sobom vuku i **upotrebljivost**. **Performanse** ove arhitekture čuvaju zlatnu sredinu, nisu najbolje ocijenjene ali su idalje bolje od performansi mikroservisne arhitekture i SOA arhitekture zahvaljujući krupnoj granularnosti. Modularnost koju distribuirane arhitekture posjeduju u kombinaciji sa remote pristupom servisima zahtjeva da i **sigurnost**

bude na višem nivou, svi korisnici moraju da budu autentificirani i autorizirani. Zbog krupne granularnosti, veći su servisi što znači manje mrežnog prometa čime je sistem sigurniji u odnosu na mrežu.

5.2 Ograničenja paterna

Service-oriented arhitektura ima poteškoća u osiguravanju sistema. Zbog decentralizirane prirode, podaci teku u svim smjerovima i zahtjeva zaštitu u svakom trenutku što dovodi do izazova u sferi **sigurnosti**. Još jedan nedostatak jeste i to što iz testabilnosti ima nisku ocjenu pa **sposobnost održavanja** kao jedan od bitnih ciljeva nije ispunjen u onoj mjeri u kojoj se očekuje od arhitekture. Zbog dijeljenja poslovnih zahtjeva kroz arhitekturu **performanse** trpe i ocijenjene su kao jako loše.

Najveći nedostatak **mikroservisne arhitekture** je međuservisna komunikacija koja obuhvata više poziva i stalne sigurnosne provjere čime se dovode u pitanje **performanse.** Obzirom da postoji mnoštvo servisa koji su u upotrebi tokom rada aplikacije, a s tim dolazi i veliki broj poziva svaki od tih pristupa servisima mora biti provjeren što znači da postoji i više mogućnosti za proboj i narušavanje **sigurnosti**.

Service – based arhitektura ima problem po pitanju elastičnosti koja utiče manjim dijelom na **performanse**, a značajnije na **upotrebljivost** i to znači da se slabije prilagođava opterećenju u stvarnom vremenu i veliki je problem voditi računa o odgovorima koje svaki servis zasebno mora da vraća.

5.3 Odluke i obrazloženja

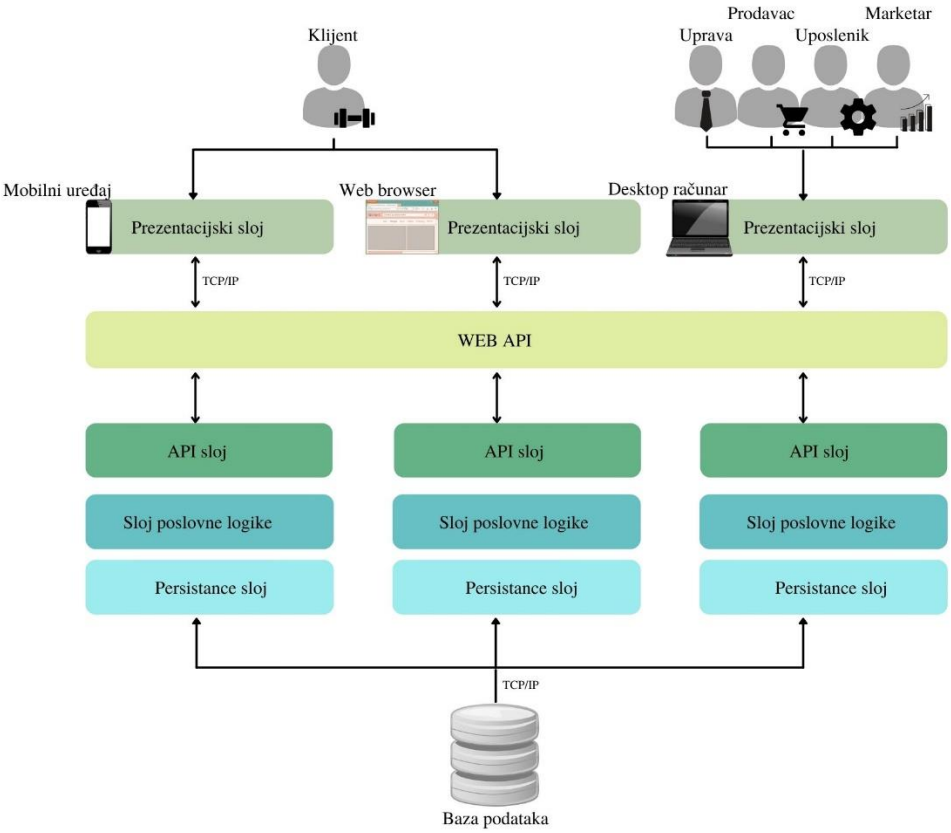
Nakon detaljne analize svake od navedenih arhitektura i u poređenju sa zahtjevima naše aplikacije odabrana je **service-based arhitektura**. Obzirom da su sve ove arhitekture distribuirane i svaka radi sa servisima kao osnovnim gradivnim jedinicama, pri izboru se vodilo računa na izabrana arhitektura svojim karakteristikama pokriva sve zahtjeve, te da niti jedan zahtjev ne trpi zbog toga što je drugi fenomenalno pokriven.

Svaka od ovih arhitektura dobro skalira pa ne bi trebalo biti značajnijih problema prilikom **održavanja** sistema. Najvažnije karakteristike aplikacije za fitness centar su sigurnost i performanse. Važno je da aplikacija u svakom momentu bude sigurna obzirom da se radi o privatnim korisničkim računima sa ličnim informacijama, te da sistem ne bude usporen zbog rada određenih funkcionalnosti u real time-u. **Sigurnost** je izazov svake distribuirane arhitekture. Service oriented arhitektura od početka razvoja se muči sa sigurnosnim propustima. S druge strane granularnost mikroservisne arhitekture doprinosi tome da je sigurnost jedan od značajnih problema i ove arhitekture. Borbu sa **performansama** igraju sve tri arhitekture, ali na tom polju mikroservisna i SOA arhitektura imaju jednake ocjene i lošije su rangirane u odnosu na service based arhitekturu. Mikroservisna i SOA sada već imaju po dvije karakteristike koje im ne idu u prilog i time su se dovele do eliminacije. Nakon sumiranja svega šta nam koja arhitektura nudi, a šta uskraćuje shvatamo da je najpogodnija service – based arhitektura koja ne briljira niti na jednom polju ali na ima i više nego dobre ocjene ukoliko izuzmemo probleme **upotrebljivosti** koji se neće osjetiti na aplikaciji ove vrste i kompleksnosti.

6 Logički pogled

Koje funkcionalnosti sistem pruža možemo uočiti kroz logički pregled sistema prikazan na slici 6.

Sistem je kreiran tako da svi servisi pristupaju jednom serveru na kojem se nalazi određena baza podataka za ovaj sistem, dakle svi su spojeni na jednu bazu podataka. Svaki servis ima višeslojnu arhitekturu koja se sastoji od API sloja, sloja poslovne logike, i persistance sloja. Korisnička sučelja putem kojeg upravnik, uposlenici, prodavači i marketari obavljaju svoj posao je web aplikacija, a korisnici za pristup sistemu mogu koristiti mobilnu ili web aplikaciju. Svaki od ovih korisničkih sučelja pristupa istom API-u.



Slika 6. Logički pogled

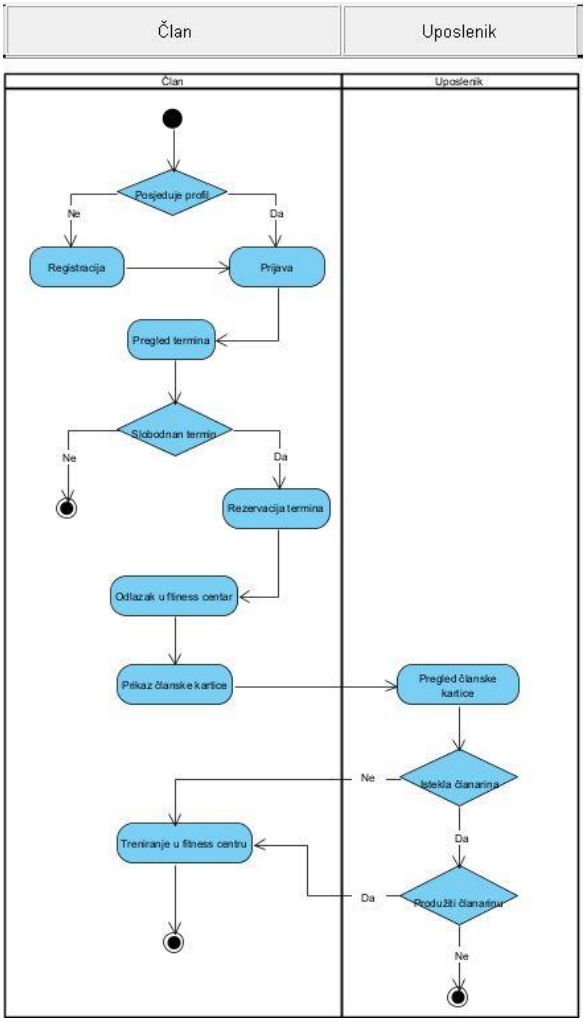
7 Procesni pogled

Za prikaz dinamičkih aspekata sistema i procesa koji su od velike važnosti koristit će se dijagram aktivnosti. Aktivnosti se sastoje od manjih akcija i prikazat ćemo samo neke, a te koje ćemo prikazati su :

1. Odlazak na trening
2. Prijava i registracija
3. Narudžbe

7.1 Odlazak na trening

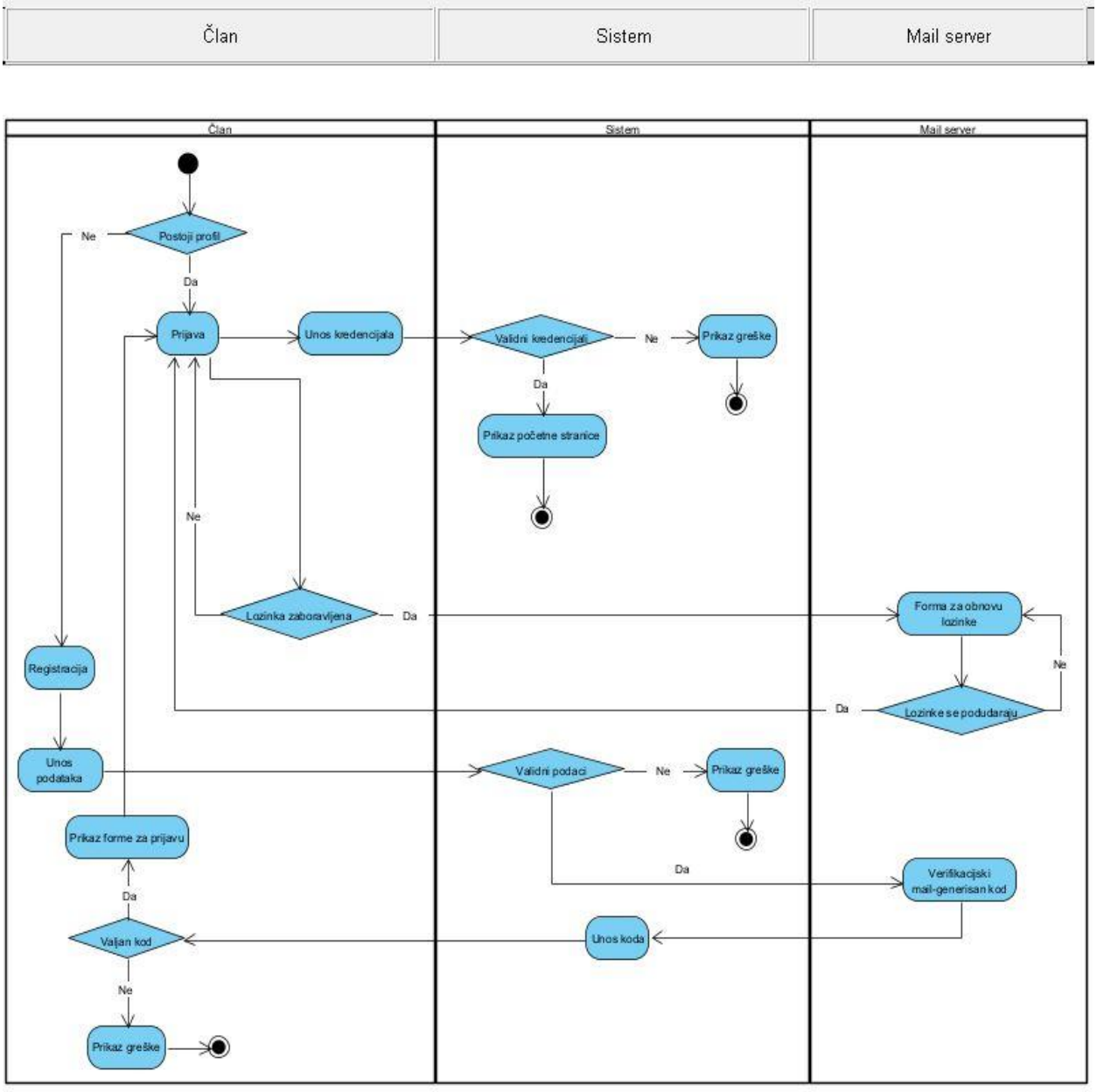
Prije nego osoba ode na trening u fitness centar potrebno je da ima svoj termin. A da bi imala svoj termin mora biti registrovana na sistemu. Nakon registracije, osoba se prijavi na sistem i ima uvid u sve termine. Ukoliko ne postoji slobodan termin, onda nije moguće ni doći na trening, a ukoliko postoji slobodan termin, isti je potrebno rezervisati. Kada član dođe u fitness centar on daje na uvid svoju člansku karticu pri čemu se provjerava validnost uplate članarine. Ukoliko članske obaveze nisu ispunjene, član ima mogućnost da nastavi trening uz uplatu ili da ne nastavlja trening. Cijeli proces je prikazan na slici 7.



Slika 7. Odlazak na trening – Dijagram Aktivnosti

7.2 Registracija i prijava

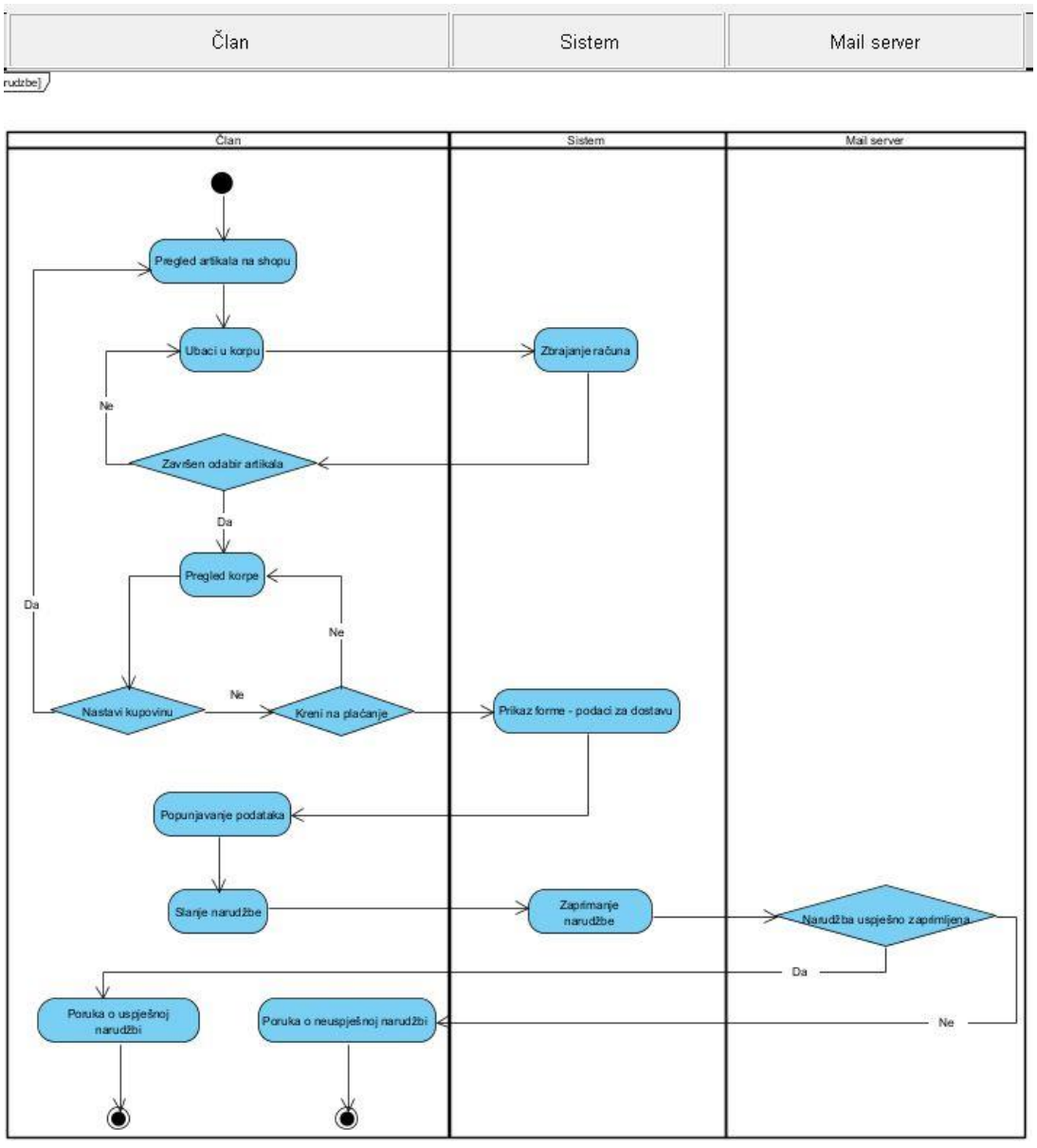
Kao na svaki sistem, tako i na ovaj potrebno je izvršiti registraciju i prijavu. Osobe koje imaju već registrovan profil cijelu proceduru završavaju jednostavnom prijavom, a ukoliko nisu registrovani onda imaju mogućnost registracije. Kod registracije se unose podaci za registraciju koje poslije unosa sistem validira i ukoliko su validni, korisnik će od mail servera dobiti verifikacijski kod kojim potvrđuje svoju email adresu i registraciju. Ukoliko kod koji korisnik unese nije isti kao onaj dobijen u email-u on dobija grešku, a ukoliko je sve validno onda ga prebacuje na formu za prijavu. Proces prijave i registracije na sistem je na slici 8.



Slika 8. Registracija i prijava – Dijagram Aktivnosti

7.3 Narudžbe

Spominjali smo da postoji i mogućnost naručivanja opreme i suplemenata. Naravno da se podrazumijeva da je član registrovan na sistem i prijavljen. Nakon prijave ima uvid u sve artikle koji su na stanju i od kojih bira one koje želi kupiti. U toku dodavanja novih artikala, sistem nakon svakog dodanog artikla sračuna vrijednost računa kojoj član svakog trenutka može pristupiti. Nakon što završi odabir artikala, može pregledati korpu i ukoliko želi nastaviti kupovati, ima tu opciju da se sa pregleda korpe vrati na shop. Ako želi kupiti te artikle onda ide dalje na plaćanje gdje sistem nudi formu u koju se unose lični podaci kao i podaci za dostavu obzirom da još uvijek nije u planu razvijati kartično online plaćanje. Nakon što se popune podaci i potvrdi narudžba, sistem zaprima narudžbu i čeka od mail servera da mu javi da je narudžba uspješno ili neuspješno zaprimljena. I u jednom i u drugom slučaju korisnik će dobiti poruku o statusu narudžbe. Tok procesa narudžbe proizvoda se može vidjeti na slici 9.



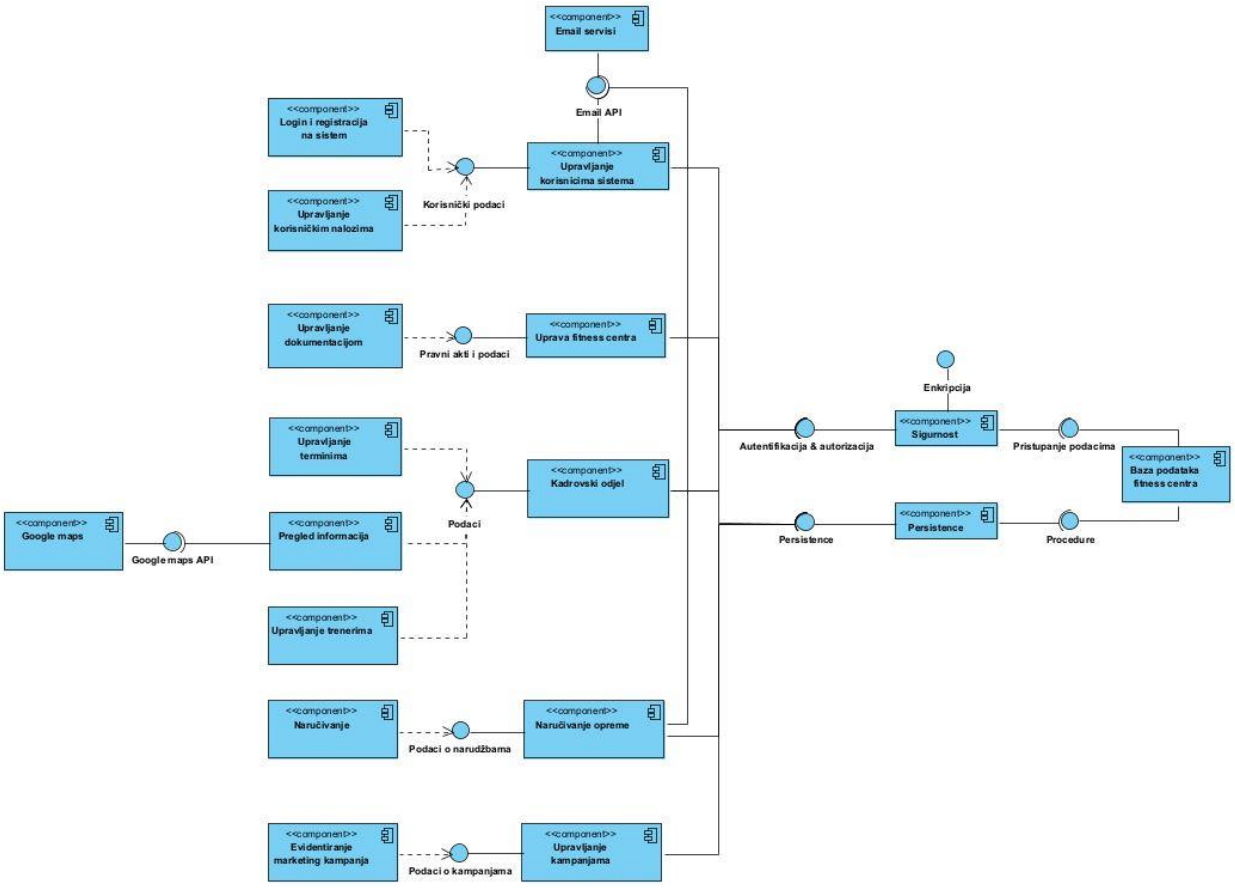
Slika 9. Narudžbe – Dijagram Aktivnosti

8 Implementacijski pogled

Prikaz arhitekturalnih odlika koje su u direktnoj vezi sa implementacijom sistema su najbolje prikazane kroz implementacijski pogled, te se u ove svrhe koristi dijagram komponenti koji je prikazan na slici 10.

U dijagramu su prikazani artifakti koji sudjeluju u gradnji kompletnog sistema, a to uključuje:

1. Bazu podataka
2. Infrastrukturu
3. Servise
4. Korisničko sučelje
5. Eksterne API-jeve



Slika 10. Implementacijski pogled – dijagram komponenti

9 Veličina i performanse

Iako ovo nije sistem koji je namijenjen da bude krovni i da posjeduje više fitness centara nego je osmišljen kao jedan sistem koji radi za sebe, on itekako ima ozbiljne zahtjeve za dobrim performansama. Već smo u nekoliko navrata spominjali da su performanse od velike važnosti jer sistem služi 5 grupa korisnika, četiri na operatorskoj strani i korisnike na klijentskoj strani. Važno je da su sve informacije uvijek ažurirane i servirane u real time-u obzirom da svaki član fitness centra želi da u bilo kojem momentu ima pravovremene informacije o statusu zauzetih/slobodnih termina u centru, te da se iste ažuriraju ukoliko on rezervira svoj termin. Isto tako je važno da se stanje skladišta ažurira prilikom narudžbe određenih artikala. Zamišljena je dokerizacija koja u svakom slučaju omogućava poboljšanje performansi i zamišljeno je da se ostavi prostora za eventualne izmjene u smislu performansi tj. da se u bilo kojem momentu može pristupiti određenom servisu i raditi na performansama istog bez da se ostatak sistema mijenja.

10 Kvaliteta

Kod izbora arhitekture se vodilo računa da se izborom pokrije najveći dio klijentskih zahtjeva. Ova arhitektura sigurno ispunjava velikim dijelom sve što je zamišljeno. Domensko particioniranje i samodovoljni servisi kao čimbenici ove arhitekture uz modularnost kao bitnu karakteristiku service – based arhitekture su od velikog značaja u bilo kojoj aplikaciji pa i u ovoj, čime se olakšava rad, pojednostavljuje i dobija na agilnosti nadogradnje, modifikacije i testiranja. Service – based arhitektura daje prostora da se nekad u nekom momentu može preći i na neku od dvije prethodno spomenute arhitekture uz određene izmjene i balansiranja. Poznata je kao kompleksna arhitektura ali njena kompleksnost nosi dobre strane a to je da povećava produktivnost u timovima, razvijaju se pouzdane aplikacije, te poboljšava brzinu izlaska na tržište.

11 Reference

1. Gašpar, D. (2021). *Materijali sa predavanja.* FIT Mostar.
2. Len Bass, P. C. (2012). *Software architecture in practice (Third edition).* Massachusetts: Addison-Wesley Professional.
3. Nepoznat. (2020). Servisno orijentirana arhitektura (SOA). *SV Group*, 1. Retrieved Decembar 01, 2021, from https://www.svgroup.hr/rjesenja/service-oriented-architecture-soa/
4. Nepoznat. (n.d.). *Microservices architecture style*. Retrieved from Microsoft: https://docs.microsoft.com/en-us/azure/architecture/guide/architecture-styles/microservices
5. Pečenac, N. (n.d.). *Mikroservisna arhitektura.* Retrieved from PDF slide: https://pdfslide.tips/documents/mikroservisna-arhitektura-2018-anac-baza-podataka-rdbms-nosql-serverska.html
6. Richards, M. (2015). *Software Architecture Patterns.* California: O'Reilly Media, Inc.
7. Richards, M. (2015, Oktobar). The Challenges of Service-Based Architecture. *Not fuff just stuff - NFJS*, 1. Retrieved Decembar 01, 2021, from https://nofluffjuststuff.com/magazine/2015/10/the_challenges_of_service_based_architecture
8. Servisno orijentisana arhitektura (SOA). (2021). *Serbian Business Systems - SBS*, 1. Retrieved Decembar 01, 2021, from https://www.sbs.rs/servisno-orijentisana-arhitektura-soa
9. M. Richards and N. Ford, *Fundamentals of software architecture*. O'Reilly Media, 2020.