

*Jesenji semestar, 2023/24*

*PREDMET: SE201 Uvod u softversko inzenjerstvo*

Projektni zadatak – Frizerski salon

Predmetni profesor: Jovana Jovic

Predmetni asistent: Bogdan Jankovic Autor: Andrija Jovanovic

1. **Izabrati jedan agilni metod i opisati ga**

Scrum agilni metod, stavke ce biti razlozene na manje i koristice se sprintovi, naravno ovaj metod nije potpun bez scrum tima I scrum mastera koji daje sustinu svemu. Zaostale stavke idu u listu cekanja, a na kraju svakog sprinta se sumira sta je u sprintu uradjeno.

1. **Murov obrazac**

ZA: Vlasnike i osoblje muških frizerskih salona KOJIMA je potrebna efikasnija evidencija o frizerima i klijentima. KREIRANA JE APLIKACIJA: BarberManager KOJA OBEZBEĐUJE: Svim zaposlenima u salonu jednostavno upravljanje evidencijom frizera, zakazivanjem tretmana, i vođenjem podataka o klijentima. ZA RAZLIKU OD: ostalih aplikacija slične prirode koje su zastarele i imaju oskudan korisnički interfejs, NAŠ PROIZVOD: omogućava vlasnicima, menadžerima, i frizerima lakše praćenje frizerskog osoblja, dostupnosti i rasporeda tretmana, kao i kompletne evidencije o mušterijama.

ŠTA: Aplikacija koja olakšava vođenje evidencije o frizerima, zakazivanje tretmana, praćenje dostupnosti frizera i vođenje evidencije o klijentima.

KO: Proizvod je namenjen vlasnicima, menadžerima, i frizerima u muškim frizerskim salonima.

ZAŠTO: Aplikacija donosi sveže promene i umanjuje potrebno tehničko znanje uz jednostavan korisnički interfejs.

1. **Napisati funkcionalne zahteve**

Korisnik sistema ima mogućnost dodavanja novog frizera unosom potrebnih podataka (ime, prezime, specijalnosti, kontakt informacije) putem forme.

Korisnik sistema ima mogućnost uklanjanja postojećeg frizera iz evidencije.

Korisnik sistema ima mogućnost ažuriranja podataka o frizerima.

Korisnik sistema ima mogućnost zakazivanja tretmana izborom frizera i tipa tretmana.

Korisnik sistema ima mogućnost otkazivanja ili izmene postojećeg zakazanog tretmana.

Korisnik sistema ima mogućnost dodavanja novih klijenata unosom osnovnih informacija (ime, prezime, kontakt informacije) putem forme.

Korisnik sistema ima mogućnost pregleda zarade za svakog frizera posebno.

1. **Napisati nefunkcionalne zahteve**

Performanse: Sistem mora da ima optimalnu brzinu otvaranja do 3 sekunde za 500 poseta u isto vreme, kao i optimalne performanse za do 500 korisnika.

Skalabilnost: Sistem mora da podrzi do 1000 korisnika u isto vreme bez znatnih smanjenja performansi.

Portabilnost: Sistem mora da bude lak za prelazak sa jednog operativnog sistema na drugi, kao i na druge uredjaje.

Kompatibilnost: Sistem mora da bude kompatibilan sa svim verzijama operativnih sistema, pogotovo na tablet racunarima i telefonima, za koje je prvenstveno namenjen.

Pouzdanost: Softver treba da radi u otprilike 95% slucajeva bez greske, ako je moguce i vise.

Odrzivost: Kada padne (ukoliko padne) sistemu ne sme da treba vise od 10 minuta da se ponovo podigne.

Dostupnost: Sistem bi trebao da bude dostupan korisnicima bar 98% vremena dnevno, sto je 84672 od mogucih 86400 sekundi dnevno.

Sigurnost: Sigurnost mora da bude na nivou gde svako ko radi u na sistemu moze da vidi samo svoje dozvoljene role, ali ne treba da bude na tolikom nivou da predstavlja neki dugacak zadatak.

Lokalizacija: Sistem, takav kakav jeste odnosno kreiran za evropsko trziste treba da ima datum u formatu yyyy/mm/dd.

1. **Napisati od pet do sedam korisničkih priča za svaku grupu korisnika sistema.**

Kao administrator sistema, želim da imam mogućnost dodavanja novih frizera unosom potrebnih podataka (ime, prezime, specijalnosti, kontakt informacije) putem forme, kako bih mogao ažurirati evidenciju zaposlenih.

Kao administrator sistema, želim da imam mogućnost uklanjanja postojećeg frizera iz evidencije, kako bih mogao održavati tačne podatke o zaposlenima.

Kao administrator zelim da imam veliku odzivnost sistema, od preko 90% dnevno.

Kao administrator sistema, želim da imam mogućnost pregleda zarade za svakog frizera posebno, kako bih mogao analizirati individualne performanse.

Kao administrator sistema zelim da imam sigurnost koriscenja softvera.

Kao korisnik, želim da imam mogućnost ažuriranja svojih kontakt podataka i preferencija, kako bih mogao osigurati tačne informacije za zakazivanje.

Kao korisnik, želim da imam mogućnost pregleda cena usluga, kako bih mogao planirati svoje troškove.

Svaki korisnik sistema treba da ima mogucnost pretrage frizera po tome koji su trenutno dostupni.

Kao klijent, želim da imam mogućnost pregleda istorije svojih tretmana, kako bih mogao pratiti koje usluge sam koristio.

Kao posetilac sistema zelim da performanse sajta (brzina) uvek bude prihvatljiva.

1. **Kreirati od tri do pet Persona.**

Marko Jovanović, 35 godina, vlasnik je muškog frizerskog salona. Njegovo obrazovanje jeste srednja frizerska škola I poseduje sertifikat za poslovni menadžment. Marko radi kao vlasnik salona već 12 godina. Iskusanje u vođenju poslova I upravljanju osobljem. Ima osnovno poznavanje računara I poslovnog softvera. Koristi tehnologiju za upravljanje terminima I komunikaciju s klijentima.

Nikola Petrović, 28 godina, frizer je u frizerskom salonu. Završio je srednju frizersku školu. Njegovo radno iskustvo je već 7 godina. Specijalizovan je za modern frizure I brijanje. Dobro se snalazi sa mobilnim aplikacijama I osnovnim poslovnim softverom. Njegov cilj je da pruži uslugu klijentima, imajući uvid u istoriju tretmana klijenata I efikasno upravljati svojim rasporedom. Nikola nije dovoljno informisan o poslovnim aspektima salona kao što su finansije i marketing, pa mu je potrebna pomoć vlasnika ili menadžera salona kako bi bolje razumeo kako njegov rad utiče na celokupno poslovanje.

Ivan Rašić, 22 godine, recepcioner u muškom frizerskom salonu. Završio je srednju ekonomsku školu I trenutno je student na Fakultetu organizacionih nauka. Radi kao recepcioner godinu dana. Ima iskustva u radu s ljudima I upravljanju terminima. Dobro se snalazi sa računarima, koristi poslovne aplikacije za zakazivanje I vođenje evidencije. Želi osigurati tačno I brzo zakazivanje tretmana, efikasno upravljati komunikacijom s klijentima I podržavati frizere u njihovom radu. Ivan nije informisan o frizerskim tehnikama koje frizeri pružaju, pa mu često treba pomoć frizera da bi tačno i adekvatno zakazao tretmane za klijente.

1. **Za svaku personu kreirati od jedan do tri scenarija.**

**Marko Jovanović – Vlasnik muškog frizerskog salona**

Marko želi da optimizuje rad svog salona i smanji čekanje klijenata. On želi da ima uvid u trenutno zauzetost frizera kako bi mogao pravilno da planira termine i rasporede. Razgovara sa Ivanom koji mu potvrđuje da aplikacija BarberManager omogućava pregled trenutne dostupnosti frizera, što će omogućiti Marku da efikasnije upravlja radnim vremenom salona i smanji čekanje klijenata.

Marko želi da prati zaradu svakog frizera pojedinačno kako bi mogao da analizira njihove performanse i motiviše ih na bolje rezultate. Razgovara sa Nikolom koji mu pokazuje kako može koristiti aplikaciju BarberManager da lako generiše izveštaje o zaradi za svakog frizera, čime Marko može identifikovati najbolje performanse i raditi na motivaciji ostatka tima.

Marko želi da osigura da svi klijenti dobiju kvalitetnu uslugu, pa želi uvid u istoriju tretmana svakog klijenta. Razgovara sa Ivanom koji mu objašnjava kako aplikacija BarberManager omogućava praćenje istorije tretmana klijenata, što omogućava Marku da personalizuje uslugu i poveća zadovoljstvo klijenata.

**Nikola Petrović – Frizer u muškom frizerskom salonu**

Nikola želi da svojim klijentima pruži personalizovanu uslugu i ima uvid u istoriju njihovih prethodnih tretmana. Razgovara sa Ivanom koji mu pokazuje kako aplikacija BarberManager omogućava pregled istorije tretmana klijenata, što omogućava Nikoli da pruži uslugu prilagođenu svakom klijentu.

Nikola želi da upravlja svojim rasporedom kako bi izbegao preklapanje termina i osigurao da ima dovoljno vremena za svakog klijenta. Razgovara sa Markom koji mu objašnjava kako može koristiti aplikaciju BarberManager za efikasno planiranje termina, što će pomoći Nikoli da bolje organizuje svoj radni dan.

**Ivan Rasić – Recepcioner u muškom frizerskom salonu**

Ivan želi da osigura brzo i tačno zakazivanje tretmana kako bi smanjio čekanje klijenata i povećao njihovo zadovoljstvo. Razgovara sa Markom koji mu pokazuje kako aplikacija BarberManager omogućava jednostavno zakazivanje i praćenje termina, čime Ivan može efikasno upravljati rasporedom salona.

Ivan želi da ima uvid u specifične usluge koje frizeri pružaju kako bi mogao da adekvatno informiše klijente i zakazuje odgovarajuće tretmane. Razgovara sa Nikolom koji mu objašnjava sve frizerske usluge i tehnike, a Ivan koristi aplikaciju BarberManager da precizno unosi i ažurira informacije o uslugama.

**8. Izabrati od jedan do tri stila softverske arhitekture, opisati i dijagramom**

**predstaviti izabrane modele rešenja softverske arhitekture za Vaš sistem.**

**Opisati prednosti i mane tih softverskih arhitektura.**

Monolitna softverska arhitektura aplikacije i softverska arhitektura mikroservisa.

Monolitna softverska arhitektura:

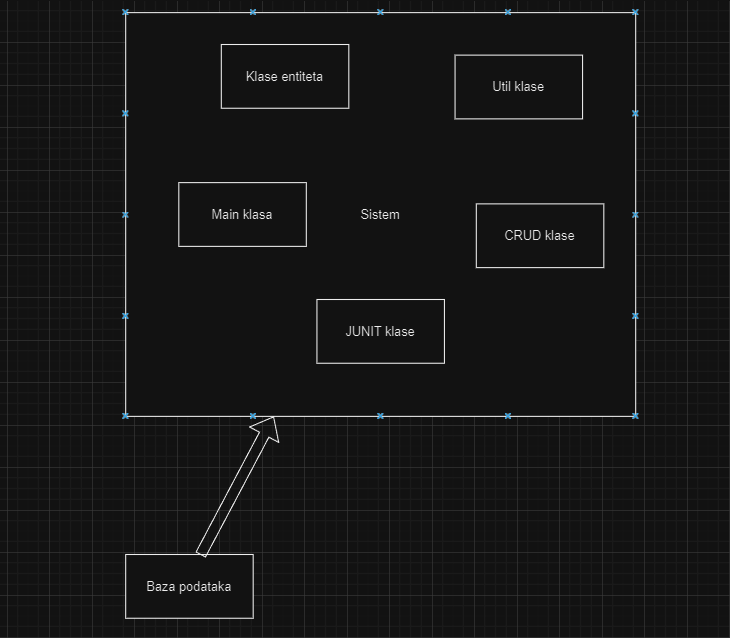
Monolitna softverska arhitektura jeste arhitektura na koju tipicno mislimo kada mislimo o nekoj aplikaciji i ona nam je najbliza. Kao takva, mnogo je laksa za implementaciju.

Prednosti:

Monolitna softverska arhitektura ima manji broj resursa neophodan za funkcionisanje, upravo jer aplikacija deli jednu bazu. Aplikacija je prirodnija ljudima pa je i laksa za pisanje i konstruisanje, jer ne zahteva neke kompleksne nacine implementacija i povezivanja funkcionalnosti – sve je povezano u jednu robusnu aplikaciju.

Mane:

Monolitna softverska arhitektura znaci da cela aplikacija deli jednu bazu. Problem kod toga jeste da ukoliko dodje do nekog proboja baze, sve informacije i podaci bivaju kompromitovani, te je cela aplikacija i sistem ostecen. Takodje skalabilnost jeste ogranicena na ovaj nacin, retko je moguce implementirati funkcionalnost iz jedne u drugu aplikaciju. Upravljanje promenama je teze, funkcionalnosti su usko povezane (cak i funkcionalnosti koje nisu povezane) pa je samim tim menjanje sistema mnogo teze i nosi sa sobom promene celog sistema, i za najmanje promene, dolazi do spageti-koda – koda koji na duze vremenske staze menja mnogo programera, ogranicenog iskustva ili vestina, uz nedostatak konvencija pa samim tim to dovodi do koda koji lici na “spagete” – sve je upleteno i nema neku strukturu.



Arhitektura mikroservisa:

Arhitektura mikroservisa podrazumeva razlaganje funkcionalnosti na manje cinioce – gde se cela aplikacija na kraju sastoji od mnostvo mikroservisa koji kasnije mogu biti upotrebljeni na drugim aplikacijama.

Prednosti:

Arhitektura mikroservisa podrazumeva da se svaka funkcionalnost implementira posebno u sistem – svaka funkcionalnost ima svoju bazu podataka (mi moramo odluciti koliko je mala (odnosno velika) funkcionalnost i baza podataka koja prati istu). Funkcionalnosti se kasnije mogu lako implementirati u druge aplikacije, nove funkcionalnosti se lakse implementiraju u sistem i stare se lakse brisu, bez da sa sobom vuku ostale funkcionalnosti i kompromituju ih.

Mane:  
Arhitektura mikroservisa nije intuitivan nacin pisanja softvera i aplikacija, te je kao takva jako teska i kompleksna za implementaciju – mnogo teza nego monolitna arhitektura softvera. Troskovi su mnogo veci - ne moze svako raditi na ovaj nacin te se zahtevaju iskusniji inzenjeri softvera, troskovi odrzavanja baze su mnogo veci jer svaka funkcionalnost ima svoju bazu podataka, sto moze da bude mnogo baza podataka koje se posebno azuriraju i administriraju. Testiranje je dugo, jer zahteva da se svaka funkcionalnost posebno testira. Takodje upravljanje na kraju moze postati jako kompleksno i tesko.

**9. Kreirati 10-15 klasa.**

**10. Kreirati bazu podataka i logiku u Javi za povezivanje na istu.**

**11. Izdvojiti komponente sistema koje mogu biti ponovo upotrebljene.**

Komponente koje mogu biti ponovo upotrebljene jesu:

Konektor baze podataka – Klasa koja povezuje aplikaciju sa bazom podataka, uz male izmene moze biti ponovo upotrebljena u drugom sistemu.

Klase za rad sa izuzecima – Klase koje vrse rukovanje izuzecima mogu opet biti upotrebljene u drugim sistemima.

Klase entiteta – Za slican sistem, klase koje su susti entiteti aplikacije mogu ponovo biti upotrebljeni u nekom drugom sistemu.

**12. Izdvojiti delove sistema koji mogu biti primer računarstva u oblaku i navesti**

**koji bi to tip bio (IaaS, SaaS, PaaS). Takođe, izdvojiti one koji mogu biti**

**zamenjeni već postojećim rešenjima, kao i one delove sistema za koje**

**računarstvo u oblaku ne bi bilo najbolje rešenje (opsati zašto).**

SaaS – Software as a service bi mogao biti ceo sistem, implementacija bi se mogla primeniti na bilo koji drugi sistem frizerskog salona (uz izmenu baze). Mogao bi funkcionisati kao one-time buy ili kao subscriber servis – sa mesecnom pretplatom.

IaaS – Takodje bi sistem mogao da funkcionise sa IaaS, gde bi administratori drugih sistema imali potpunu kontrolu nad vec postojecim sistemom.

PaaS – Ne bi bilo najbolje resenje jer je sistem prvenstveno namenjen administraciji baze i samom frizerskom salonu – nije namenjen korisnicima toliko.

**13. Izdvojiti komponente sistema koji mogu biti implementirani kao**

**mikroservisi sa odvojenim bazama podataka.**

Klase entiteta – Svaka klasa entiteta bi mogla da ima svoju posebnu bazu podataka.

CRUD – svaki CRUD moze biti izdvojen kao mikroservis, mada bi bili povezani sa svojom bazom podataka o kojoj je rec.

BruteForce napad – Moze biti izdvojen kao mikroservis, mada opet nema potrebe za bazom podataka.

**14. Na Vašem sistemu demonstrirati i dokumentovati Brute Force ili SQL**

**injection napad.**

**15. Navedite i opišite atribute kvaliteta Vašeg sistema.**

Jednostavnost, siguran unos

**16. Primeniti tri koncepta pouzdanog programiranja:**

* Validacija input-a
* Šabloni (Pattern)
* Regularni izrazi (RegEx)

**17. Kreirati od pet do deset scenarija testiranja (tabela je data u prilogu).**

**18. Na osnovu kreiranih scenarija testiranja izvršiti Junit testiranje na Vašem**

**sistemu.**

**19. Ceo proces planiranja i implementacije softvera ispratiti koristeći alat**

**Trello.**

<https://trello.com/invite/b/VHaDOpwO/ATTIc7379353b4e740f7553e7d90c534359376CC08A8/se201-pz>

**20. Pratiti razvoj projektnog zadatka verzioniranjem na Git-u.**