SE201-Projekat



iBioskop

Lazar Nikolic 5396

Contents

[1. Odabrati jedan agilni metod koji će biti korišćen pri razvoju projektnog zadatka. Opisati izabrani model, objasniti zašto ste se odlučili za njega I primeniti model na Vašem projektu. 2](#_Toc156431231)

[2. Koristeći Murov obrazac kreirati viziju proizvoda. 2](#_Toc156431232)

[3. Napisati spisak funkcionalnih zahteva. 3](#_Toc156431233)

[4. Napisati nefunkcionalne zahteve. 3](#_Toc156431234)

[5. Napisati od pet do sedam korisničkih priča za svaku grupu korisnika sistema. 3](#_Toc156431235)

[6. Kreirati od tri do pet Persona. 4](#_Toc156431236)

[7.Za svaku personu kreirati od jedan do tri scenarija. 5](#_Toc156431237)

[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. 6](#_Toc156431238)

[9. Kreirati 10-15 klasa. 9](#_Toc156431239)

[10. Kreirati bazu podataka i logiku u Javi za povezivanje na istu. 9](#_Toc156431240)

[11. Izdvojiti komponente sistema koje mogu biti ponovo upotrebljene. 9](#_Toc156431241)

[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). 9](#_Toc156431242)

[13. Izdvojiti komponente sistema koji mogu biti implementirani kao mikroservisi sa odvojenim bazama podataka. 10](#_Toc156431243)

[14. Na Vašem sistemu demonstrirati i dokumentovati Brute Force ili SQL injection napad. 11](#_Toc156431244)

[15. Navedite i opišite atribute kvaliteta Vašeg sistema. 11](#_Toc156431245)

[16. Primeniti tri koncepta pouzdanog programiranja: 11](#_Toc156431246)

[17. Kreirati od pet do deset scenarija testiranja (tabela je data u prilogu). 11](#_Toc156431247)

[18. Na osnovu kreiranih scenarija testiranja izvršiti Junit testiranje na Vašem sistemu. 12](#_Toc156431248)

[19. Ceo proces planiranja i implementacije softvera ispratiti koristeći alat Trello 12](#_Toc156431249)

[20. Pratiti razvoj projektnog zadatka verzioniranjem na Git-u. 12](#_Toc156431250)

1. Odabrati jedan agilni metod koji će biti korišćen pri razvoju projektnog zadatka. Opisati izabrani model, objasniti zašto ste se odlučili za njega I primeniti model na Vašem projektu.

Za agilni metod razvijanja softvera bih koristio metodu “Scrum”.  
Scrum je metoda u kojoj se na pocetku svakog ciklusa donose odluke o tome koje svojstvo softvera treba biti prioritetno, kako bi ih trebalo razviti I sta svaki clan tima treba tacno da radi.

U “SCRUM” metodi postoje dve kljucne uloge a to su :

* Vlasnik proizvoda je clan tima koji je odgovoran da identifikuje karakteristike I atribute proizvoda. On vrsi pregled odradjenog posla I ucestvuje u testiranju prozivoda.
* Scrum Master je timski trener koji vodi tim.

Prema ovoj metodi razvoja, softver bi trebao biti razvijen u nizu procesa koji se nazivaju “sprintovi”. Sprint je aktivnost za koju je tacno odredjen vremenski okvir izvrsenja. Obicno traje od dve do cetiri nedelje I tokom svakog sprinta, tim ima dnevne sastanke u Kojima se pregleda sta je odradjenog do tada I kako bi se dogovorili oko toga sta treba da se uradi tog dana.

**Timski Rad:** Scrum podstiče timski rad i saradnju između članova tima, što je važno za uspešan razvoj aplikacije.

**Opis Scrum-a:**

**Timovi:** Tim se organizuje u interdisciplinarne timove koji rade zajedno na različitim aspektima razvoja.

**Sprintovi:** Razvoj se organizuje u sprintovima, obično dvonedeljne iteracije, tokom kojih se završavaju određene funkcionalnosti.

**Product Owner:** Odgovoran je za vođenje Product Backlog-a, prioritizaciju zadataka i komunikaciju sa timom.

**Scrum Master:** Pruža podršku timu, rešava prepreke i pomaže u primeni Scrum principa.

**Sastanci:** Redovni sastanci uključuju Sprint Planning, Daily Standup, Sprint Review i Sprint Retrospective.

1. Koristeći Murov obrazac kreirati viziju proizvoda.

Korisnik

ZA korisnike bioskopa KOJI zeli da poboljsa iskustvo korisnika, kreirana je aplikacija “iBioskop” KOJA olaksava korisnicima da rezervisu karte,provere dostupnost mesta bioskopa za zeljenu projekciju I da vide koji filmovi se projektuju u tog dana. ZA RAZLIKU od tradicionalnog sistema za rezervaciju I proveru aktuelnih projekcija, NAS PROIZVOD olaksava korisnicima da sve to urade bez obaveze dolaska u sam bioskop.

Podrska

ZA zaposlene u korisnickoj podrsci samog bioskopa, KOJI zeli da poboljsa iskustvo korisnika I samih zaposljenih u korisnickoj podrsci, kreiran je softver “iBioskop”, KOJI olaksava rad samih zaposlenih I unapredjuje korisnicko iskustvo I olaksava resavanje problema korisnika. ZA RAZLIKU OD tradicionalnog sistema za resavanje problema, NAS PROIZVOD pruza korisnicku podrsku 24/7 , koja ne zahteva dolazak do samog bioskopa, vec se sve resava preko softvera.

Radnik

ZA zaposlene u bioskopu, KOJI zeli da olaksa rad zaposlenih I poboljsa iskustvo svojim korisnicima, kreirana je aplikacija “iBioskop”, KOJA olaksava rad svojih zaposlenih I poboljsava korisnicko iskustvo samog bioskopa. Za RAZLIKU OD tradicionalnog sistema, zaposleni ne moraju vrsiti svaku rezervaciju I proveravati klijentima kada je projekcija samog filma jer ce klijenti biti u mogucnosti da to urade sami preko aplikacije, sto ce naknadno olaksati posao samih zaposlenih.

1. Napisati spisak funkcionalnih zahteva.
2. Softver mora imati intuitivan korisnicki interfejs koji prikazuje filmove u tabelarnom obliku , koji ce korisnicima pruziti lako rezervisanje karata, prijatno iskustvo I olaksati koriscenje samog softvera.
3. Softver mora imati login I register za pristup softveru preko osnovnih podataka korisnika.
4. Softver mora prikazati dostupne projekcije filmova sa informacijama o filmovima.
5. Korisnik mora imati mogucnost da ocenjuje filmove.
6. Korisnik mora imati mogucnost da napise recenziju.
7. Zaposleni mora imati mogucnost da pristupi bazi podataka, izmeniti podatke o projekciji.
8. Napisati nefunkcionalne zahteve.
9. Dostupnost - Softver mora biti dostupan 24/7.
10. Skalabilnost- Softver mora biti u mogucnosti da se lako prosiri u slucaju povecanja korisnika istog.
11. Sigurnost - Svi podaci klijenata I zaposlenih u bazi podataka moraju biti sifrovani I nedostupni neautorizovanim licima.
12. Pouzdanost - Sve funkcije softvera moraju izvrsavati zamisljene zadatke.
13. Odrzivost - Kod mora biti pregledan, klase moraju biti jasno definisane radi sto lakse odrzivosti.
14. Napisati od pet do sedam korisničkih priča za svaku grupu korisnika sistema.
15. Kao korisnik, zelim da imam mogucnost da ostavim recenziju filma.
16. Kao korisnik, zelim da imam mogucnost da ocenim film.
17. Kao korisnik, zelim da imam mogucnost da kreiram nalog, kome mogu da pristupim I rezervisem kartu za zeljenu projekciju.
18. Kao zaposleni, zelim da budem u mogucnosti da vidim koliko karata je rezervisano za zeljenu projekciju.
19. Kao korisnicka podrska u sistemu, zelim da korisnicima automatski budu prikazana par najcescih pitanja sa mogucnosi prikaza odgovora na odabrano pitanje.
20. Kreirati od tri do pet Persona.

Persona broj 1 – Korisnik

Marija Nikolić, 30 godina, je marketing menadžer u Beogradu. Marija voli da provodi slobodno vreme gledajući filmove i često posećuje bioskope sa prijateljima i porodicom. Ima želju da bude u toku sa najnovijim filmskim premijerama i da pronađe zanimljive filmove za gledanje.

Marijin glavni cilj je da koristi aplikaciju koja će joj omogućiti lako rezervisanje karata za željene projekcije, kao i da pročita recenzije i ocene drugih korisnika kako bi donela odluku o izboru filma. Problemi sa kojima se Marija suočava jesu dugački redovi za kupovinu karata i neinformisanost o dostupnim projekcijama i filmovima.

Korišćenjem ovakve aplikacije, Marija će biti u mogućnosti da brzo i jednostavno rezerviše karte, pročita recenzije i ocene filmova, što će joj omogućiti prijatno i efikasno iskustvo gledanja filmova u bioskopu. Glavna motivacija Marije je da maksimalno iskoristi svoje vreme i da uvek bude informisana o aktuelnim filmovima i projekcijama.

Persona broj 2 – Zaposleni u bioskopu

Lazar Nikolić, 38 godina, je menadžer u bioskopu u Novom Sadu.Lazar ima dugogodišnje iskustvo u upravljanju bioskopom i želi da pruži najbolje moguće iskustvo posetiocima. Njegov glavni cilj je da ima uvid u broj rezervacija za svaku projekciju kako bi efikasno upravljao kapacitetom i osigurao dobar doživljaj za sve posetioce.

Lazar se često suočava sa problemom nedostatka informacija o broju rezervacija za određene projekcije, što otežava planiranje i organizaciju rada. Korišćenjem softvera koji pruža sve potrebne informacije o rezervacijama i projekcijama, Nenad će moći da efikasno upravlja radom i osigura zadovoljstvo posetilaca. Glavna motivacija Nenada je da poboljša korisničko iskustvo i da maksimalno iskoristi resurse bioskopa.

Persona broj 3 – Korisnička podrška

Milica Jovanović, 25 godina, radi u korisničkoj podršci za softver "iBioskop". Milica je veoma komunikativna i želi da pruži najbolju moguću podršku korisnicima. Njena glavna uloga je da odgovara na pitanja korisnika i rešava eventualne probleme sa kojima se korisnici suočavaju.

Milica često dobija upite o dostupnim projekcijama, rezervacijama i tehničkim pitanjima vezanim za korišćenje softvera. Njen problem je što često mora da pretražuje informacije na više mesta kako bi pružila tačne odgovore korisnicima. Korišćenjem sistema koji automatski prikazuje najčešće postavljana pitanja i odgovore, Milica će moći da brže i efikasnije pruži podršku korisnicima. Glavna motivacija Milice je da pruži vrhunsku uslugu korisnicima i reši sve njihove nedoumice što je brže moguće.

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

1. Scenario - Korisnik

Marija, posle napornog radnog dana, želi da provede veče u bioskopu sa prijateljima. Ulazi u aplikaciju "iBioskop" i otkriva da je omiljeni film koji želi da vidi dostupan za rezervaciju. Brzo i lako rezerviše karte za sebe i prijatelje, uz dodatno čitanje nekoliko recenzija kako bi bila sigurna da je izabrala pravu projekciju za veče. Marija je zadovoljna što je sve obavila brzo i jednostavno, omogućavajući joj da se fokusira na uživanje u filmu.

2. Scenario - Zaposleni u bioskopu

Lazar, menadžer bioskopa, dolazi rano ujutru da pripremi repertoar za narednu nedelju. Pristupa softveru "iBioskop" da pregleda statistiku rezervacija i popularnost filmova. Na osnovu ovih informacija, Lazar odlučuje da promeni raspored nekih projekcija kako bi bolje odgovarao potrebama posetilaca. Takođe, koristi softver da pošalje obaveštenja o novim projekcijama korisnicima koji su se pretplatili na newsletter bioskopa, povećavajući tako prodaju za predstojeći vikend.

3. Scenario - Korisnička podrška

Milica, kao član korisničke podrške, prima poziv od korisnika koji ima problem sa svojom rezervacijom. Uzima informacije o korisnikovom problemu i brzo koristi softver "iBioskop" da proveri detalje rezervacije. Nakon što identifikuje problem, Milica daje jasne i precizne instrukcije korisniku kako da ga reši. Korisnik je zahvalan na brzoj i efikasnoj pomoći, a Milica zatvara slučaj i nastavlja da pruža podršku sledećem korisniku.

1. 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.

Slojna arhitektura deli aplikaciju na slojeve koji svaki ima specifičnu odgovornost. U tipičnom scenariju, postoji prezentacijski sloj (UI), poslovni sloj (business logic), sloj pristupa podacima (data access), i sloj za skladištenje podataka (database).

Dijagram:

+---------------------------- +

| Presentation Layer |

+----------------------------+

| Business Layer |

+----------------------------+

| Data Access Layer |

+----------------------------+

| Database Layer |

+----------------------------+

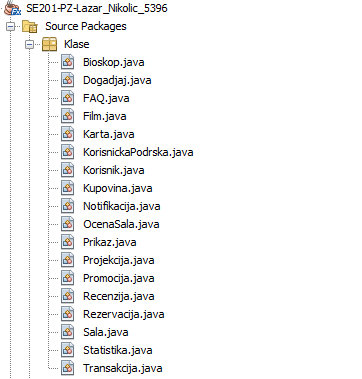
**Prednosti:**

* **Jasna separacija odgovornosti:** Lako je razumeti koji deo sistema radi šta.
* **Jednostavnije održavanje:** Izmene u jednom sloju obično ne utiču direktno na druge slojeve.

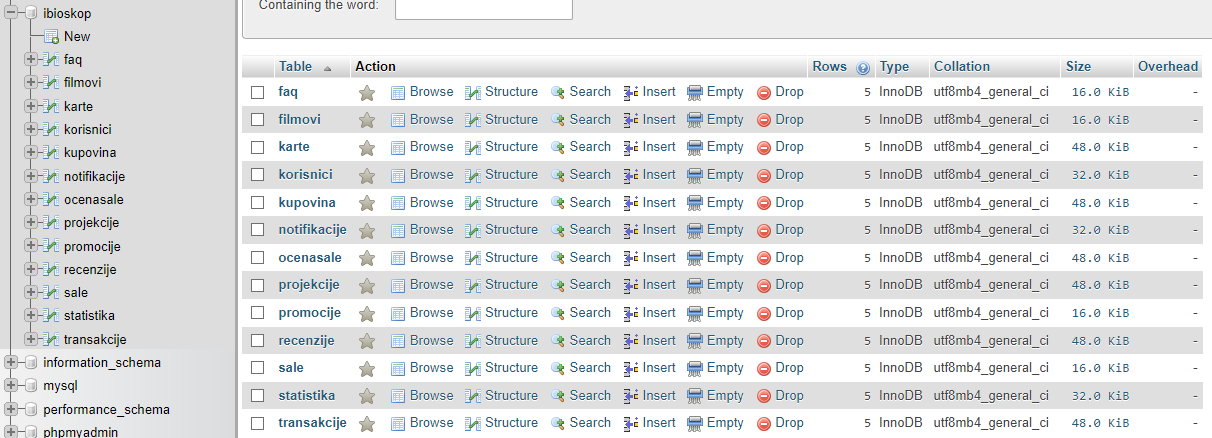
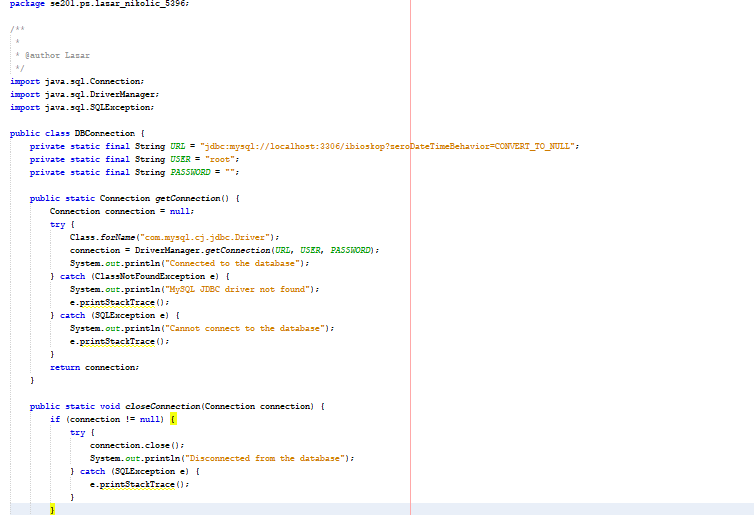
**Mane:**

* **Performanse:** Može doći do smanjenja performansi zbog višestrukih slojeva.
* **Teškoće u kompleksnim scenarijima:** Neki složeni zahtevi mogu biti teški za implementaciju.

1. Kreirati 10-15 klasa.



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



1. Izdvojiti komponente sistema koje mogu biti ponovo upotrebljene.

1. User klasa: Može se koristiti u različitim aplikacijama gde je potrebna autentifikacija korisnika.

2. DatabaseConnection klasa: Generička klasa za povezivanje sa bazom podataka.

3. Review klasa: Može se koristiti za aplikacije koje imaju sistem recenzija.

1. 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).

**Delovi sistema koji mogu biti primer racunarstva u oblaku:**

SaaS: Aplikacija "iBioskop" može biti implementirana kao SaaS gde korisnici pristupaju funkcionalnostima rezervacije i pregleda filmova preko weba.

PaaS: Korišćenje platformi kao što su Heroku ili AWS Elastic Beanstalk za hostovanje aplikacije.

IaaS: Korišćenje virtualnih mašina na AWS ili Google Cloud za deploy backend servera.

**Sto se tice tipa to bi bili tipovi:**

**PaaS(Platform as a Service)**-Ovaj tip nam dozvoljava razvoj I izvrsavanje aplikacije koja moze biti putem oblaka,omogucavajuci programerima da se fokusiraju na ravoj bez brige o infrastrukturi.

**Drugi tip moze biti:**

**SaaS(Software as a Service)**-To su aplikacije poput sistema za upravljanje bazom podataka,za analizu podataka.

**Komponente koje mogu biti zamenjene vec postojecim resenjima:**

* Autentifikacija: Može se koristiti Auth0 ili Firebase Authentication umesto implementacije sopstvenog sistema.
* Skladištenje recenzija: Može se koristiti usluga kao što je Google Cloud Datastore ili Amazon DynamoDB.

**Delovi sistema za koje racunarstvo u oblaku nije najbolje resenje:**

Osnovne interne operacije koje zahtevaju minimalan broj korisnika: Npr. administrativne funkcije koje se ne koriste često mogu ostati lokalne zbog sigurnosti i troškova.

1. Izdvojiti komponente sistema koji mogu biti implementirani kao mikroservisi sa odvojenim bazama podataka.

**Autentifikacija i Autorizacija:**

Mikroservis: Servis za upravljanje korisničkim nalozima, autentifikacijom i kontrolom pristupa.

**Baza podataka:**

Sadrži informacije o korisnicima, njihovim pravima i pristupu.

**Upravljanje Filmovima:**

**Mikroservis**: Servis koji upravlja informacijama o filmovima.

**Baza podataka:** Čuva informacije o filmovima, uključujući naslove, opise, trajanje, žanr i režisere.

Upravljanje Projekcijama:

**Mikroservis:** Servis za upravljanje terminima projekcija.

**Baza podataka**: Čuva informacije o projekcijama, uključujući vreme, datum, salu i dostupnost karata.

**Upravljanje Rezervacijama:**

**Mikroservis:** Servis za upravljanje rezervacijama i kupovinom karata.

Baza podataka: Čuva informacije o rezervacijama, kupljenim kartama, korisnicima i njihovim rezervacijama.

**Upravljanje Recenzijama:**

**Mikroservis:** Servis koji upravlja recenzijama korisnika.

**Baza podataka:** Čuva informacije o recenzijama, uključujući ocene, komentare i korisničke profile koji su ostavili recenzije.

**Korisnički Interfejs i Povezivanje sa Uređajima:**

**Mikroservis:** Servis koji pruža korisnički interfejs i podržava povezivanje sa raznim uređajima.

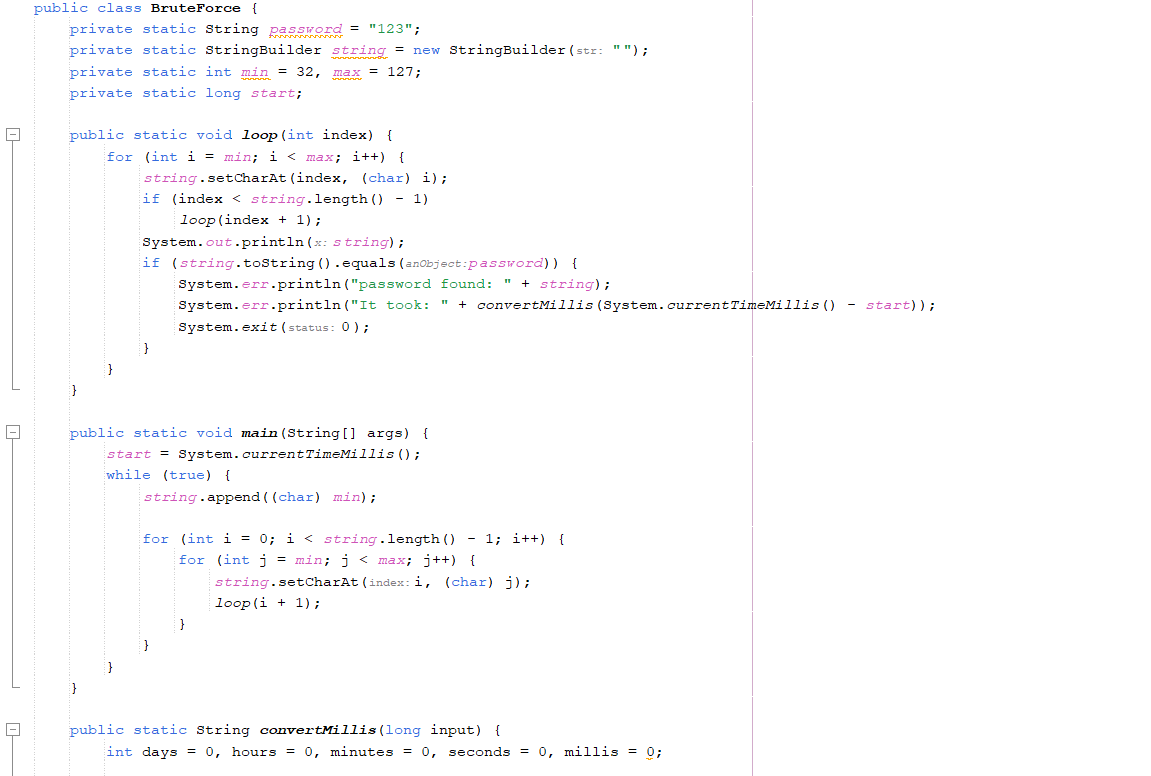
**Baza podataka:** Može sadržavati informacije o preferencama korisnika i istoriji aktivnosti.

**Upravljanje Bioskopskim Salama:**

**Mikroservis:** Servis za upravljanje informacijama o bioskopskim salama.

**Baza podataka:** Čuva informacije o salama, uključujući kapacitet, opremu i dostupnost.

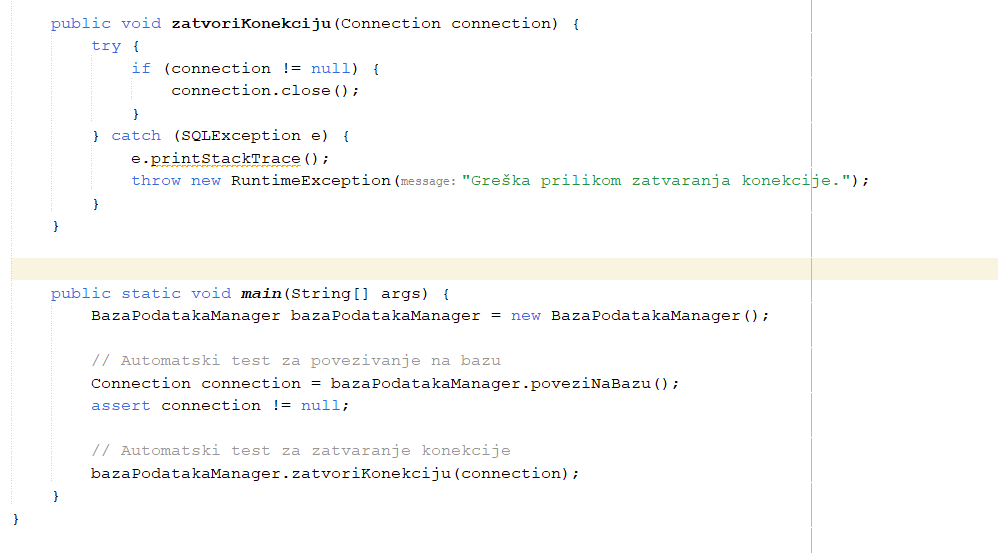
1. Na Vašem sistemu demonstrirati i dokumentovati Brute Force ili SQL injection napad.



1. Navedite i opišite atribute kvaliteta Vašeg sistema.
2. **Dostupnost (Availability)**: Sistem mora biti operativan 24/7 kako bi korisnici i zaposleni mogli neprekidno da koriste usluge. Ovo osigurava da korisnici mogu u bilo kom trenutku da rezervišu karte i pregledaju rasporede projekcija, dok zaposleni mogu upravljati sistemom bez prekida.
3. **Skalabilnost (Scalability)**: Sistem mora biti u stanju da obradi sve veći broj korisnika i transakcija kako popularnost iBioskopa raste. Treba da se efikasno skalira kako bi mogao da podnese periodične vršne opterećenja bez degradacije performansi.
4. **Sigurnost (Security)**: Sistem mora štititi korisničke podatke i informacije o transakcijama. Ovo uključuje enkripciju osetljivih podataka, sigurne mehanizme autentifikacije i zaštitu od uobičajenih ranjivosti kao što su SQL injekcija i XSS napadi.
5. Primeniti tri koncepta pouzdanog programiranja:

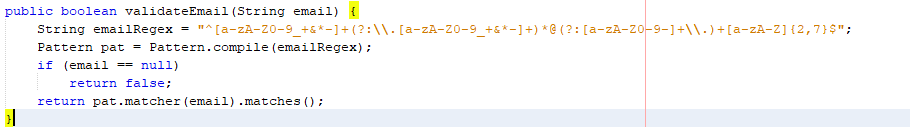
**Rukovanje greškama:**

Umesto čekanja da se greške pojave, razvijajte sistem s fokusom na identifikaciju i rukovanje potencijalnim problemima unapred.



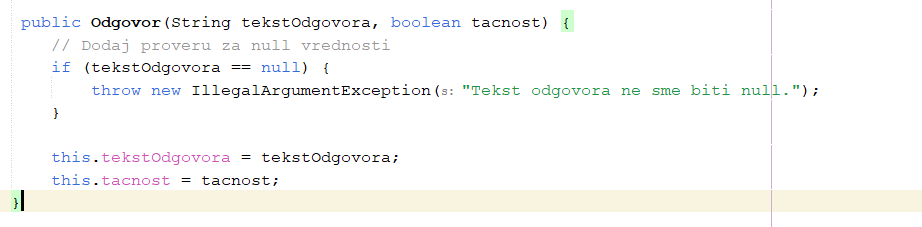
**Validacija inputa:**

Validacija unosa je ključna za obezbeđenje sigurnosti i stabilnosti sistema. Validacija osigurava da podaci uneseni od strane korisnika ispunjavaju određene kriterijume pre nego što budu procesuirani.



**Provera za null:**

Ovo nam omugucava sprecavanje iznenadnih NullPointerExeption.



1. Kreirati od pet do deset scenarija testiranja (tabela je data u prilogu).
2. **Kreiranje novog filma:**

Scenario: Korisnik unosi sve potrebne informacije o novom filmu.

Ulazni podaci: Naslov: "Novi film", Opis: "Opis filma", Trajanje: 120 min, Žanr: "Akcija"

Očekivani rezultat: Film je uspešno sačuvan u bazi podataka.

**2**. **Dodavanje nove projekcije:**

Scenario: Administrator dodaje novu projekciju za film.

Ulazni podaci: FilmID: 123, Vreme: "2024-06-14 20:00", Sala: "Sala 1"

Očekivani rezultat: Projekcija je uspešno sačuvana u bazi podataka.

**3. Rezervacija karte:**

Scenario: Korisnik rezerviše karte za određenu projekciju.

Ulazni podaci: ProjekcijaID: 456, Broj karata: 2

Očekivani rezultat: Rezervacija je uspešno sačuvana u bazi podataka.

**4. Dodavanje recenzije za film:**

Scenario: Korisnik dodaje recenziju za film.

Ulazni podaci: FilmID: 123, Ocena: 5, Komentar: "Odličan film!"

Očekivani rezultat: Recenzija je uspešno sačuvana u bazi podataka.

**5. Prikazivanje informacija o filmu:**

Scenario: Korisnik bira film i prikazuju se sve informacije o filmu.

Ulazni podaci: FilmID: 123

Očekivani rezultat: Informacije o filmu su tačno prikazane korisniku.

**6. Izmena informacija o filmu:**

Scenario: Administrator menja informacije o postojećem filmu.

Ulazni podaci: FilmID: 123, Novi naslov: "Izmenjeni film", Novi opis: "Novi opis filma"

Očekivani rezultat: Izmene su uspešno ažurirane u bazi podataka.

**7. Brisanje filma:**

Scenario: Administrator briše postojeći film iz sistema.

Ulazni podaci: FilmID: 123

Očekivani rezultat: Film je uspešno obrisan iz baze podataka.

**8. Prikaz rasporeda projekcija:**

Scenario: Korisnik traži raspored projekcija za određeni datum.

Ulazni podaci: Datum: "2024-06-14"

Očekivani rezultat: Prikazuju se sve projekcije za taj datum.

**9. Odgovaranje na pitanje tokom registracije:**

Scenario: Korisnik unosi odgovore na pitanja tokom registracije.

Ulazni podaci: Korisničko ime, Lozinka, Email

Očekivani rezultat: Korisnik je uspešno registrovan i informacije su sačuvane u bazi podataka.

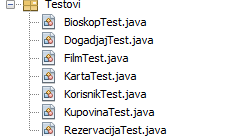
**10. Promena lozinke korisnika:**

Scenario: Korisnik menja svoju lozinku.

Ulazni podaci: Stara lozinka, Nova lozinka

Očekivani rezultat: Lozinka je uspešno promenjena i ažurirana u bazi podataka.

1. Na osnovu kreiranih scenarija testiranja izvršiti Junit testiranje na Vašem sistemu.



1. Ceo proces planiranja i implementacije softvera ispratiti koristeći alat Trello

https://trello.com/invite/b/df7pLDj5/ATTI519ab8c1537119b2577cff8edaa07ff3BA621348/se201-projekat

1. Pratiti razvoj projektnog zadatka verzioniranjem na Git-u.

https://github.com/nevidljivicovek/SE201-Projekat