VISOKA ŠKOLA STRUKOVNIH STUDIJA ZA INFORMACIONE TEHNOLOGIJE

Završni rad

Projektovanje i implementacija Java Web aplikacije za podršku poslovanju planinarskog kluba "Summit Explorers" korišćenjem Spring framework-a

Mentor:
dr Svetlana Jevremović

Student:
Uroš Mitrašinović 610/17

Beograd
Februar, 2023

SADRŽAJ

REZIME

Ovaj završni rad sastoji se iz tri dela. U prvom delu je data osnovna teorija Spring framework-a i objašnjeno je zašto je korišćena baš ova tehnologija za izradu veb aplikacije ovog završnog rada. U drugom delu, kroz korišćenje Larmanove metode razvoja softvera, detaljno je dokumentovan proces izrade softvera. Svaki od zahteva koje je softver predviđeno da ispuni je prikazan kroz faze koje sadrže mnoštvo modela, dijagrama, i prikaza korisničkog interfejsa, kroz koje se olakšava dalji proces održavanja softvera. Treći deo je konačni proizvod ovog završnog rada, a to je veb aplikacija koja ima za cilj da olakša i optimizuje rad planinarskog kluba.

SPISAK KLJUČNIH REČI

Java, Spring, Larmanova metoda, Spring Boot, Spring Data JPA, Dependency Injection, MySQL, Thymeleaf

UVOD

Temu ovog završnog rada izabrao sam sa ciljem da za ovo specifično tržište - planinarski klub – izradim veb aplikaciju koja će imati kompletniju veb prezentaciju i funkcionalnosti za podršku poslovanja u odnosu na konkurenciju. Istraživanjem tržišta na našem prostoru, zaključio sam da ne postoji kompletna digitalna podrška za poslovanje u okviru ovog specifičnog domena. Većina planinarskih klubova, odnosno planinarskih udruženja, ima samo delimičan ili nikakav digitalni poslovni model, sa osiromašenom IT podrškom. Uvođenjem kompletne IT podrške, usluga bi bila znatno olakšana i privukla bi nove članove, ali i popularizovala ovo tržište, kako bi se moglo nositi sa recimo, tržištima turističkih agencija, koji su već odavno uveli slične tipove planinarskih putovanja u svoje ponude i imaju daleko bolju IT podršku.

U nastavku će biti data teorija Spring radnog okvira kao glavnog alata za implementaciju veb aplikacije ovog završnog rada. Poglavlja su podeljena na osnovna načela na kojima počiva Spring. Opisaće se kako funkcioniše njegova arhitektura, kao i značenje njegovih potprojekata u implementaciji softvera.

Nakon toga, fokus će biti preusmeren na praktični deo, koji podrazumeva detaljno opisan proces projektovanja i implementacije veb aplikacije kroz Larmanovu metodu razvoja softvera. Poglavlja su podeljena na faze definisane ovom metodom.

1. SPRING FRAMEWORK

Radni okvir Spring obezbeđuje sveobuhvatni model programiranja i konfiguracije za moderne poslovne aplikacije zasnovane na jeziku Java. Njegova glavna svrha je da vodi računa o svim tehničkim vezama i zavisnostima koje su potrebne da bi se povezali različiti delovi aplikacije. To omogućava programerima da se fokusiraju na srž svog posla – na pisanje poslovne logike [7].

Prvo produkcijsko izdanje Spring-a (Spring 1.0) izašlo je u martu 2004. godine.

Ovaj radni okvir je nastao sa namerom da se pojednostavi složenost aplikacija rađenih u J2EE tehnologiji (Java 2 Platform Enterprise Edition, danas poznatiji kao Jakarta EE). Baziran na dependecy injection („ubrizgavanje zavisnosti") mehanizmu, kreiran je kao alternativa EJB (Enterprise Java Bean, specifikacija J2EE) steku sa distribuiranim objektima, čija kompleksnost izrade je bila nepotrebna u većini aplikacija. Tradicionalni pristup J2EE tehnologiji je uneo mnogo složenosti kada se koristio za podizanje aplikacija. Povrh toga, uključivao je dodatnu bespotrebnu složenost prilikom ostvarivanja poslovnih zahteva koje je morao da reši. Kao rezultat svega ovoga, aplikacije su bile preskupe za razvijanje i održavanje, i bilo ih je teško testirati [2].

Spring je u to vreme uneo pravo osveženje u razvoju poslovnih aplikacija, omogućavajući pre svega lako testiranje i održavanje. Zahvaljujući konstantnom unapređivanju, ostaje relevantan i danas, i važi za jedan od najpopularnijih i najkorišćenijih frameworka za razvoj poslovnih i veb aplikacija.

Glavna prednost Spring radnog okvira, iz ugla programera, predstavlja njegov ekosistem, kojeg čine veliki broj modula i projekata. Ova modularnost omogućava da aplikacija, ručno ili upotrebom alata za upravljanje zavisnostima (eng. Dependency management), kao što je Maven, importuje samo one module koji joj zaista trebaju.

Spring je otvorena (eng. Open-source) Java platforma, čiji moduli, kojih ima preko 20, pokrivaju gotovo sve moguće oblasti programiranja:

- Razvoj veb aplikacija povezanih sa relacionom bazom podataka
- Integrisanje ugrađenih (eng. Embedded) web servera i baze podataka u izvršni fajl aplikacije
- Kreiranje Representational state transfer (REST) servisa
- Upotreba neke NoSQL varijante umesto relacione baze
- Povezivanje na neku društvenu mrežu
- Komunikacija sa drugim aplikacijama upotrebom messaging servera tipa ActiveMQ ili RabbitMQ
- Itd.

Spring radni okvir je izuzetno popularan. Njegovo znanje je traženo, ne samo kod nas nego i širom sveta. Neki od glavnih razloga koje Spring radni okvir čine popularnim su sledeći:

- Pojednostavljeno testiranje koda - Spring Inversion of Control kontejner - IoC, koncept Dependency Injection-a, ili u slobodnom prevodu "ubrizgavanje zavisnosti"
- Smanjivanje upravljačkog koda - npr. Spring JDBC modul uklanja potrebu za kreiranjem upravljačkog koda potrebnog za konekciju sa bazom podataka
- Arhitekturna fleksibilnost - npr. u sloju podataka umesto Spring JDBC, možemo koristiti bilo koji drugi radni okvir sloja podataka – JPA, Hibernate (sa ili bez JPA-a) ili iBatis. Ako ne želimo da upotrebimo specifičan radni okvir/modul, možemo da ga zamenimo drugim
- Praćenje promena (npr. projekat Spring Cloud)

1.1. SPRING MODULI

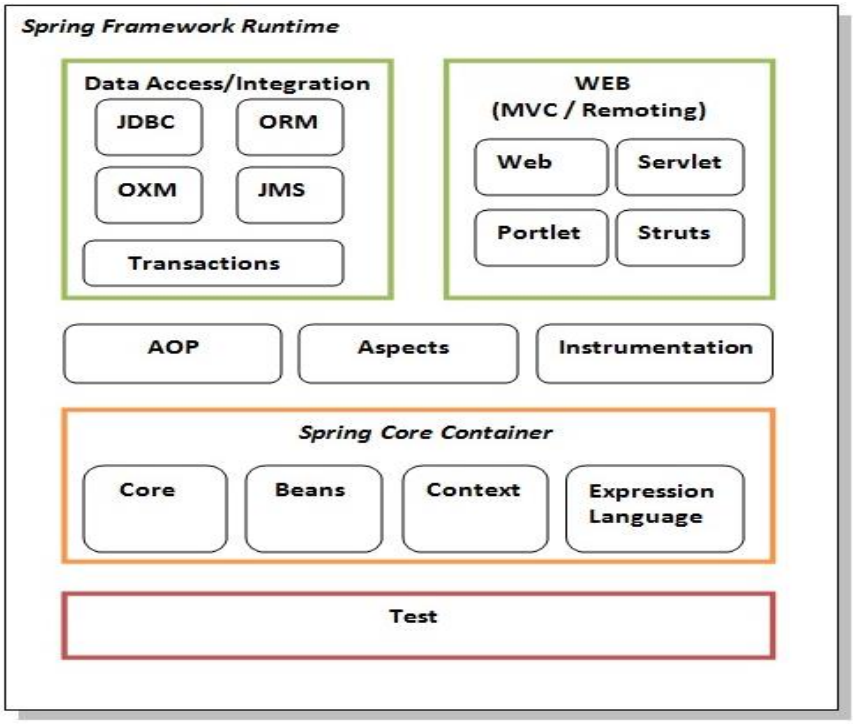
Modularnost radnog okvira Spring je jedan od najvažnijih razloga za njegovu popularnost [7]. Modularan pristup omogućava da se uz pomoć Spring radnog okvira razviju Java aplikacije različitog nivoa složenosti. U zavisnosti od zahteva aplikacije, moguće je koristiti bilo koji Spring modul nezavisno od drugih modula. Upravljanje slojem poslovne logike u aplikaciji se može izvršiti pomoću Springovog osnovnog (eng. core) kontejnera, dok se za druge delove aplikacije mogu koristiti neke druge tehnologije. To znači da Springova fleksibilna arhitektura omogućava i integraciju sa velikim brojem tehnologija.

Radni okvir Spring sadrži više od 20 različitih modula koji imaju jasno definisane granice [7].

Ovi moduli se mogu grupisati na osnovu njihovih primarnih karakteristika u 6 slojeva aplikacije:

- Osnovni (eng. Core) Kontejner sloj
- Sloj Pristup Podataka/Integracija (eng. Data Access/Integration)
- Web
- Aspect Oriented Programming - AOP
- Instrumentacija (eng. Instrumentation)
- Test

Navedeni slojevi arhitekture Springa prikazani su na slici 1.



Slika 1. Arhitektura Spring-a po slojevima (Izvor: https://www.w3school-learn.com/2017/12/spring-modules-tutorial.html)

Spring Core Kontejner sloj se sastoji od *Core, Beans ("Zrna"), Context* i *Expression Language* modula [3].

Spring Core čine pomoćni programi koje koriste drugi Spring moduli.

Spring Beans modul je podrška za Spring Bean-ove (zrna) i uključuje implementaciju interfejsa BeanFactory. U kontekstu Spring radnog okvira, termin bean se koristi za sve tipove objekata koji su kreirani od strane Spring Core kontejnera i kojima kontejner upravlja.

BeanFactory interfejs obezbeđuje napredni mehanizam konfiguracije sposoban za upravljanje bilo kojim tipom objekta (bean-a) [13].

Core i Beans moduli u kombinaciji obezbeđuju fundamentalne delove Spring-a, uključujući Inverziju Kontrole (Inversion of Control - IoC) i Dependency Injection mehanizme.

Context modul se gradi na temelju postavljenom od strane Core i Beans modula. Implementira ApplicationContext interfejs, koji proširuje BeanFactory i obezbeđuje dodatne fukncionalnosti, kao što su na primer učitavanje resursa i internacionalizacija.

Expression Language modul obezbeđuje poseban jezik za pristup i manipulaciju svojstava bean (uključujući nizove i kolekcije) na JSP (Java Server Pages) stranicama. Predstavlja ekstenziju ujedinjenog Expression Language-a (unified EL) specificiranog u JSP tehnologiji.

Sloj Pristup Podataka/Integracije se sastoji od JDBC, ORM, OXM, JMS i Transakcijskih modula [3].

Spring JDBC (Java Database Connectivity) modul pruža JDBC-apstrakcijski sloj koji otklanja potrebu za pisanjem šablonskog koda vezanog za konekciju sa bazom podataka.

Spring ORM modul obezbeđuje integraciju sa popularnim API-jima za objektno-relaciono mapiranje, na primer sa radnim okvirima JPA i Hibernate [3].

Spring OXM modul obezbeđuje apstrakcijski sloj za integraciju XML mapiranja. Podržava radne okvire kao što su JAXB, Castor, XMLBeans, JiBX i Xstream [3].

Spring JMS (Java Messaging Service) modul obezbeđuje apstrakcijski sloj za produkciju i konzumaciju poruka. Ovim modulom se, kao što je to slučaj i sa Spring JDBC modulom, otklanja potreba pisanja šablonskog koda.

Transakcioni modul Springa podržava programabilno i deklarativno upravljanje transkacijama za klase koje implementuju specijalne interfejse i za sve POJO (Plain Old Java Object) objekte [3].

Web sloj se sastoji od Spring Web, Spring Webmvc, WebSocket i Web-Portlet modula.

Spring Web modul obezbeđuje osnovne veb funkcije, kao što je slanje fajla koji se sastoji iz više delova. Obezbeđuje i podršku za integraciju sa drugim veb radnim okvirima, kao što je Struts [7].

Spring WebMvc modul sadrži Springov MVC (model-view-controller) radni okvir za veb aplikacije. Ovaj radni okvir obezbeđuje jasnu odvojenost između koda domenskog modela i veb formi, i lako se integriše sa svim ostalim funkcionalnostima Spring radnog okvira.

Spring WebSocket modul pruža podršku posebnoj arhitekturu za razmenu poruka poznatoj kao WebSocket. WebSocket-i su bidirekcione (u oba smera), full-duplex (u isto vreme, u oba smera), perzistentne konekcije između web browser-a i servera. Kada se jednom uspostavi ova konekcija, ona ostaje otvorena dok god klijent ili server ne odluče da je zatvore. Tipičan primer je aplikacija sa razmenom poruka između više korisnika, kao što je čet (chat).

Spring Web-Portlet modul pruža podršku primene MVC implementacije u portlet okruženju [3].

Sloj AOP (Aspect-Oriented Programming) podrazume više Spring modula vezanih za programiranje koje je vođeno aspektom – upotrebom presretača i pointcut metoda.

Spring Aspects modul obezbeđuje integraciju sa najpopularnijim i potpuno opremljenim AOP radnim okvirom AspectJ [7].

Spring Instrumentation modul obezbeđuje osnovnu podršku za instrumentaciju.

Spring Test modul obezbeđuje osnovnu podršku za testiranje koda i testiranje integracije [7].

U ovom poglavlju dat je opis Springove fleksibilne modularne arhitekture. Navedeni su slojevi po kojima su grupisani moduli, i dat je kratak opis za neke od tih modula.

1.2. INVERSION OF CONTROL I DEPENDENCY INJECTION

U ovom poglavlju će biti dat kratak opis samog principa inverzije zavisnosti (DIP) i zašto se koristi. Potom ću se osvrnuti na Spring Inversion of Control (IoC) kontejner i Dependency Injection mehanizam na kojem se zasniva. Ove funkcionalnosti su sadržane u osnovnom Spring Core Container sloju i zajedno čine temelj nad kojim se nadograđuju svi ostali Spring moduli i projekti.

1.2.1. Dependency Inversion Principle (DIP)

Princip inverzije zavisnosti (Dependency Inversion Principle – DIP) je princip dizajna aplikacije koji predstavlja određenu vrstu vodiča kako da se aplikacijski kod organizuje da bude labavo povezan. Princip kaže sledeće [9]:

- Moduli visokog nivoa ne trebaju da zavise od modula niskog nivoa za realizaciju njihovih odgovornosti. Obe vrste modula treba da zavise od apstrakcija.
- Apstrakcije ne trebaju da ovise o detaljima. Detalji trebaju da ovise od apstrakcija.

Promene su uvek riskantne kada se prave u ovisnom kodu. DIP govori o tome da se određeni deo koda (zavisnost) drži podalje od glavnog programa sa kojim nije direktno povezan [9].

Predlaže eliminisanje direktne zavisnosti modula visokog nivoa, koji samim tim izvršava bitnije i kompleksnije zadatke, od modula niskog nivoa.

Na ovaj način, sama implementacija modula niskog nivoa može da se menja, bez da te promene utiču na promene u modulu visokog nivoa. Iz ovoga proističe velika fleksibilnost i molekularnost sistema. Sve dok je bilo koja implementacija niskog nivoa vezana za apstrakciju, moduli visokog nivoa mogu da se pozovu na nju.

Principi dizajna, kao što je ovaj, uvek prikazuju dobru praksu i rešenje problema dizajniranja. Međutim, ne govore o samom implementiranju.

Ovde nastupa Inverzija Kontrole - IoC (Inversion of Control). IoC prikazuje način definisanja pomenutih apstrakcija između modula. Ukratko, IoC je način implementiranja DIP [9].

U ovom poglavlju dat je opis i značenje Principa Inverzije Zavisnosti. Njegovo razumevanje je bitno, jer predstavlja prvi korak ka pravljenju aplikacionog koda koji je čistiji, čitljiviji, raščlanjeniji, održiv i modularan.

U narednom poglavlju, želim da se osvrnem na to šta je pojam Inverzija Kontrole (IoC), i kako je IoC definisan i implementiran u Spring radnom okviru kroz Spring IoC kontejner i Dependency Injection mehanizam.

1.2.2. Inverzija kontrole i Spring IoC kontejner

Pojam Inversion of Control (IoC) predstavlja metodologiju dizajniranja, odnosno projektovanja, koja se koristi za kreiranje takozvanog labavo povezanog (eng. loosely coupled) sistema u softverskom inženjerstvu, tako što invertuje kontrolu toka aplikacije sa glavnog programa na neki drugi entitet ili radni okvir [9].

U ovom kontekstu, kontrola se odnosni na dodatne aktivnosti koje program radi, a koje se ne smatraju glavnim, kao što je održavanje zavisnih objekata – u kontekstu Spring-a to su bean-ovi, upravljanje aplikacionim tokom, itd. Uprošćeno, kontrola većine aktivnosti koje se ne odnose na poslovnu logiku aplikacije.

IoC implementuje princip DIP, opisan u prethodnom poglavlju, tako što obezbeđuje apstrakcijski sloj koji povezuje module visokog i niskog nivoa. Postoji više tipova inverzije kontrole [9]:

- Invertovanje intefejsa: Modul višeg nivoa definiše interfejs, dok modul nižeg nivoa ga implementira
- Invertovanje kreiranja objekta: Pomeranje odgovornosti kreiranja zavisnosti sa glavnog modula na neki drugi program ili radni okvir
- Invertovanje toka: Promena toka aplikacije

Rešenja za IoC metodologiju dolaze najčešće u vidu IoC kontejnera. Kontejner se brine o kreiranju, konfigurisanju i upravljanju objektima. Programer samo treba da implementira konfiguraciju, odnosno dostavi konfiguracione metapodatke koji predstavljaju instrukcije kontejneru, nakon čega će IoC kontejner da se pobrine za instanciranje objekata i upravljanje zavisnostima. Dodatno kodiranje nije potrebno.

U kontekstu Spring-a, IoC implementacija dolazi u vidu Spring IoC kontejnera.

Spring IoC kontejner kreira zrna i povezuje ih u skladu sa podešenom konfiguracijom koju kreira programer aplikacije. Bitno je napomenuti da kontejner svoj posao upravljanja životnim ciklusom bean-ova i ubrizgavanja zavisnosti radi u toku izvršenja aplikacije. Implementacija ovog kontejnera realizovana je kroz dva intefejsa, koja sam kratko pomenuo u poglavlju o Spring modulima, a to su:

- BeanFactory intefejs paketa *org.springframework.beans.factory*
- ApplicationContext intefejs paketa *org.springframework.context*

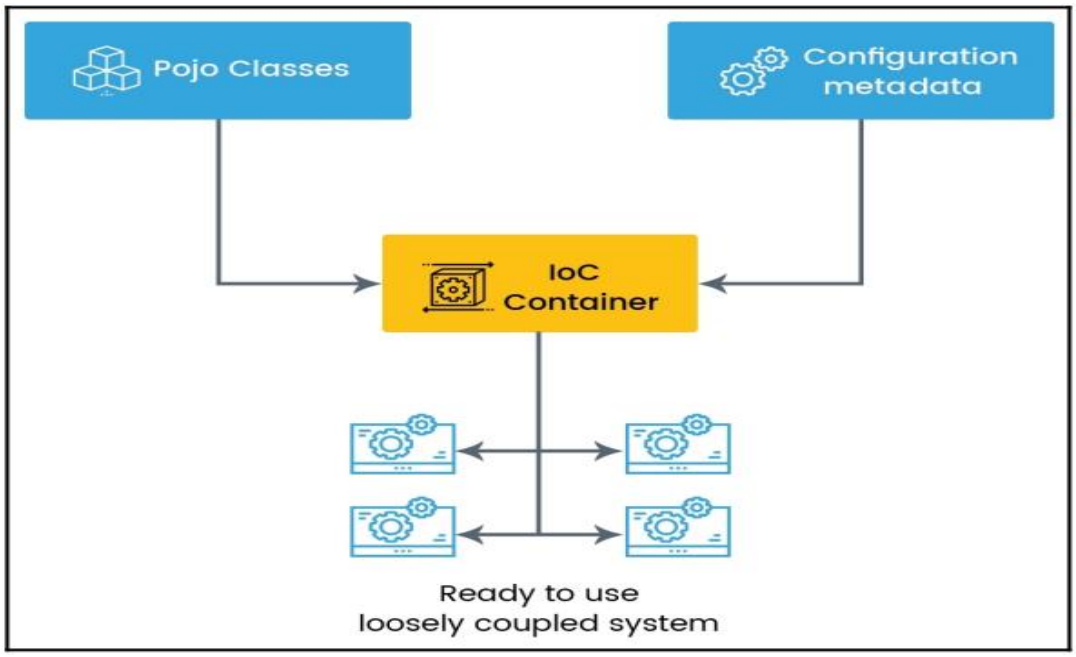
BeanFactory je osnovni (koreni) intefejs Spring kontejnera. Obezbeđuje osnovne funkcionalnosti kreiranja i upravljanja životnim ciklusom bean-ovima.

ApplicationContext intefejs je pod-intefejs BeanFactory-ja. To znači da on obezbeđuje sve funkcionalnosti koje BeanFactory nudi. Osim toga, obezbeđuje više funkcionalnosti za enterprise aplikacije. Bitne dodatne funkcionalnosti ApplicationContext-a su [5]:

- Lakša integracija sa Spring AOP (Aspect-Oriented Programming) funkcionalnostima
- Rukovanje resursima poruka (podrška u internacionalizaciji softvera)
- Propagacija događaja (event propagation)
- Dodatni konteksti aplikacionog sloja kao što je WebApplicationContext koji se koristi u veb aplikacijama

Zaključuje se da ApplicationContext intefejs predstavlja superset BeanFactory-ja, pa se zato koristi kao uobičajena reprezentacija Spring IoC kontejnera.

Dijagram, prikazan na slici 2, prikazuje ulazno-izlazne parametre Spring IoC kontejnera.



Slika 2. Ulazno-izlazni parametri IoC kontejnera [9]

Kao što se može videti na slici iznad, programer je zadužen da Spring IoC kontejneru dostavi POJO klase (bean-ove) i konfiguracijske metapodatke kao ulazne parametre. Spring IoC kontejner će, na osnovu toga, kreirati i organizovati bean-ove na način na koji oni proizvode sistem spreman za upotrebu.

1.2.3. Definisanje konfiguracionih metapodataka

Prilikom definisanja i dostavljanja konfiguracijskih metapodataka Spring kontejneru, bitno je spomenuti načine koje programer ima na raspolaganju da to uradi. Postoje tri načina i to su:

- XML format: Jedan ili više unosa sa konfiguracijskim metapodacima o bean-ovima u XML konfiguracijskom fajlu (na primer web.xml)
- Java anotacije: Konfiguracijski metapodaci su postavljeni u formi anotacija u bean-ovima (npr. anotacije @PostConstruct, @Autowired, @Entity, itd.). Ovakav način je podržan od Spring-ove verzije 2.5.
- Čist java kod: Od Spring-ove verzije 3.0., podržan je način definisanja konfiguracije putem čistog java koda. Koristeći isključivo ovaj način, XML konfiguracijski fajlovi nisu potrebni. Sva konfiguracija bean-ova sadržana je u Java klasi anotiranoj sa @Configuration u okviru koje se definišu metode anotirane sa @Bean.

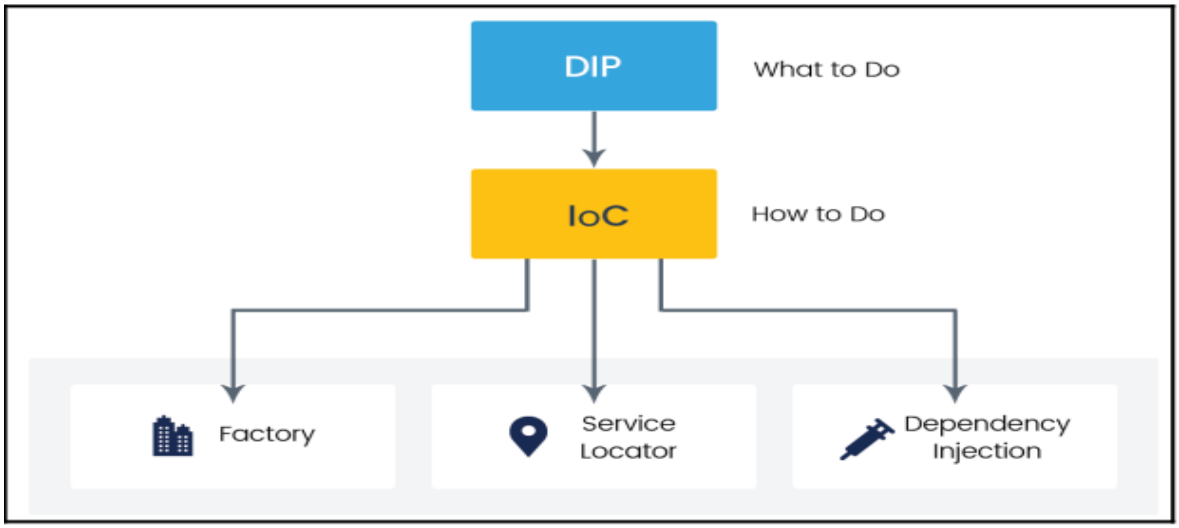
U ovom završnom radu koristio sam se sa sva tri načina definisanja konfiguracijskih metapodataka. Aplikacija sadrži pom.xml (POM – Project Object Model) fajl. Ovaj fajl predstavlja određenu vrstu XML konfiguracije koja sadrži osnovne informacije o projektu i konfiguracijske detalje koje koristi Maven alat za podizanje projekta. Što se tiče načina pomoću java anotacija, korišćene su razne anotacije, na primer: @Autowired, @Entity, @Repository, @Service, @Id, itd. Konfiguracija u obliku čistog java koda korišćena je, na primer, prilikom podešavanja koncepata autentifikacije i autorizacije, u okviru Spring Security-ja.

1.2.4. Dependency Injection (DI) mehanizam

Inverzija kontrole se može postići kroz više šablona dizajniranja. Neki od njih su:

- Factory pattern
- Service locator
- Dependency injection

Veza između principa inverzije zavisnosti (DIP), inverzije kontrole (IoC) i načina njenog implementovanja se može videti na slici 3.



Slika 3. Načini implementacije IoC [9]

Ovde će biti dat opis DI mehanizma, budući da je DI mehanizam najpopularniji od ovih načina, i da sam se u radu na aplikaciji isključivo koristio njime za implementaciju inverzije kontrole.

DI (Dependency Injection) je mehanizam, ili šablon dizajniranja (design pattern), po kojem Spring IoC kontejner funkcioniše.

Jedan objekat obično zahteva objekte drugih klasa kako bi izvršio svoje operacije. Ti objekti se nazivaju zavisnostima (dependencies).

Injection, u slobodnom prevodu „ubrizgavanje", je proces obezbeđivanja neophodnih zavisnosti objektu.

Dakle, DI predstavlja metod „ubrizgavanja" zavisnosti između objekata – bean-ova. Postoje tri načina ubrizgavanja/ubacivanja zavisnosti. To su:

- Constructor Injection (ubrizgavanje konstruktora)
- Setter Injection (ubrizgavanja setter metoda)
- Field Injection (ubrizgavanje polja/atributa)

Pre nego što opišem svaki od ovih načina, potrebno je spomenuti anotaciju @Autowired. Registrovanjem svih komponenti projekta u Springovom aplikacionom kontekstu, odnosno kontejneru, objekti postaju bean-ovi, i kao takvi omogućavaju kontejneru da ih automatski poveže u skladu sa definisanim zavisnostima. Ovo takozvano springovo automatsko povezivanje između bean-ova (Spring bean autowiring) se postiže postavljanjem anotacije @Autowired na odgovarajuća mesta.

U DI zasnovanom na konstruktoru (constructor-based injection), zavisnosti koje su potrebne bean-u su navedene kao argumenti parametrizovanog konstruktora. Anotacija @Autowired se

stavlja iznad definicije konstruktora, ukoliko klasa ima definisano više od jednog konstruktora. Ukoliko nema, anotacija @Autowired je opciona.

U DI zasnovanom na setter metodi (setter-based injection), u bean klasi se definiše zavisnost kao posebno polje/atribut, dok se vrednost tog polja dodeljuje u njegovoj setter metodi. Anotacija @Autowired se postavlja iznad setter metode.

Što se tiče DI zasnovanom na polju/atributu (field-based injection), anotacija @Autowired se postavlja direktno iznad definisanog polja/atributa koje predstavlja zavisnost.

Sve tri varijante DI mehanizma daju iste rezultate. Međutim, sa aspekta objektno-orjentisanog principa dizajna, postoje određene razlike i preporuke u kojim slučajevima koristiti svake od ovih varijanti.

Što se tiče aspekta čitljivosti, koji pomaže u održavanju softvera, najbolje je primeniti Field Injection varijantu.

Sada pogledajmo sa aspekta nepromenljivosti. Objekat se smatra nepromenljivim (immutable) ukoliko se njegovo stanje ne može izmeniti nakon kreiranja. Ovo je veoma važan aspekt objektno-orjentisanog principa, jer omogućava takozvani thread-safety (sigurnost i integritet podataka u Javi usred njihovog korišćenja ili deljenja tokom istovremenog izvršenja više delova programa, garancija očekivanog ponašanja objekta) i sigurnost stanja objekta (state safety). Samo Constructor Injection varijanta podržava princip nepromenljivosti.

Aspekt sigurnosti stanja (state safety) podrazumeva korektno i potpuno instanciranje objekata koji je spreman za dalju upotrebu. U Spring framework-u korišćenjem anotacije @Autowired, sve tri varijante garantuju sigurnost stanja. Međutim, u nekim drugim slučajevima, kao što je korišćenje sintakse new za instanciranje objekata, sigurnost stanja je drugačija. U ovom slučaju, Constructor Injection varijanta je i dalje najbolja, s obzirom na to da se objekat instancira u svoje potpuno stanje, ili se neće instancirati uopšte.

Sa aspekta klase koja ima previše atributa, Field Injection varijanta je najbolja opcija. Međutim, problem ovog scenarija je u lošem dizajnu, jer klasa ima previše zavisnosti.

Dolazi se do zaključka da je najbolja praksa da se koristi DI varijanta zasnovana na konstruktoru.

1.3. SPRING PROJEKTI

U ovom poglavlju biće dat kratak opis Springovih projekata. Kao što sam i pomenuo u prethodnim poglavljima, u okviru Spring frameworka, postoje različiti Spring moduli. Svi ti moduli može se reći da pripadaju istom projektu. Mogli bismo da nazovemo taj projekat kao Spring Framework projekat. Međutim, spring razvojni tim je prepoznao da, kako bi njihov framework ostao među najpopularnijima i najtraženijima, morao bi da pokrije sve savremene probleme razvijanja poslovnih aplikacija. Takvih problema u modernom svetu ima mnogo i sve ih je više, te sam Spring Framework projekat nije dovoljan. Kako bi ostali „u toku" sa modernim tehnologijama, Spring razvojni tim je uveo nove Spring projekte. Ovi projekti predstavljaju određenu nadogradnju na Spring framework. Drugim rečim, Spring framework je kamen temeljac svih ovih projekata. Termin „Spring" u IT industriji se može protumačiti na različite načine, u zavisnosti od konteksta. Međutim, najčešće danas kada se kaže „Spring", misli se na čitavu familiju projekata.

Nudeći različite infrastrukture, Spring projekti istražuju integraciju i rešenja za različite probleme u poslovnom okruženju – kao što su primena i projekti Cloud, Big Data, Batch i Security [7].

Neki od najvažnijih i najrazvijenijih Spring projekata su:

- Spring Boot – okvir koji postavlja i podiže aplikaciju
- Spring Data – okvir za rad sa bazama podataka
- Spring Batch („gomila") – okvir kao podrška za masovnu obradu podataka za izvršenje kompleksnih poslovnih operacija, pozadinske (background) operacije i slično
- Spring Cloud – okvir koji obezbeđuje alate za rad u „oblaku"
- Spring Security – okvir za obezbeđivanje sigurnosti aplikacije
- Spring Integration – okvir za upravljanje porukama
- Spring HATEOAS (Hypermedia as The Engine of Application State) – okvir koji obezbeđuje alate za podršku REST servisima u razdvajanju provajdera servisa (servera) i korisnika servisa (klijenta)

U narednim potpoglavljima opisaću detaljnije neke od ovih projekata koji su primenjeni u veb aplikaciji ovog završnog rada – Spring Boot, Spring Data i Spring Security.

1.3.1. Spring Boot

Spring radni okvir se naširoko koristi za razvijanje skalabilnih aplikacija. Skalabilnost u današnje vreme predstavlja jednu od najznačajnijih odlika sistema, a opisuje njegovu sposobnost da se lako prilagodi povećanom radnom opterećenju (na primer održanje visokih performansi tokom opsluživanja više korisnika istovremeno). Za veb aplikacije Spring obezbeđuje Spring MVC modul koji se koristi za razvijanje skalabilnih veb aplikacija. Ali, glavni nedostatak osnovnog Spring projekta je nužnost pisanja konfiguracije, koja je monotona, oduzima puno vremena i može biti preteška za početnike. Podešavanje aplikacije da bude produkcijski spremna (production-ready) traži određeno vreme, pogotovo programerima koji se tek upoznaju sa Spring-om. Rešenje za ovaj problem je Spring Boot. Spring Boot, kao i ostali Spring projekti, je građen na temelju osnovnog Spring frameworka i sadrži sve njegove funkcionalnosti. Baziran na mikroservisima, njegovo okruženje je produkcijski spremno, tako da je pokretanje aplikacije lako.

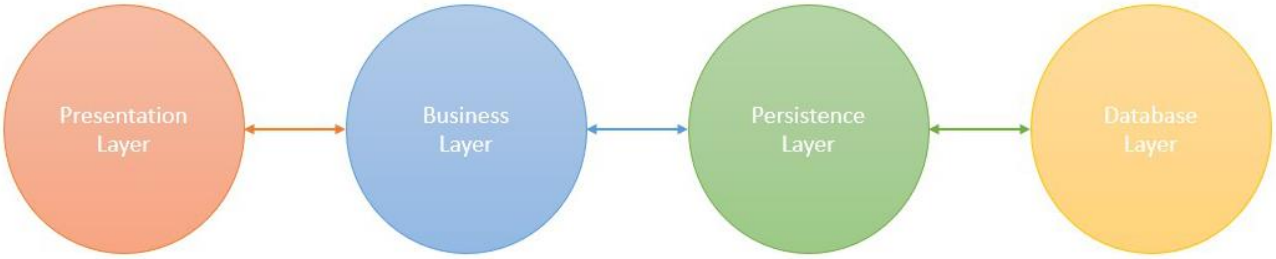
Zauzima stav, odnosno mišljenje na koji način aplikacije treba da budu razvijene. Neke od glavnih funkcionalnosti koje Spring Boot nudi su [8]:

- Podrška za razvijanje samostalnih Spring aplikacija
- Ugrađuje veb server Tomcat, Jetty, ili Undertow direktno, bez potrebe za otpremanjem WAR fajlova
- Obezbeđivanje mehanizma za eksternalizaciju konfiguracije za rad u različitim okruženjima sa istim aplikacionim kodom
- Pojednostavljuje Maven konfiguraciju tako što obezbeđuje početni POM (Project Object Model) xml fajl sa već ugrađenim početnim projektnim zavisnostima
- Eliminiše potrebu za generisanjem koda i kompleksnom XML konfiguracijom, a samim tim i neophodnost korišćenja previše anotacija. Zbog svega toga, on u velikoj meri automatizuje konfiguracijski proces.
- Obezbeđuje podršku za produkcijska svojstva kao što je metrika, takozvani „health check" za proveru operativnog statusa aplikacije, nadgledanje aplikacije (application monitoring) itd. Sve ovo pomoću modula Spring Boot Actuator koji je zadužen za praćenje aplikacije.

Arhitektura Spring Boot-a je slojevita. Sastoji se od 4 sloja a to su:

- Prezentacioni sloj
- Biznis sloj
- Perzistentni sloj
- Sloj baze podataka

Na slici 4 se može viditi slojevita arhitektura Spring Boot-a, kao i način komunikacije između slojeva.



Slika 4. Slojevi Spring Boot-a (Izvor: https://www.geeksforgeeks.org/spring-boot-architecture/)

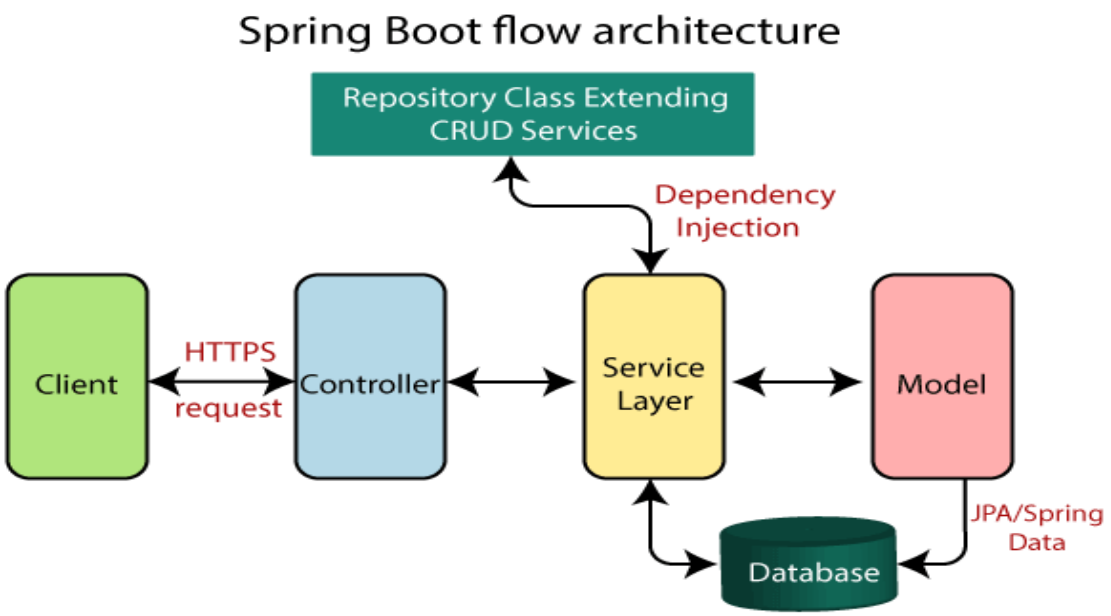
Prezentacioni sloj se bavi opsluživanjem HTTP zahteva, prevođenjem JSON podatka u objekat i obrnuto. Takođe, autentifikuje zahtev i prosleđuje ga biznis sloju. Sastoji se od View-ova (pogleda), odnosno predstavlja front-end deo aplikacije.

Biznis sloj sadrži svu poslovnu logiku. Njega čine servisne klase. Odgovoran je za validaciju i autorizaciju.

Perzistentni sloj se bavi prevođenjem biznis objekata u redove baze podataka. Sadrži svu logiku za skladištenje.

Sloj baze podataka sadrži sve baze podataka, kao što je MySQL, MongoDB, PostgreSQL itd. Odgovoran je za izvršavanje CRUD (Create Read Update Delete) operacija.

Na slici 5 može se videti protok obrade jednog zahteva u Spring Boot-u.



Slika 5. Spring Boot obrada zahteva (Izvor: https://www.javatpoint.com/spring-boot-architecture)

Objašnjenje slike iznad po koracima:

- Klijent (Client) kreira HTTP zahtev (GET, POST, PUT, itd.).
- HTTP zahtev se prosleđuje kontroler komponenti (Controller). Kontroler mapira zahtev sa kontroler metodom i obrađuje ga. Ukoliko je neophodno, poziva servisnu logiku, odnosno servisne klase za dalju obradu.
- U servisnom sloju se izvodi sva biznis logika. Spring Boot izvodi svu logiku nad podacima iz baze podataka koji su mapirani sa Spring Boot Model klasom preko takozvane Java Persistence biblioteke (JPA library)
- Kao HTTP odgovor (HTTP response), klijentu se vraća JSP/HTML stranica od kontrolera.

Kada sam navodio koje to glavne Spring Boot funkcionalnosti nudi, spomenuo sam Maven i POM.

Maven je alat za upravljanje zavisnostima (dependency management). Spring Boot ga koristi kako bi lakše učitao zavisnosti, odnosno uveo korektne verzije JAR (Java ARchive) fajlova za različite njegove potprojekte. Zbog toga ne moramo da navedemo verzije zavisnosti u našoj

konfiguraciji, izbegavajući problem kompatibilnosti. Spring Boot automatski nadogradi verzije svih navedenih zavisnosti kada ažuriramo samu Spring Boot-ovu verziju.

Maven, dakle, obezbeđuje centralizaciju informacija o zavisnostima. U Spring Boot-u, pom.xml fajl je osnovna jedinica za rad Maven-a.

POM je akronim za Project Object Model (projektni objektni model). To je XML fajl koji sadrži glavne informacije o projektu i konfiguracijske detalje koje koristi Maven kako bi izgradio projekat [4].

Prilikom izgradnje Spring Boot aplikacije, pretražuju se razni početni projekti koji nam trebaju i ubacuju u pom.xml fajl.

Početni projekti predstavljaju pojednostavljene opise zavisnosti koji su prilagođeni za različite namene. Na primer, spring-boot-starter-web je početni projekat za izgradnju veb aplikacije, uključujući RESTful veb servise, pomoću Spring MVC-a [7].

Koristi Tomcat kao veb server, u okviru spring-boot-starter-tomcat početnog projekta, koji je unapred konfigurisan i uključen u spring-boot-starter-web.

Spring Boot Starter Web početni projekat je takođe i jedna od glavnih zavisnosti uključenih u ovaj završni rad.

Svi početni projekti Spring Boot-a prate sličan obrazac imenovanja: spring-boot-starter-*, gde * predstavlja različite tipove aplikacije. Navešću još nekoliko početnih projekata koje Spring Boot obezbeđuje, a to su:

- spring-boot-starter – Osnovni početni projekat za Spring Boot aplikacije. Podrška za automatsku konfiguraciju i evidentiranje.
- spring-boot-starter-test – Podrška za radne okvire za testiranje koda, kao što su JUnit, Mockito i Hamcrest machers.
- spring-boot-starter-security – Obezbeđuje automatsku konfiguraciju za Spring Security.
- spring-boot-starter-data-jpa – Podrška za Spring Data JPA interfejs. Standardna implementacija je Hibernate radni okvir.
- spring-boot-starter-mail – Podrška Java Mail-u i za Spring-ovo slanje mejlova
- spring-boot-starter-thymeleaf – Podrška izgradnji MVC veb aplikacija koristeći Thymeleaf poglede. Thymeleaf je takozvani template engine (bukvalan prevod „šablonski mehanizam“) koji jako kompatibilan sa Spring MVC-om i predstavlja odličnu alternativu za JSP stranice. Glavna prednost mu je bolja preglednost i čitljivost u odnosu na JSP. Takođe, radi direktno u HTML stranicama (nad ekstenzijom .html, za razliku od JSP-ovog .jsp), čineći front-end dosta prirodnijim i intuitivnijim. Komunikacija između developera i dizajnera je tako znatno olakšana, pa je i produktivnost rada bolja.

U ovom poglavlju pokrivene su sve važne funkcije i prednosti korišćenja projekta Spring Boot. Opisana je njegova slojevita arhitektura, proces obrade zahteva i komunikacija između softverskih komponenti, kao i prednosti korišćenja Maven alata za upravljanje zavisnostima. Na kraju, dat je kratak osvrt na spring početne projekte.

Iz svega navedenog, može se zaključiti da je Spring Boot pojednostavljena i automatizovana verzija Spring-ovog frameworka. Spring Boot-ov visok nivo apstrakcije njegovih potprojekata čini rad sa različitim Spring-ovim modulima dosta jednostavnijim i bržim. Olakšava početak izgradnje veb aplikacija zasnovanih na Spring-u, i kao takav je korišćen kao podloga za veb aplikaciju ovog završnog rada.

1.3.2. Spring Data

Ovaj projekat nudi dodatni apstrakcioni sloj za pristup skladištu podataka [2].

Glavni cilj mu je da olakša komunikaciju sa bazom podataka i korišćenje Spring-ovih modula sloja pristupa podataka (data access layer modula).

Sadrži pregršt interfejsa koje možemo da proširimo (extend) kako bismo iskoristili prednost ugrađenih funkcionalnosti koje ovaj projekat nudi [2].

Olakšava korišćenje tehnologija pristupa podataka, relacionih i nerelacionih baza podataka (SQL i NoSQL baze podataka), radnih okvira sa smanjenim mapiranjem (map-reduce), i servisa podataka zasnovanih na Cloud-u. Spring Data predstavlja projekat koji sadrži čitav niz potprojekata koji su specifični za datu bazu podataka [11].

Neke od važnih funkcionalnosti koje Spring Data nudi su [8]:

- Obezbeđuje podršku za integraciju sa prilagođenim repozitorijumskim kodom.
- Obezbeđuje apstrakcije objektnog mapiranja i repozitorijuma koristeći se mehanizmom generisanja dinamičnih upita iz naziva metoda repozitorijuma. Glavna prednost ovoga je izbegavanje nepotrebnog koda za pisanje upita prilikom izvlačenja podataka iz baze, čime je produktivnost rada znatno uvećana.
- Napredna integraciona podrška sa Spring MVC kontrolerima
- Podrška za transparentne funkcije za nadgledanje kao što je: created by, created date, last changed by i last changed date.
- Eksperimentalna podrška za napredni način mapiranja perzistentnih podataka u više od jednog skladišta podataka (cross-store persistence)

Spring Data obezbeđuje integraciju sa različitim specifikacijama i/ili skladištima podataka [7, 8]:

- JPA – Java Persistance API specifikacija definiše smernice i standard za organizaciju relacionih podataka, mehanizam objektno-relacionog mapiranja i perzistencije podataka. Jedna od JPA implementacije je radni okvir Hibernate. Spring Data JPA je apstrakcija nad JPA-om i koristi Hibernate kao glavni JPA provajder u Spring Boot veb aplikacijama, te je u ovom završnom radu Spring Data JPA korišćen sa Hibernate-om kako bi obezbedio kompletnu i olakšanu kontrolu nad slojem pristupa podataka.
- JDBC
- MongoDB
- LDAP
- Gemfire
- REST
- Redis
- Apache Cassandra
- Apache Solr

Interfejsi repozitorijuma u Springu su glavne spone između aplikacije i baze podataka, i sadrže sve osnovne funkcije za komunikaciju sa bazom podataka. Dva glavna interfejsa koje Spring Data projekat nudi su:

- CrudRepository – obezbeđuje osnovne CRUD operacije nad entitetima (java POJO klasama koje predstavljaju tabele u relacionoj bazi podataka i označene su sa @Entity anotacijom)
- PagingAndSortingRepository – proširuje CrudRepository i dodaje metode za sortiranje i paginaciju entiteta. Paginacija je proces koji omogućava razdvajanje veb sadržaja na više pojedinačnih stranica. Samo određeni broj zapisa entiteta se može prikazati na jednoj stranici, čime je preglednost sadržaja poboljšana.

Oba ova interfejsa proširuju osnovni Spring Data repozitorijum – Repository, koji kao parametere uzima domensku klasu (entitet) za koju je namenjen i tip atributa ID-ja entiteta (atribut ID-ja je anotiran sa @Id u entitetu/klasi modela).

Pošto je CrudRepository igrao znatnu ulogu u veb aplikaciji ovog završnog rada, osvrnuo bih se na neke od metoda koje on nudi:

- save() – Kao parametar uzima pripremljenu instancu entiteta i čuva ga u bazi podataka. Vraća sačuvanu instancu radi daljih operacija, pošto je moguće da je sačuvana instanca promenjena. Ova metoda se koristi za operacije kreiranja nove instance i ažuriranja postojeće instance entiteta, odnosno klase modela (CREATE i UPDATE operacije).
- findById() – Kao parametar uzima id entiteta i vraća jedan zapis entiteta sa datim id-jem. Entitet koji vraća je opcioni, navodi se kao parametar klase omotača (wrapper class) java.util.Optional pa ukoliko ne pronađe nijedan zapis, vraća praznu instancu Optional klase. Može se zaključiti da ova metoda pripada operaciji READ, po parametru id. U pozadini kreira Select upit.
- findAll() – Vraća sve zapise entiteta iz baze podataka.
- deleteById() – Kao parametar uzima id entiteta, i briše zapis entiteta po zadatom id-ju iz baze podataka.

Ove metode nemaju svoju implementaciju. Implementacija će biti odrađena od strane radnog okvira tokom izvršenja aplikacije. Kada klijent pozove neku od metoda repozitorijuma, automatski će se generisati i izvršiti upit nad bazom podataka, koristeći se parametrima koji su zadati repozitorijumu i konvencionalnim nazivima njegovih metoda.

Nakon opisa dva glavna repozitorijuma Spring Data projekta, spomenuo bih još jedan repozitorijumski interfejs koji nasleđuje sve funkcije CrudRepository-ja i PagingAndSortingRepository-ja, a to je JpaRepository.

JpaRepository proširuje PagingAndSortingRepository koji proširuje CrudRepository, te ukoliko su neophodne funkcionalnosti oba ova interfejsa koristiće se JpaRepository. Pored pune funkcionalnosti oba interfejsa, JpaRepository obezbeđuje i neke metode vezane za JPA, kao što je takozvano ispiranje (flushing) perzistentnog konteksta i brisanje ogromne količine (batch) zapisa iz baze podataka.

Kao što je već spomenuto, Spring Data JPA je potprojekat od Spring Data, koji obezbeđuje apstrakcioni sloj nad JPA-om. Preko svojih JPA repozitorijuma, odnosno seta interfejsa definisanih iznad, uvodi takozvane metode za izvođenje upita, koje imaju konvencionalni standard imenovanja. Osnovna sintaksa imenovanja je findBy<ime atributa>, na primer: findByName(String name). Prvi deo – find – predstavlja uvodnik (prilikom izvođenja upita postaće SQL-ov Select), a nastavak – kao što je ByName – je kriterijum upita i u ovom slučaju će vratiti entitete sa atributom Name koji odgovara parametru name.

Spring Data JPA, dakle, nudi dva načina definisanja upita nad entitetima baze podataka:

- Definisanje upita preko deklarisanja naziva metoda u repozitorijumima čime Spring Data JPA automatski generiše upit
- Definisanjem sopstvenih JPQL (Java Persistance Query Language) ili prirodnih upita (upita pomoću SQL) koristeći anotaciju @Query. Glavna razlika korišćenja JPQL jezika za pravljenje upita je da on radi direktno nad modelom objekta entiteta definisanog u Java aplikaciji, čime olakšava pisanje upita i nudi bolju preglednost, dok prirodni upiti rade nad tabelama u bazi podataka.

Prvi način definisanja upita je odličan način za izvođenje jednostavnijih upita. Ali, čim se javi potreba za upotrebom više od dva parametara upita, ili upit postane malo kompleksniji, treba da se koristi drugi navedeni način. Razlog tome je zbog naziva upita, odnosno metode, koji postane suviše komplikovan za pisanje i čitanje, ili su prevaziđene mogućnosti parsera naziva metode.

U ovom poglavlju opisao sam šta je Spring Data projekat i njegov značaj u izgradnji i podršci sloju pristupa podataka.

Dat je kratak opis funkcionalnosti koje on nudi i navedena podrška u integraciji sa različitim specifikacijama i skladištima podataka. Zatim, navedeni su glavni repozitorijumski interfejsi, funkcionalnosti koje oni nude, kao i kratak opis nekih od konkretnih njihovih metoda. Potom, opisan je jedan od glavnih potprojekata u okviru Spring Data – Spring Data JPA. Poseban akcenat

stavljen je na deklarativne metode za izvođenje upita, način na koji se one imenuju i njihov značaj u izbegavanju pisanja nepotrebnog koda čime doprinose bržem razvoju aplikacija. Na kraju, navedena su dva načina definisanja upita u Spring Data JPA-ju, kao i različiti scenariji njihove upotrebe.

Funkcije Spring Data projekta, a pogotovo njegovog potprojekta Spring Data JPA, korišćene su u ogromnoj meri u organizaciji i izgradnji sloja podataka veb aplikacije ovog završnog rada.

1.3.3. Spring Security

Spring Security je moćan i izuzetno prilagodljiv radni okvir autentifikacije i kontrole pristupa. U praksi predstavlja standard za obezbeđivanje svih aplikacija zasnovanih na Springu [12]. Prava snaga Spring Security-ja se ogleda u njegovoj visokoj fleksibilnosti i mogućnosti proširenja kako bi zadovoljio zahteve različitih kompleksnosti.

Za bolje razumevanje Spring Security projekta, potrebno je biti upoznat sa njegovom terminologijom, odnosno bezbednosnim pojmovima i konceptima kojima se on služi [6]:

- Principal – Korisnik aplikacije: Bilo koji korisnik, uređaj ili sistem (aplikacija) koji bi želeo da stupi u interakciju sa Spring Security veb aplikacijom.
- Autentifikacija: Proces pomoću kojeg se aplikacija uverava da je korisnik aplikacije onaj koji tvrdi da jeste, odnosno proces utvrđivanja identiteta korisnika.
- Kredencijali: Kada korisnik aplikacije pokuša da stupi u interakciju sa veb aplikacijom, pokreće se proces autentifikacije koji traži od njega da priloži određene vrednosti. Jedan od primera je unos kombinacija korisničkog imena i lozinke, i ove vrednosti/podaci se nazivaju kredencijali. Proces autentifikacije testira podudaranje unetih vrednosti sa podacima iz skladišta podataka i vraća odgovarajući rezultat.
- Autorizacija: Posle uspešne autentifikacije, korisnik aplikacije se ponovo proverava za akcije koje može da odradi nad veb aplikacijom. Ovaj proces provere prava koja ima korisnik aplikacije, a potom davanje potrebnih dozvola se zove autorizacija.
- Osigurana stavka/resurs: Stavka ili resurs koji je označen kao osiguran i zahteva od korisnika da uspešno prođe procese autentifikacije i autorizacije.
- GrantedAuthority: Spring Security objekat (org.springframework.security.core.GrantedAuthority interfejs) koji u sebi sadrži detalje o dozvolama i pravima pristupa za korisnika.
- SecurityContext: Spring Security objekat koji sadrži korisnikove detalje autentifikacije.

Spring Security radi na četiri osnovna koncepta bezbednosti:

- Autentifikacija
- Autorizacija
- Skladištenje lozinki
- Sigurnosni Servlet filteri

Servlet filteri su Java klase koje se koriste da presretnu HTTP zahteve i odgovore aplikacije. Koriste se za izvršenje operacija u dva slučaja - Pre slanja zahteva kontroleru i pre slanja odgovora klijentu (web browser-u). Spring Security veb arhitektura bazirana je na standardnim servlet filterima.

Spring Security interno održava sigurnosi lanac filtera. Svaki od filtera ima posebnu odgovornost i filteri se dodaju ili izbacuju iz konfiguracije, odnosno lanca, u zavisnosti koji su servisi neophodni [14].

Zahvaljujući Spring Boot-ovom početnom projektu za sigurnost – spring-boot-starter-security – obezbeđena je automatska konfiguracija za Spring Security, te eksplicitno definisanje sigurnosnog lanca filtera, u vidu posebnih Spring bean-ova, nije neophodno, osim u specijalnim slučajevima. Time je rad sa Spring Security radnim okvirom znatno olakšan.

Neke od glavnih funkcija koje Spring Security nudi su [8]:

- Obimna podrška za autentifikaciju i autorizaciju
- Podrška za integraciju sa Servlet API-jima i Spring MVC-om
- Moduli za integraciju sa SAML-om (Security Assertion Markup Language) i LDAP-om (Lightweight Directory Access Protocol) mehanizmima autentifikacije
- Podrška za JAAS (Java Authentication and Authorization Service) standard za autentifikaciju i autorizaciju
- Obezbeđuje odbranu od uobičajenih bezbednosnih napada kao što su Cross-site request forgery (CSRF), session fixation ("fiksiranje sesije"), clickjacking, itd.

Jedan od najpoznatijih napada za koje Spring Security nudi protekciju je CSRF (Cross-Site Request Forgery) napad. CSRF je napad koji primorava krajnjeg korisnika da izvrši neželjene radnje na veb aplikaciji u kojoj je trenutno autentifikovan. Napadač koristi socijalni inženjering, kao što je slanje linka preko email-a ili chat-a. Ukoliko je žrtva običan korisnik, uspešan CSRF napad može da primora korisnika da pokrene HTTP zahteve promene stanja, kao što je transfer sredstava, promena email adrese, itd. Ukoliko je žrtva administrativni nalog, CSRF može da kompromituje čitavu veb aplikaciju.

U Spring Boot Security početnom projektu, podrazumevana konfiguracija obezbeđuje protekciju od CSRF napada. To radi tako što generiše CSRF token za svaki zahtevani pogled (view), koji server validira pre početka obrade zahteva, i odbija zahtev ukoliko token nedostaje ili je nevažeći.

Sada će biti opisan mehanizam autentifikacije koji se koristio u veb aplikaciji ovog završnog rada. Korišćena je autentifikacija na osnovu provere podataka korisnika iz relacione baze. Ključni termini na koje treba obraditi pažnju su UserDetails i UserDetailsService interfejs, kao i DaoAuthenticationProvider klasa.

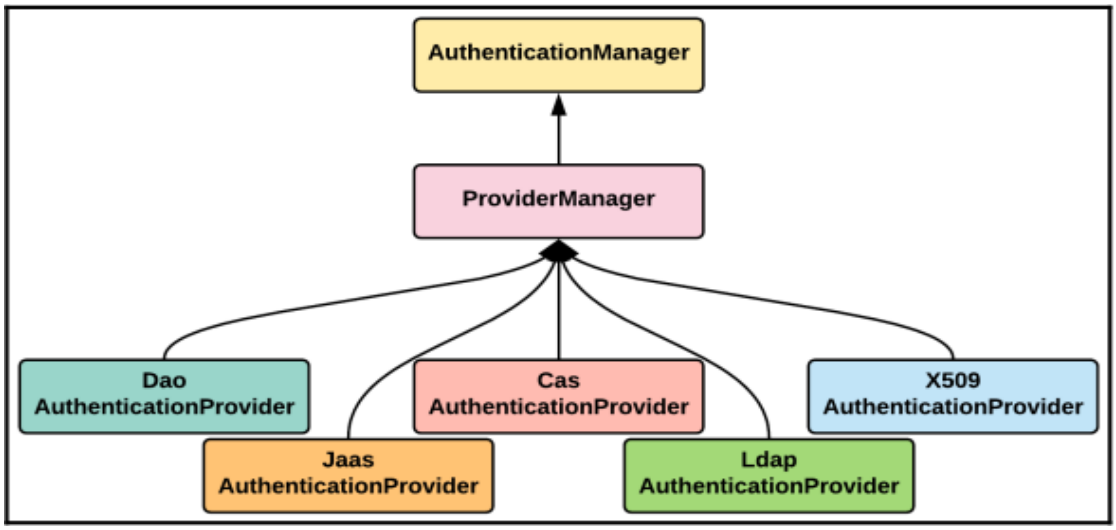
U veb aplikaciji ovog završnog rada, autentifikacija je odrađena kroz Spring Security API-je uz pomoć Spring Data JPA sloja pristupa podataka i MySQL relacionom bazom podataka. Implementirana su dva interfejsa Spring Security-ja:

- UserDetails – Implementacija ovog interfejsa sadrži osnovne informacije o korisniku za autentifikaciju, kao što je lozinka i korisničko ime, ali i listu GrantedAuthority objekata sa podacima vezanim za pravo pristupa resursima veb aplikacije. Pored toga, sadrži i razne bezbednosne indikatore o korisničkom nalogu, na primer da li je nalog istekao, da li je nalog zaključan itd. Nakon autentifikacije, informacije koje se ovde skladište se enkapsuliraju u Authentication objekat koji se koristi za izvlačenje informacija o ulogovanom korisniku tokom trajanja sesije. Zbog toga se implementacija UserDetails-a koristi i za skladištenje dodatnih informacija o ulogovanom korisniku koje nisu vezane za sigurnost.
- UserDetailsService – Implementacija ovog interfejsa se koristi za izvlačenje podataka korisnika koji želi da se prijavi. Sadrži jednu metodu – loadUserByUsername() koja može da se pregazi (override) kako bi se prilagodio proces pronalaženja korisnika iz skladišta podataka. Ova metoda vraća instancu UserDetails implementacije interfejsa u kojoj su smešteni podaci korisnika. Koristi se kao strategija DaoAuthenticationProvider-a da učita podatke o korisniku tokom autentifikacije.

Pomenuti DaoAuthenticationProvider je jedan od mnogih autentifikacionih snabdevača (provajdera), čija je svrha da izvrše specifičnu autentifikaciju, u zavisnosti od tehnologija i autentifikacionog mehanizma koji se koriti. Jedan je od više implementacija

AuthenticationProvider interfejsa koji se nalaze u sklopu Spring Security-jevog ProviderManager-a. ProviderManager klasa u sebi sadrži listu konfigurisanih AuthenticationProvider interfejsa. Kada dođe do procesa autentifikacije, ProviderManager klasa se koristi za prolazak kroz listu provajdera, kako bi utvrdila koji provajder je pogodan za izvršenje specifične autentifikacije. U slučaju uspešne autentifikacije korisnika, autentifikacioni provajder će vratiti objekat klase Authentication koji sadrži podatke o ulogovanom korisniku.

Na slici 6 može se videti koje to ugrađene AuthenticationProvider implementacije nudi Spring Security.



Slika 6. Spring Security ugrađeni AuthenticationProvider-i [6]

DaoAuthetnicationProvider se najčešće koristi za autentifikaciju korisnika na osnovu datog korisničkog imena (username-a) i lozinke. Koristi UserDetailsService interfejs implementaciju kako bi učitao korisnikove podatke. Takođe, koristi se i za postavljanje takozvanog PasswordEncoder-a koji se definiše radi enkriptovanja lozinke u bazi podataka. U sebi sadrži implementovanu metodu authenticate(), koja, u slučaju uspešnog procesa autentifikacije, vraća popunjeni Authentication objekat, koji sadrži sve informacije prethodno definisane u UserDetails implementaciji za ulogovanog korisnika. Authentication objekat se zatim skladišti u SecurityContext objektu, radi dalje upotrebe tokom rada korisnika na aplikaciji.

Konfigurisanje DaoAuthenticationProvider-a u veb aplikaciji, kao i ostalih funkcija Spring Security-ja, kao što je autorizacija, definisanje PasswordEncoder bean-a itd., se radi u Spring Security konfiguracionoj klasi koja proširuje WebSecurityConfigurerAdapter klasu. Proširenjem ove klase mogu se pregaziti (override) razne metode za konfigurisanje raznih koncepata Spring Security-ja, od kojih su najvažniji autentifikacija i autorizacija.

Implementacijom Spring Security konfiguracione klase u Spring Boot projektu, prevazilazimo automatski konfigurisanu bezbednost Spring Boot Security početnog projekta. Ova klasa označena je sa @EnableWebSecurity anotacijom koja omogućava Spring Security-jevu podršku za web sigurnost i obezbeđuje integraciju sa Spring MVC-om.

U ovoj konfiguracionoj klasi posebno bih napomenuo sledeću metodu: protected void configure(HttpSecurity http). Ova metoda se koristi za podešavanje autorizacije, tako što osigurava različite URL-ove veb aplikacije. Koristi se takozvanim Web URL pristupom autorizaciji. Web URL pristup omogućava da se kontrolišu resursi aplikacije na osnovu različitih URL-ova ili URL šablona. Nakon definisanja URL-ova, definiše se tip (uloga) korisnika aplikacije koji može da im pristupi. Web URL tip autorizacije korišćen je u veb aplikaciji ovog završnog rada.

U ovom poglavlju opisao sam šta je to Spring Security projekat i kakvu on igra ulogu i značaj u izgradnji Spring Veb aplikacija.

Najpre, dat je osvrt na terminologiju i pojmove kojima se on služi, i navedeni su glavni koncepti bezbednosti za koje nudi rešenja. Zatim, kratko je opisana njegova veb arhitektura, sastojana iz lanca sigurnosnih filtera. Navedene su neke od glavnih funkcija koje Spring Security nudi, osvrćući se i na bezbednosne napade za koje nudi zaštitu. Poseban akcenat stavljen je na CSRF napad.

U nastavku, osvrnuo sam se na mehanizam autentifikacije korišćen u ovom završnom radu. Opisane su njegove glavne komponente u okviru Spring Security-ja, navedeni interfejsi koji se koriste, kao i pojam autentifikacionog provajdera, koja je njegova svrha i kako omogućava izvršenje mehanizma autentifikacije.

Na kraju, dat je kratak opis Spring Security konfiguracione klase i čemu ona služi. U okviru nje, posebno je opisana metoda za implementaciju autorizacije, kao i Web URL pristup autorizaciji i kako on funkcioniše.

Iz svega navedenog, može se zaključiti koliko je Spring Security opsežan radni okvir, i koliki zapravo apstrakcijski sloj bezbednosti nudi u cilju olakšane i brže implementacije sigurnosti aplikacije.

2. PROJEKTOVANJE I IMPLEMENTACIJA JAVA VEB APLIKACIJE ZA PODRŠKU POSLOVANJU PLANINARSKOG KLUBA "SUMMIT EXPLORERS"

Razvoj studentskog primera ove aplikacije će biti realizovan korišćenjem Larmanove metode razvoja softvera, koja se sastoji od sledećih faza:

- Specifikacija zahteva
- Analiza
- Projektovanje
- Implementacija
- Testiranje

2.1. SPECIFIKACIJA ZAHTEVA

Specifikacija zahteva je prva faza Larmanove metode u kojoj se akcenat stavlja na intenzivnu saradnju između projektanata i stručnjaka koji poznaju domen problema. Cilj je napraviti verbalni model sistema, a zatim iz njega izvući i opisati korisničke zahteve koje softver treba da ispuni, koristeći se modelom slučajeva korišćenja (eng. Use-case).

2.1.1. Verbalni model

Potrebno je projektovati i implementirati veb aplikaciju koja će u osnovi omogućiti rezervaciju planinarskih putovanja i naručivanje planinarske sportske opreme u online prodavnici. Postoji više korisnika aplikacije: gosti, registrovani korisnici, administratori i menadžeri.

Gosti aplikacije su korisnici koji nisu registrovani. Pored toga što mogu da se registruju na platformu, imaju mogućnost da pregledaju brojne aranžmane putovanja koja predstavljaju razne aktivnosti na otvorenom kao što su planinarske ture, trekking, hiking, kajaking, rafting itd. Klikom na neki od aranžmana gostima se omogućava detaljan pregled aranžmana: opis putovanja, plan i program, način organizacije, zahtevnost ture itd. Takođe, imaju uvid u sve polaske i cene polazaka u narednom periodu za izabrani aranžman, kao i pregled galerije slika. U online prodavnici, gosti imaju mogućnost pregleda svih proizvoda – planinarske sportske opreme, mogućnost sortiranja na osnovu cene proizvoda. Klikom na neki od proizvoda pristupaju delu detaljnog pregleda proizvoda, gde mogu videti opis, benefite, tehničke informacije i galeriju slika.

Registrovanim korisnicima je nakon prijave na sistem, pored pregleda aranžmana i njihovih polazaka, omogućeno da izvrše rezervaciju polazaka. Imaju mogućnost pregleda svojih rezervacija polazaka, kako aktuelnih tako i prošlih, kao i otkazivanje rezervacija za predstojeće polaske. U online prodavnici registrovan korisnik može da izvrši naručivanje proizvoda. Svaki proizvod se prvo stavlja u korisnikovu korpu proizvoda kao stavka korpe, odakle korisnik može da briše ili dodaje nove stavke. Kada se odluči na poručivanje, korisnik ide na "checkout" stranicu gde upisuje podatke o dostavi. Odatle, korisnik pravi narudžbenicu. Registrovan korisnik ima i mogućnost pregleda istorije svojih narudžbenica. Prilikom registracije korisniku se dodelju i članska kartica planinarskog kluba na kojoj korisnik može da skuplja bodove. Ove bodove korisnik prikuplja prilikom svake naredne napravljene narudžbenice u prodavnici. Kada skupi dovoljan broj bodova, korisnik ostvaruje pravo na popuste prilikom sledećih naručivanja, koje može ili ne mora da iskoristi. Što više bodova skupi, ostvaruje pravo na veći popust. Registrovan korisnik takođe može da pregleda svoje lične podatke i da ih izmeni.

Administratori sistema imaju potpunu kontrolu nad aplikacijom. Mogu da pregledaju, izmene, obrišu ili dodaju nove proizvode, aranžmane i njihove polaske i rezervacije polazaka. Mogu da pregledaju ili obrišu korisnike. Imaju i kontrolu nad podacima članskih kartica korisnika. Za aranžmane i proizvode je omogućen pregled, dodavanje i brisanje slika iz galerije slika. Takođe, omogućena im je pretraga kako korisnika tako i proizvoda.

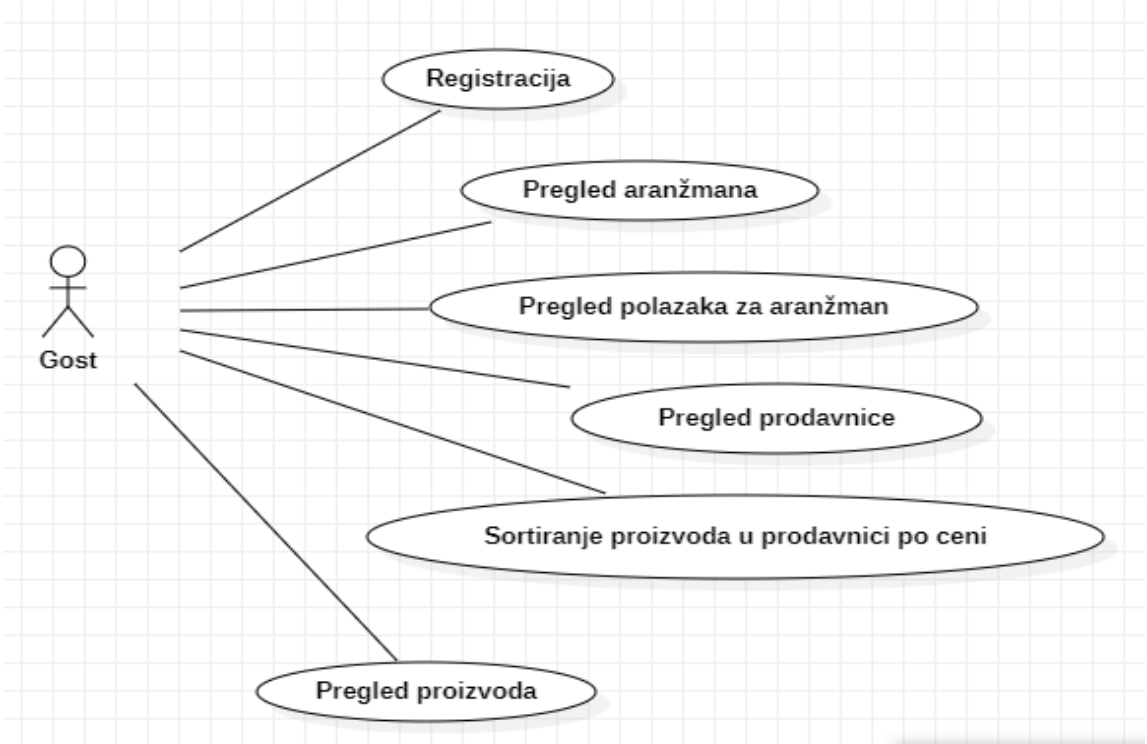
Menadžeri imaju uvid u razne izveštaje poslovanja planinarskog kluba. Mogu da pregledaju proizvode gde imaju uvid u njihovu uspešnost prodaje na osnovu raznih parametara kao što su broj prodatih jedinica, dosadašnja ukupna zarada, količina u magacinu itd. Imaju uvid u izveštaje budućih i prošlih polazaka aranžmana i njihovu uspešnost, kao i uspešnost na nivou celih aranžmana. U svim izveštajima dostupnim menadžeru su prikazani razni parametri poslovanja po kojima menadžer može da sortira rezultate izveštaja na način koji njemu odgovara, i tako iznese zaključke, izvrši procene i donose odluke o unapređenju poslovanja planinarskog kluba.

2.1.2. Slučajevi korišćenja za gosta

Na osnovu verbalnog modela uočeni su sledeći slučajevi korišćenja za gosta aplikacije:

- Registracija
- Pregled aranžmana
- Pregled polazaka za aranžman
- Pregled prodavnice
- Sortiranje proizvoda u prodavnici po ceni
- Pregled proizvoda

Dijagram slučajeva korišćenja (eng. use-case) za gosta aplikacije prikazan je na slici 7.



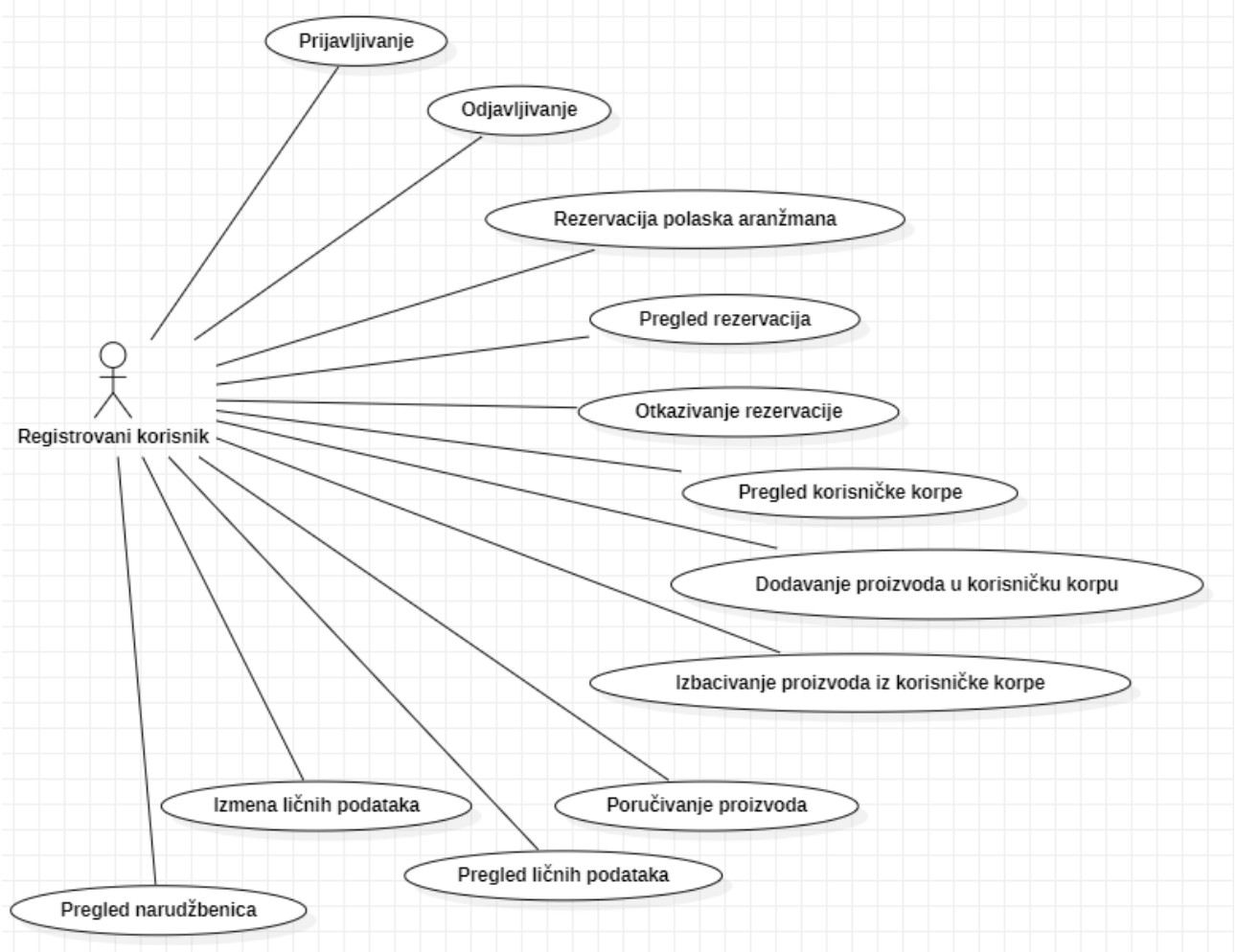
Slika 7. Dijagram slučajeva korišćenja – Gost

2.1.3. Slučajevi korišćenja za registrovanog korisnika

Na osnovu verbalnog modela uočeni su sledeći slučajevi korišćenja za registrovanog korisnika:

- Prijavljivanje
- Odjavljivanje
- Rezervacija polaska aranžmana
- Pregled rezervacija
- Otkazivanje rezervacije
- Pregled korisničke korpe
- Dodavanje proizvoda u korisničku korpu
- Izbacivanje proizvoda iz korisničke korpe
- Poručivanje proizvoda
- Pregled ličnih podataka
- Izmena ličnih podataka
- Pregled narudžbenica

Dijagram slučajeva korišćenja (eng. use-case) za registrovanog korisnika prikazan je na slici 8.



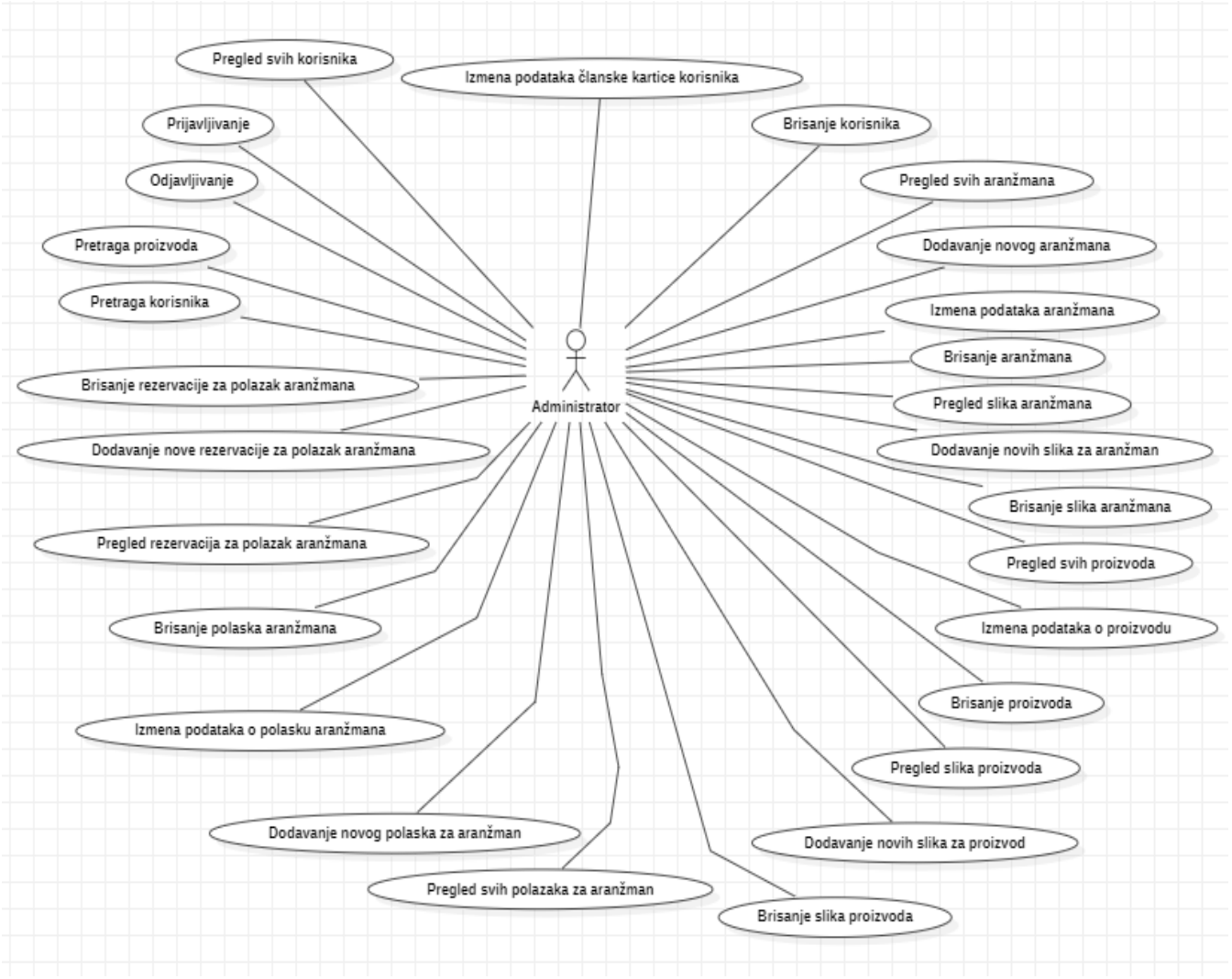
Slika 8. Dijagram slučajeva korišćenja – Registrovan korisnik

2.1.4. Slučajevi korišćenja za administratora

Na osnovu verbalnog modela uočeni su sledeći slučajevi korišćenja za administratora:

- Prijavljivanje
- Odjavljivanje
- Pregled svih korisnika
- Izmena podataka članske kartice korisnika
- Brisanje korisnika
- Pregled svih aranžmana
- Dodavanje novog aranžmana
- Izmena podataka aranžmana
- Brisanje aranžmana
- Pregled slika aranžmana
- Dodavanje novih slika za aranžman
- Brisanje slika aranžmana
- Pregled svih proizvoda
- Izmena podataka o proizvodu
- Brisanje proizvoda
- Pregled slika proizvoda
- Dodavanje novih slika za proizvod
- Brisanje slika proizvoda
- Pregled svih polazaka za aranžman
- Dodavanje novog polaska za aranžman
- Izmena podataka o polasku aranžmana
- Brisanje polaska aranžmana
- Pregled rezervacija za polazak aranžmana
- Dodavanje nove rezervacije za polazak aranžmana
- Brisanje rezervacije za polazak aranžmana
- Pretraga korisnika
- Pretraga proizvoda

Dijagram slučajeva korišćenja (eng. use-case) za administratora prikazan je na slici 9.



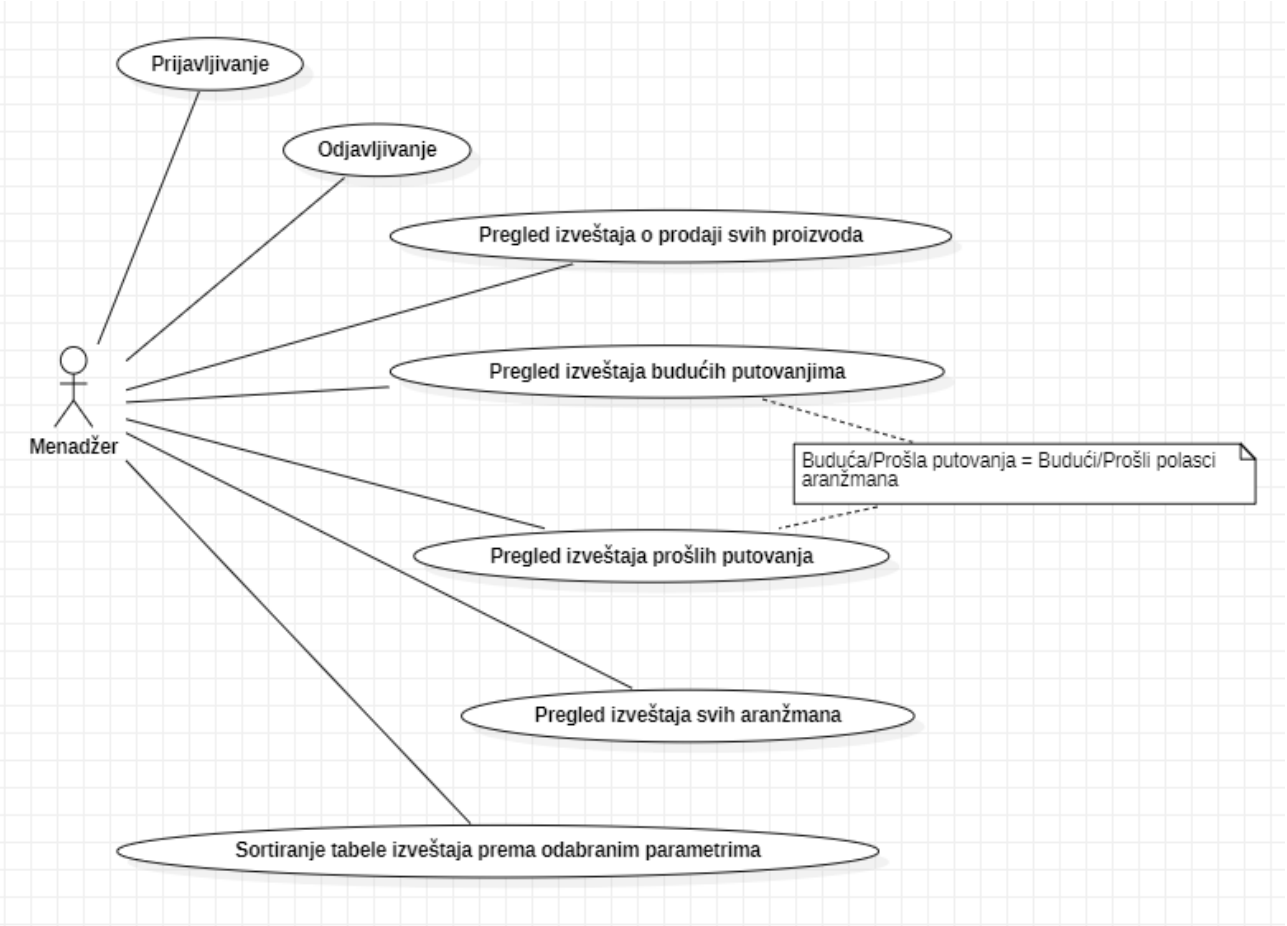
Slika 9. Dijagram slučajeva korišćenja – Administrator

2.1.5. Slučajevi korišćenja za menadžera

Na osnovu verbalnog modela uočeni su sledeći slučajevi korišćenja za menadžera:

- Prijavljivanje
- Odjavljivanje
- Pregled izveštaja o prodaji svih proizvoda
- Pregled izveštaja budućih putovanja
- Pregled izveštaja prošlih putovanja
- Pregled izveštaja svih aranžmana
- Sortiranje tabele izveštaja prema odabranim parametrima

Dijagram slučajeva korišćenja (eng. use-case) za menadžera prikazan je na slici 10.



Slika 10. Dijagram slučajeva korišćenja – Menadžer

2.1.6. Opis slučajeva korišćenja za gosta

Nakon predstavljenih dijagrama slučajeva korišćenja za svakog korisnika sistema (aktora), ovde će biti dat detaljan opis svakog pojedinačnog slučaja korišćenja (SK) koji se sastoji od naziva, aktora, učesnika, preduslova, osnovnog i alternativnog/ih scenarija. U ovom poglavlju predstavljeni su opisi slučajeva korišćenja za gosta aplikacije.

SK1: Registracija

Naziv: Registracija

Aktori: Gost

Učesnici: Gost i sistem

Preduslovi: Sistem je aktivan i prikazana je veb strana forme za registraciju

Osnovni scenario:

1. Gost unosi podatke neophodne za registraciju (APUSO)
2. Gost poziva sistem da obradi podatke za registraciju (APSO)
3. Sistem obrađuje unete podatke i proverava validnost (SO)

4. Sistem prikazuje gostu poruku o uspešnoj registraciji (IA)

Alternativna scenarija:

3.1. Gost je uneo već postojeću email adresu, sistem prikazuje odgovarajuću poruku gostu (IA)

3.2. Sistem je utvrdio da gost nije uneo validne podatke za registraciju i prikazuje odgovarajuću poruku gostu (IA)

3.3. Sistem ne može da obradi unete podatke i prikazuje odgovarajuću poruku gostu (IA)

SK2: Pregled aranžmana

Naziv: Pregled aranžmana

Aktori: Gost

Učesnici: Gost i sistem

Preduslovi: Sistem je aktivan, prikazana je stranica izabrane vrste putovanja

Osnovni scenario:

1. Gost poziva sistem da prikaže izabran aranžman (APSO)
2. Sistem pretražuje podatke o izabranom aranžmanu (SO)
3. Sistem prikazuje gostu detalje o traženom aranžmanu (IA)

Alternativni scenario:

2.1. Sistem ne može da pronađe podatke o izabranom aranžmanu , prikazuje poruku gostu (IA)

SK3: Pregled polazaka za aranžman

Naziv: Pregled polazaka za aranžman

Aktori: Gost

Učesnici: Gost i sistem

Preduslovi: Sistem je aktivan i prikazani su detalji o aranžmanu

Osnovni scenario:

1. Gost poziva sistem da prikaže polaske za izabrani aranžman (APSO)
2. Sistem pretražuje polaske za izabrani aranžman (SO)
3. Sistem prikazuje gostu listu polazaka za izabrani aranžman (IA)

Alternativni scenario:

2.1. Sistem ne može da pronađe nijedan polazak za izabrani aranžman i prikazuje odgovarajuću poruku gostu (IA)

SK4: Pregled prodavnice

Naziv: Pregled prodavnice

Aktori: Gost

Učesnici: Gost i sistem

Preduslovi: Sistem je aktivan

Osnovni scenario:

1. Gost poziva sistem da prikaže prodavnicu (APSO)
2. Sistem obrađuje zahtev za prikazom prodavnice (SO)
3. Sistem prikazuje gostu prodavnicu (IA)

Alternativni scenario:

2.1. Sistem ne može da obradi zahtev za prikaz prodavnice i prikazuje gostu odgovarajuću poruku (IA)

SK5: Sortiranje proizvoda u prodavnici po ceni

Naziv: Sortiranje proizvoda u prodavnici po ceni

Aktori: Gost

Učesnici: Gost i sistem

Preduslovi: Sistem je aktivan i prikazani su detalji prodavnice

Osnovni scenario:

1. Gost bira opciju po kojoj želi da sortira proizvode - cena rastuća ili opadajuća (APSO)
2. Sistem sortira proizvode po izabranoj opciji (SO)
3. Sistem prikazuje gostu sortirane proizvode u prodavnici po izabranoj opciji (IA)

Alternativni scenario:

2.1. Sistem ne može da sortira proizvode po izabranoj opciji, prikazuje odgovarajuću poruku gostu (IA)

SK6: Pregled proizvoda

Naziv: Pregled proizvoda

Aktori: Gost

Učesnici: Gost i sistem

Preduslovi: Sistem je aktivan i prikazani su detalji prodavnice

Osnovni scenario:

1. Gost poziva sistem da prikaže detalje izabranog proizvoda (APSO)
2. Sistem obrađuje zahtev za prikazom detalja izabranog proizvoda (SO)
3. Sistem prikazuje gostu detalje izabranog proizvoda (IA)

Alternativni scenario:

2.1. Sistem ne može da prikaže detalje izabranog proizvoda i prikazuje poruku gostu (IA)

2.1.7. Opis slučajeva korišćenja za registrovanog korisnika

Nakon predstavljenih dijagrama slučajeva korišćenja za svakog korisnika sistema (aktora), ovde će biti dat detaljan opis svakog pojedinačnog slučaja korišćenja (SK) koji se sastoji od naziva, aktora, učesnika, preduslova, osnovnog i alternativnog/ih scenarija. U ovom poglavlju predstavljeni su opisi slučajeva korišćenja za registrovanog korisnika.

SK7: Prijavljivanje

Naziv: Prijavljivanje

Aktori: Registrovan korisnik, Administrator, Menadžer

Učesnici: Registrovan korisnik, administrator, menadžer i sistem

Preduslovi: Sistem je aktivan i prikazana je forma za prijavljivanje

Osnovni scenario:

1. Korisnik unosi lične podatke vezane za prijavu na sistem (email, password) (APUSO)
2. Korisnik poziva sistem da izvrši prijavljivanje sa unetim podacima (APSO)
3. Sistem izvršava prijavljivanje korisnika (SO)
4. Sistem prikazuje korisniku početnu stranicu (IA)

Alternativna scenarija:

3.1. Sistem ne može da pronađe korisnika sa unetim podacima i prikazuje korisniku odgovarajuću poruku (IA)

3.2. Sistem je pronašao korisnika sa unetim podacima, ali ne može da izvrši prijavljivanje, prikazuje korisniku odgovarajuću poruku (IA)

SK8: Odjavljivanje

Naziv: Odjavljivanje

Aktori: Registrovan korisnik, administrator, menadžer

Učesnici: Registrovan korisnik, administrator, menadžer i sistem

Preduslovi: Sistem je aktivan i korisnik je prijavljen

Osnovni scenario:

1. Korisnik poziva sistem da izvrši odjavljivanje (APSO)
2. Sistem izvršava proces odjavljivanja korisnika (SO)
3. Sistem prikazuje korisniku formu za prijavljivanje (IA)

Alternativni scenario:

2.1. Sistem ne može da izvrši proces odjavljivanja, prikazuje korisniku odgovarajuću poruku (IA)

SK9: Rezervacija polaska aranžmana

Naziv: Rezervacija polaska aranžmana

Aktori: Registrovan korisnik

Učesnici: Registrovan korisnik i sistem

Preduslovi: Sistem je aktivan, korisnik je prijavljen i prikazani su detalji aranžmana sa listom polazaka

Osnovni scenario:

1. Korisnik poziva sistem da rezerviše odabrani polazak aranžmana (APSO)
2. Sistem obrađuje zahtev za rezervacijom odabranog polaska aranžmana (SO)
3. Sistem prikazuje detalje rezervacije odabranog polaska aranžmana (IA)
4. Korisnik pregledava detalje svoje rezervacije (ANSO)
5. Korisnik potvrđuje sistemu rezervaciju polaska (APSO)
6. Sistem kreira novu rezervaciju polaska aranžmana (SO)
7. Sistem prikazuje korisniku potvrdu rezervacije polaska aranžmana (IA)

Alternativna scenarija:

2.1. Sistem je utvrdio da korisnik ima već rezervaciju za odabrani polazak aranžmana, prikazuje odgovarajuću poruku korisniku i prekida izvršavanje scenarija (IA)

2.2. Sistem ne može da obradi zahtev za rezervacijom odabranog polaska aranžmana, prikazuje korisniku odgovarajuću poruku i prekida izvršavanje scenarija (IA)

5.1. Korisnik otkazuje proces rezervacije (APSO), sistem poništava proces rezervacije (SO), prikazuje korisniku listu svih polazaka aranžmana i prekida izvršavanje scenarija (IA)

6.1. Sistem ne može da kreira novu rezervaciju polaska aranžmana i prikazuje odgovarajuću poruku (IA)

SK10: Pregled rezervacija

Naziv: Pregled rezervacija

Aktori: Registrovan korisnik

Učesnici: Registrovan korisnik i sistem

Preduslovi: Sistem je aktivan, korisnik je prijavljen

Osnovni scenario:

1. Korisnik poziva sistem da prikaže sve njegove rezervacije (APSO)
2. Sistem obrađuje zahtev za prikazom svih korisnikovih rezervacija (SO)
3. Sistem prikazuje korisniku sve njegove rezervacije (IA)

Alternativni scenario:

2.1. Sistem ne može da obradi zahtev za prikazom svih korisnikovih rezervacija, prikazuje korisniku odgovarajuću poruku (IA)

SK11: Otkazivanje rezervacije

Naziv: Otkazivanje rezervacije

Aktori: Registrovan korisnik

Učesnici: Registrovan korisnik i sistem

Preduslovi: Sistem je aktivan, korisnik je prijavljen i prikazana je lista svih rezervacija polazaka za korisnika

Osnovni scenario:

1. Korisnik poziva sistem da otkaže odabranu rezervaciju (APSO)
2. Sistem pokreće proces otkazivanja rezervacije (SO)
3. Sistem traži od korisnika da potvrdi otkazivanja odabrane rezervacije (IA)
4. Korisnik potvrđuje otkazivanje odabrane rezervacije (APSO)
5. Sistem otkazuje odabranu rezervaciju polaska korisnika (SO)
6. Sistem prikazuje korisniku potvrdu o otkazivanju rezervacije (IA)

Alternativna scenarija:

2.1. Sistem ne može da pokrene proces otkazivanja rezervacije, prikazuje odgovarajuću poruku korisniku i prekida izvršavanje scenarija (IA)

4.1. Korisnik odustaje od otkazivanja rezervacije (APSO), sistem poništava proces otkazivanja (SO), prikazuje korisniku listu svih rezervacija polazaka i prekida izvršavanje scenarija (IA)

5.1. Sistem ne može da izvrši proces otkazivanja odabrane rezervacije i prikazuje korisniku odgovarajuću poruku (IA)

SK12: Pregled korisničke korpe

Naziv: Pregled korisničke korpe

Aktori: Registrovan korisnik

Učesnici: Registrovan korisnik i sistem

Preduslovi: Sistem je aktivan, korisnik je prijavljen

Osnovni scenario:

1. Korisnik poziva sistem da prikaže korisničku korpu (APSO)
2. Sistem obrađuje zahtev za prikazom korisničke korpe (SO)
3. Sistem prikazuje korisniku njegovu korisničku korpu sa svim njenim stavkama (IA)

Alternativni scenario:

2.1. Sistem ne uspeva da obradi zahtev za prikaz korisničke korpe, prikazuje korisniku odgovarajuću poruku (IA)

SK13: Dodavanje proizvoda u korisničku korpu

Naziv: Dodavanje proizvoda u korisničku korpu

Aktori: Registrovan korisnik

Učesnici: Registrovan korisnik i sistem

Preduslovi: Sistem je aktivan, korisnik je prijavljen i prikazani su detalji odabranog proizvoda iz prodavnice

Osnovni scenario:

1. Korisnik bira količinu odabranog proizvoda (APUSO)
2. Korisnik poziva sistem da ubaci odabrani proizvod u korpu u biranoj količini (APSO)
3. Sistem pokreće proces ubacivanja proizvoda u korpu u odabranoj količini (SO)
4. Sistem prikazuje korisniku poruku o uspešnosti dodavanja proizvoda u korisničku korpu (IA)

Alternativna scenarija:

1.1. Korisnik ne može da odabere količinu za odabrani proizvod jer proizvoda nema na stanju, sistem prikazuje odgovarajuću poruku i prekida izvršavanje scenarija (IA)

3.1. Sistem utvrđuje da je korisnik već dodao odabrani proizvod u korisničku korpu i da sa odabranom količinom prelazi limit količine proizvoda koju može da stavi u korpu, prikazuje odgovarajuću poruku korisniku (IA)

3.2. Sistem utvrđuje da je korisnik već dodao odabrani proizvod u korisničku korpu i da je već dostigao limit količine proizvoda koju može da stavi u korpu, prikazuje korisniku odgovarajuću poruku (IA)

3.3. Sistem ne može da obradi proces ubacivanja proizvoda u korpu, prikazuje odgovarajuću poruku korisniku (IA)

SK14: Izbacivanje proizvoda iz korisničke korpe

Naziv: Izbacivanje proizvoda iz korisničke korpe

Aktori: Registrovan korisnik

Učesnici: Registrovan korisnik i sistem

Preduslovi: Sistem je aktivan, korisnik je prijavljen i prikazana je korisnička korpa

Osnovni scenario:

1. Korisnik poziva sistem da izbaci odabrani proizvod iz korisničke korpe (stavku korpe) (APSO)
2. Sistem započinje proces izbacivanja proizvoda iz korisničke korpe (SO)
3. Sistem traži od korisnika da potvrdi izbacivanje proizvoda (IA)
4. Korisnik potvrđuje izbacivanje proizvoda iz korisničke korpe (APSO)
5. Sistem izbacuje proizvod iz korisničke korpe (SO)
6. Sistem prikazuje poruku o uspešnosti izbacivanja proizvoda iz korpe (IA)

Alternativna scenarija:

2.1. Sistem ne može da nastavi proces izbacivanja proizvoda iz korisničke korpe, prikazuje korisniku odgovarajuću poruku i prekida izvršavanje scenarija (IA)

4.1. Korisnik nije potvrdio izbacivanje proizvoda iz korisničke korpe, sistem prikazuje sve stavke korisničke korpe i prekida izvršavanje scenarija (IA)

5.1. Sistem ne može da izbaci proizvod iz korisničke korpe i prikazuje korisniku odgovarajuću poruku (IA)

SK15: Poručivanje proizvoda

Naziv: Poručivanje proizvoda

Aktori: Registrovan korisnik

Učesnici: Registrovan korisnik i sistem

Preduslovi: Sistem je aktivan, korisnik je prijavljen i prikazana je korisnička korpa

Osnovni scenario:

1. Korisnik bira jednu od ponuđenih opcija za popust prilikom naručivanja, u skladu sa bodovima na članskoj kartici (APSO)

2. Sistem izračunava novu cenu moguće porudžbine (SO)

3. Sistem prikazuje korisniku novu cenu moguće porudžbine (IA)

4. Korisnik potvrđuje sistemu poručivanje proizvoda iz korisničke korpe (APSO)

5. Sistem proverava dostupnost proizvoda nakon potvrde korisnika o poručivanju (SO)

6. Sistem prikazuje korisniku formu za unos podataka o dostavi (adresa, grad, poštanski broj) (IA)

7. Korisnik unosi podatke o dostavi (APUSO)

8. Korisnik poziva sistem da obradi podatke o dostavi i izvrši porudžbinu (APSO)

9. Sistem evidentira novu narudžbenicu za korisnika (SO)

10. Sistem prikazuje korisniku poruku o uspešno izvršenoj porudžbini (IA)

Alternativna scenarija:

1.1. Korisnik nije prikupio dovoljan broj bodova ni za jedan popust, sistem prikazuje stranicu bez opcija za popust (IA) i scenario se nastavlja od koraka 4

2.1. Sistem ne može da izračuna novu cenu moguće porudžbine, prikazuje korisniku odgovarajuću poruku i prekida izvršavanje scenarija (IA)

5.1. Sistem je utvrdio da jednog ili više proizvoda iz stavki korpe nema u dovoljnoj količini na stanju, prikazuje korisniku odgovarajuću poruku i prekida izvršavanje scenarija (IA)

8.1. Korisnik odustaje od izvršenja porudžbine, poziva sistem da prekine proces poručivanja (APSO), sistem obustavlja proces (SO), prikazuje korisniku korisničku korpu i prekida izvršavanje scenarija (IA)

9.1. Sistem je utvrdio da korisnik nije uneo validne podatke o dostavi, prikazuje korisniku odgovarajuću poruku (IA) i scenario se nastavlja od koraka 7

9.2. Sistem ne može da evidentira novu narudžbenicu za korisnika, prikazuje korisniku odgovarajuću poruku (IA)

SK16: Pregled ličnih podataka

Naziv: Pregled ličnih podataka

Aktori: Registrovan korisnik

Učesnici: Registrovan korisnik i sistem

Preduslovi: Sistem je aktivan, korisnik je prijavljen

Osnovni scenario:

1. Korisnik poziva sistem da prikaže njegove lične podatke (APSO)
2. Sistem obrađuje zahtev za prikazom ličnih podataka korisnika (SO)
3. Sistem prikazuje lične podatke korisnika (IA)

Alternativni scenario:

2.1. Sistem ne može da obradi zahtev za prikazom podataka, prikazuje korisniku odgovarajuću poruku (IA)

SK17: Izmena ličnih podataka

Naziv: Izmena ličnih podataka

Aktori: Registrovan korisnik

Učesnici: Registrovan korisnik i sistem

Preduslovi: Sistem je aktivan, korisnik je prijavljen i prikazana je forma ličnih podataka korisnika

Osnovni scenario:

1. Korisnik menja svoje lične podatke (APUSO)
2. Korisnik poziva sistem da evidentira unete promene (APSO)
3. Sistem evidentira unete promene podataka (SO)
4. Sistem prikazuje korisniku poruku o uspešnoj izmeni podataka (IA)

Alternativna scenarija:

3.1. Korisnik nije uneo validne izmenjene podatke, sistem prikazuje korisniku odgovarajuću poruku i prekida izvršavanje scenarija (IA)

3.2. Sistem ne može da evidentira unete podatke, prikazuje korisniku odgovarajuću poruku (IA)

SK18: Pregled narudžbenica

Naziv: Pregled narudžbenica

Aktori: Registrovan korisnik

Učesnici: Registrovan korisnik i sistem

Preduslovi: Sistem je aktivan, korisnik je prijavljen

Osnovni scenario:

1. Korisnik poziva sistem da prikaže sve njegove narudžbenice (APSO)
2. Sistem obrađuje zahtev za prikazom korisnikovih narudžbenica (SO)
3. Sistem prikazuje korisnikove narudžbenice (IA)

Alternativni scenario:

2.1. Sistem ne može da obradi zahtev za prikazom korisnikovih narudžbenica, prikazuje korisniku odgovarajuću poruku (IA)

2.1.8. Opis slučajeva korišćenja za administratora

Nakon predstavljenih dijagrama slučajeva korišćenja za svakog korisnika sistema (aktora), ovde će biti dat detaljan opis svakog pojedinačnog slučaja korišćenja (SK) koji se sastoji od naziva, aktora, učesnika, preduslova, osnovnog i alternativnog/ih scenarija. U ovom poglavlju predstavljeni su opisi slučajeva korišćenja za administratora.

SK19: Pregled svih korisnika

Naziv: Pregled svih korisnika

Aktori: Administrator

Učesnici: Administrator i sistem

Preduslovi: Sistem je aktivan, administrator je prijavljen

Osnovni scenario:

1. Administrator poziva sistem da prikaže listu svih korisnika (APSO)
2. Sistem obrađuje zahtev za prikazom liste svih korisnika (SO)
3. Sistem prikazuje listu svih korisnika (IA)

Alternativni scenario:

2.1. Sistem ne može da obradi zahtev za prikazom liste svih korisnika, prikazuje administratoru odgovarajuću poruku (IA)

SK20: Izmena podataka članske kartice korisnika

Naziv: Izmena podataka članske kartice korisnika

Aktori: Administrator

Učesnici: Administrator i sistem

Preduslovi: Sistem je aktivan, administrator je prijavljen i prikazana je forma podataka članske kartice izabranog korisnika

Osnovni scenario:

1. Administrator menja podatke članske kartice korisnika (APUSO)
2. Administrator poziva sistem da evidentira izmenjene podatke (APSO)
3. Sistem evidentira izmenjene podatke (SO)
4. Sistem prikazuje poruku o uspešnosti izmene podataka (IA)

Alternativna scenarija:

3.1. Sistem je utvrdio da izmenjeni podaci nisu validni, prikazuje administratoru odgovarajuću poruku (IA)

3.2. Sistem ne može da evidentira izmenjene podatke članske kartice korisnika, prikazuje administratoru odgovarajuću poruku (IA)

SK21: Brisanje korisnika

Naziv: Brisanje korisnika

Aktori: Administrator

Učesnici: Administrator i sistem

Preduslovi: Sistem je aktivan, administrator je prijavljen i prikazana je lista svih korisnika

Osnovni scenario:

1. Admnistrator bira korisnika koga želi da obriše (APSO)
2. Sistem pokreće proces brisanja korisnika (SO)
3. Sistem prikazuje dijalog za potvrdu o brisanju korisnika (IA)
4. Administrator potvrđuje sistemu brisanje izabranog korisnika (APSO)
5. Sistem briše izabranog korisnika (SO)
6. Sistem prikazuje poruku o uspešnom brisanju izabranog korisnika (IA)

Alternativna scenarija:

2.1. Sistem nije uspeo da pokrene proces brisanja korisnika, prikazuje administratoru odgovarajuću poruku i prekida izvršavanje scenarija (IA)

4.1. Administrator poziva sistem da poništi proces brisanja korisnika (APSO), sistem poništava proces brisanja korisnika (SO), prikazuje listu svih korisnika i prekida izvršavanje scenarija (IA)

5.1. Sistem ne može da obriše izabranog korisnika, prikazuje administratoru odgovarajuću poruku (IA)

SK22: Pregled svih aranžmana

Naziv: Pregled svih aranžmana

Aktori: Administrator

Učesnici: Administrator i sistem

Preduslovi: Sistem je aktivan, administrator je prijavljen

Osnovni scenario:

1. Administrator poziva sistem da prikaže listu svih aranžmana (APSO)
2. Sistem obrađuje zahtev za prikazom liste svih aranžmana (SO)
3. Sistem prikazuje listu svih aranžmana (IA)

Alternativni scenario:

2.1. Sistem ne može da obradi zahtev za prikazom liste svih aranžmana, prikazuje administratoru odgovarajuću poruku (IA)

SK23: Dodavanje novog aranžmana

Naziv: Dodavanje novog aranžmana

Aktori: Administrator

Učesnici: Administrator i sistem

Preduslovi: Sistem je aktivan, administrator je prijavljen i prikazana je lista svih aranžmana

Osnovni scenario:

1. Administrator poziva sistem da mu prikaže formu za dodavanje novog aranžmana (APSO)
2. Sistem obrađuje zahtev za prikazom forme (SO)
3. Sistem prikazuje formu za dodavanje novog aranžmana (IA)
4. Administrator unosi podatke novog aranžmana (APUSO)
5. Administrator poziva sistem da evidentira novi aranžman (APSO)
6. Sistem evidentira novi aranžman (SO)
7. Sistem prikazuje poruku o uspešnom dodatom novom aranžmanu (IA)

Alternativna scenarija:

2.1. Sistem ne može da prikaže formu za dodavanje novog aranžmana, prikazuje administratoru odgovarajuću poruku i prekida izvršavanje scenarija (IA)

5.1. Administrator odustaje od evidentiranja novog aranžmana (APSO), sistem poništava proces (SO), prikazuje administratoru listu svih aranžmana i prekida izvršavanje scenarija (IA)

6.1. Sistem je utvrdio da administrator nije uneo jedan ili više podataka u validnom formatu, prikazuje administratoru odgovarajuću poruku (IA) i scenario se nastavlja od koraka 4

6.2. Sistem ne može da evidentira novi aranžman, prikazuje administratoru odgovarajuću poruku (IA)

SK24: Izmena podataka aranžmana

Naziv: Izmena podataka aranžmana

Aktori: Administrator

Učesnici: Administrator i sistem

Preduslovi: Sistem je aktivan, administrator je prijavljen i prikazana je forma za izmenu podataka aranžmana

Osnovni scenario:

1. Administrator menja podatke izabranog aranžmana (APUSO)
2. Administrator poziva sistem da evidentira izmenjene podatke (APSO)
3. Sistem evidentira izmenjene podatke aranžmana (SO)
4. Sistem prikazuje poruku o uspešnoj izmeni podataka aranžmana (IA)

Alternativna scenarija:

2.1. Administrator odustaje od izmene podataka (APSO), sistem poništava proces (SO), prikazuje administratoru listu svih aranžmana i prekida izvršavanje scenarija (IA)

3.1. Sistem je utvrdio da administrator nije uneo jedan ili više podataka u validnom formatu, prikazuje administratoru odgovarajuću poruku i prekida izvršavanje scenarija (IA)

3.2. Sistem ne može da evidentira izmenjene podatke aranžmana, prikazuje administratoru odgovarajuću poruku (IA)

SK25: Brisanje aranžmana

Naziv: Brisanje aranžmana

Aktori: Administrator

Učesnici: Administrator i sistem

Preduslovi: Sistem je aktivan, administrator je prijavljen i prikazana je lista svih aranžmana

Osnovni scenario:

1. Administrator pregledava listu aranžmana i bira aranžman za brisanje (ANSO)
2. Administrator poziva sistem da obriše izabrani aranžman (APSO)
3. Sistem pokreće proces brisanja aranžmana (SO)
4. Sistem prikazuje dijalog za potvrdu o brisanju izabranog aranžmana (IA)
5. Administrator potvrđuje sistemu brisanje izabranog aranžmana (APSO)
6. Sistem briše izabrani aranžman (SO)
7. Sistem prikazuje poruku o uspešnosti procesa brisanja aranžmana (IA)

Alternativna scenarija:

3.1. Sistem ne može da pokrene proces brisanja aranžmana, prikazuje administratoru odgovarajuću poruku i prekida izvršavanje scenarija (IA)

5.1. Administrator poziva sistem da poništi proces brisanja aranžmana (APSO), sistem poništava proces (SO), prikazuje administratoru listu svih aranžmana i prekida izvršavanje scenarija (IA)

6.1. Sistem ne može da obriše izabrani aranžman, prikazuje administratoru odgovarajuću poruku (IA)

SK26: Pregled slika aranžmana

Naziv: Pregled slika aranžmana

Aktori: Administrator

Učesnici: Administrator i sistem

Preduslovi: Sistem je aktivan, administrator je prijavljen i prikazana je lista svih aranžmana

Osnovni scenario:

1. Administrator poziva sistem da prikaže slike izabranog aranžmana (APSO)
2. Sistem obrađuje zahtev za prikazom slika izabranog aranžmana (SO)

3. Sistem prikazuje administratoru listu slika izabranog aranžmana (IA)

Alternativni scenario:

2.1. Sistem ne može da obradi zahtev za prikaz slika izabranog aranžmana, prikazuje administratoru odgovarajuću poruku (IA)

SK27: Dodavanje novih slika za aranžman

Naziv: Dodavanje novih slika za aranžman

Aktori: Administrator

Učesnici: Administrator i sistem

Preduslovi: Sistem je aktivan, administrator je prijavljen i prikazana je lista slika izabranog aranžmana

Osnovni scenario:

1. Administrator poziva sistem da otpočne proces dodavanja nove slike za aranžman (APSO)
2. Sistem otvara file explorer prozor za dodavanje nove slike za aranžman (SO)
3. Sistem prikazuje file explorer prozor za dodavanje nove slike za aranžman (IA)
4. Administrator pretražuje file explorer (ANSO)
5. Administrator odabira fajl slike koju želi da doda (APSO)
6. Sistem čuva privremeno odabrani fajl slike (SO)
7. Sistem prikazuje naziv odabranog fajla slike (IA)
8. Administrator poziva sistem da doda fajl slike u fajl sistem aplikacije i doda novu sliku u galeriju slika aranžmana (APSO)
9. Sistem evidentira novu sliku aranžmana i ubacuje je u fajl sistem aplikacije (SO)
10. Sistem prikazuje poruku o uspešnosti dodavanja nove slike aranžmana (IA)

Alternativna scenarija:

2.1. Sistem ne može da otvori file explorer prozor za dodavanje nove slike, prikazuje administratoru odgovarajuću poruku i prekida izvršavanje scenarija (IA)

5.1. Administrator poziva sistem da zatvori file explorer prozor (APSO), sistem zatvara prozor (SO), prikazuje listu slika izabranog aranžmana i prekida izvršavanje scenarija (IA)

6.1. Sistem ne može da privremeno sačuva odabrani fajl slike, prikazuje administratoru odgovarajuću poruku i prekida izvršavanje scenarija (IA)

9.1. Sistem je utvrdio da fajl slike sa tim nazivom već postoji u fajl sistemu slika za taj aranžman, prikazuje administratoru odgovarajuću poruku (IA)

9.2. U fajl sistemu aplikacije nema dovoljno memorije za dodavanje nove slike, sistem prikazuje administratoru odgovarajuću poruku (IA)

9.3. Sistem ne može da evidentira novu sliku aranžmana i prikazuje administratoru odgovarajuću poruku (IA)

SK28: Brisanje slika aranžmana

Naziv: Brisanje slika aranžmana

Aktori: Administrator

Učesnici: Administrator i sistem

Preduslovi: Sistem je aktivan, administrator je prijavljen i prikazana je lista slika izabranog aranžmana

Osnovni scenario:

1. Administrator poziva sistem da započne proces brisanja izabrane slike aranžmana (APSO)
2. Sistem otvara dijalog za potvrdu brisanja izabrane slike (SO)
3. Sistem prikazuje dijalog za potvrdu brisanja izabrane slike (IA)
4. Administrator potvrđuje sistemu brisanje izabrane slike aranžmana (APSO)
5. Sistem briše sliku iz baze podataka i iz fajl sistema aplikacije (SO)
6. Sistem prikazuje poruku o uspešnosti brisanja izabrane slike aranžmana (IA)

Alternativna scenarija:

2.1. Sistem ne može da otvori dijalog za potvrdu brisanja slike, prikazuje administratoru odgovarajuću poruku i prekida izvršavanje scenarija (IA)

4.1. Administrator poziva sistem da poništi proces brisanja slike (APSO), sistem poništava proces (SO), prikazuje listu slika aranžmana i prekida izvršavanje scenarija (IA)

5.1. Sistem ne može da obriše sliku iz fajl sistema aplikacije, prikazuje administratoru odgovarajuću poruku (IA)

5.2. Sistem ne može da obriše sliku aranžmana iz baze podataka i prikazuje administratoru odgovarajuću poruku (IA)

SK29: Pregled svih proizvoda

Naziv: Pregled svih proizvoda

Aktori: Administrator

Učesnici: Administrator i sistem

Preduslovi: Sistem je aktivan, administrator je prijavljen

Osnovni scenario:

1. Administrator poziva sistem da prikaže listu svih proizvoda (APSO)
2. Sistem obrađuje zahtev za prikazom liste svih proizvoda (SO)
3. Sistem prikazuje listu svih proizvoda (IA)

Alternativni scenario:

2.1. Sistem ne može da obradi zahtev za prikazom svih proizvoda, prikazuje administratoru odgovarajuću poruku (IA)

SK30: Izmena podataka o proizvodu

Naziv: Izmena podataka o proizvodu

Aktori: Administrator

Učesnici: Administrator i sistem

Preduslovi: Sistem je aktivan, administrator je prijavljen i prikazana je forma za izmenu podataka o proizvodu

Osnovni scenario:

1. Administrator menja podatke izabranog proizvoda (APUSO)
2. Administrator poziva sistem da evidentira izmene podataka o proizvodu (APSO)
3. Sistem evidentira izmene podataka o proizvodu (SO)
4. Sistem prikazuje poruku o uspešnosti izmene podataka (IA)

Alternativna scenarija:

2.1. Administrator poziva sistem da poništi izmene podataka o proizvodu (APSO), sistem poništava izmene (SO), prikazuje listu svih proizvoda i prekida izvršavanje scenarija (IA)

3.1. Sistem je utvrdio da izmenjeni podaci nisu u validnom formatu, prikazuje administratoru odgovarajuću poruku (IA)

3.2. Sistem ne može da evidentira izmene podataka o proizvodu, prikazuje administratoru odgovarajuću poruku (IA)

SK31: Brisanje proizvoda

Naziv: Brisanje proizvoda

Aktori: Administrator

Učesnici: Administrator i sistem

Preduslovi: Sistem je aktivan, administrator je prijavljen i prikazana je lista svih proizvoda

Osnovni scenario:

1. Administrator poziva sistem da pokrene proces brisanja izabranog proizvoda (APSO)
2. Sistem otvara dijalog za potvrdu o brisanju proizvoda (SO)
3. Sistem prikazuje dijalog za potvrdu o brisanju proizvoda (IA)
4. Administrator potvrđuje brisanje proizvoda (APSO)
5. Sistem briše izabrani proizvod (SO)
6. Sistem prikazuje poruku o uspešnosti brisanja izabranog proizvoda (IA)

Alternativna scenarija:

2.1. Sistem ne može da otvori dijalog za potvrdu o brisanju proizvoda, prikazuje administratoru odgovarajuću poruku i prekida izvršavanje scenarija (IA)

4.1. Administrator poziva sistem da poništi proces brisanje proizvoda (APSO), sistem poništava proces (SO), prikazuje listu svih proizvoda i prekida izvršavanje scenarija (IA)

5.1. Sistem ne može da obriše proizvod, prikazuje administratoru odgovarajuću poruku (IA)

SK32: Pregled slika proizvoda

Naziv: Pregled slika proizvoda

Aktori: Administrator

Učesnici: Administrator i sistem

Preduslovi: Sistem je aktivan, administrator je prijavljen i prikazana je lista svih proizvoda

Osnovni scenario:

1. Administrator pregleda proizvoda i bira proizvod kojem želi da pregleda galeriju slika (ANSO)
2. Administrator poziva sistem da prikaže galeriju slika izabranog proizvoda (APSO)
3. Sistem obrađuje zahtev za prikazom galerije slika izabranog proizvoda (SO)
4. Sistem prikazuje galeriju slika izabranog proizvoda (IA)

Alternativni scenario:

3.1. Sistem ne može da obradi zahtev za prikazom galerije slika izabranog proizvoda, prikazuje administratoru odgovarajuću poruku (IA)

SK33: Dodavanje novih slika za proizvod

Naziv: Dodavanje novih slika za proizvod

Aktori: Administrator

Učesnici: Administrator i sistem

Preduslovi: Sistem je aktivan, administrator je prijavljen i prikazana je galerija slika izabranog proizvoda

Osnovni scenario:

1. Administrator poziva sistem da otpočne proces dodavanja nove slike za proizvod (APSO)
2. Sistem otvara file explorer prozor za dodavanje nove slike za proizvod (SO)
3. Sistem prikazuje file explorer prozor za dodavanje nove slike za proizvod (IA)
4. Administrator pretražuje file explorer (ANSO)
5. Administrator odabira fajl slike koju želi da doda (APSO)
6. Sistem čuva privremeno odabrani fajl slike (SO)
7. Sistem prikazuje naziv odabranog fajla slike (IA)
8. Administrator poziva sistem da doda fajl slike u fajl sistem aplikacije i doda novu sliku u galeriju slika proizvoda (APSO)
9. Sistem evidentira novu sliku proizvoda u bazi podataka i ubacuje je u fajl sistem aplikacije (SO)
10. Sistem prikazuje poruku o uspešnosti dodavanja nove slike proizvoda (IA)

Alternativna scenarija:

2.1. Sistem ne može da otvori file explorer prozor za dodavanje nove slike, prikazuje administratoru odgovarajuću poruku i prekida izvršavanje scenarija (IA)

5.1. Administrator odustaje od dodavanja nove slike (ANSO) i poziva sistem da zatvori file explorer (APSO), sistem zatvara file explorer (SO), prikazuje sve slike proizvoda i prekida izvršavanje scenarija (IA)

6.1. Sistem ne može da privremeno sačuva odabrani fajl slike, prikazuje administratoru odgovarajuću poruku i prekida izvršavanje scenarija (IA)

9.1. Sistem je utvrdio da fajl slike proizvoda pod tim nazivom već postoji u fajl sistemu aplikacije za taj proizvod, prikazuje administratoru odgovarajuću poruku (IA)

9.2. U fajl sistemu aplikacije nema dovoljno memorije za dodavanje nove slike, sistem prikazuje administratoru odgovarajuću poruku (IA)

9.3. Sistem ne može da evidentira novu sliku proizvoda u bazi podataka, prikazuje administratoru odgovarajuću poruku (IA)

SK34: Brisanje slika proizvoda

Naziv: Brisanje slika proizvoda

Aktori: Administrator

Učesnici: Administrator i sistem

Preduslovi: Sistem je aktivan, administrator je prijavljen i prikazana je galerija slika izabranog proizvoda

Osnovni scenario:

1. Administrator poziva sistem da započne proces brisanja izabrane slike proizvoda (APSO)
2. Sistem otvara dijalog za potvrdu brisanja izabrane slike (SO)
3. Sistem prikazuje dijalog za potvrdu brisanja izabrane slike (IA)
4. Administrator potvrđuje sistemu brisanje izabrane slike proizvoda (APSO)
5. Sistem briše sliku iz baze podataka i iz fajl sistema aplikacije (SO)
6. Sistem prikazuje poruku o uspešnosti brisanja izabrane slike proizvoda (IA)

Alternativna scenarija:

2.1. Sistem ne može da otvori dijalog za potvrdu brisanja slike, prikazuje administratoru odgovarajuću poruku i prekida izvršavanje scenarija (IA)

4.1. Administrator poziva sistem da poništi proces brisanja slike (APSO), sistem poništava proces (SO), prikazuje listu slika aranžmana i prekida izvršavanje scenarija (IA)

5.1. Sistem ne može da obriše sliku iz fajl sistema aplikacije, prikazuje administratoru odgovarajuću poruku (IA)

5.2. Sistem ne može da obriše sliku aranžmana iz baze podataka i prikazuje administratoru odgovarajuću poruku (IA)

SK35: Pregled svih polazaka za aranžman

Naziv: Pregled svih polazaka za aranžman

Aktori: Administrator

Učesnici: Administrator i sistem

Preduslovi: Sistem je aktivan, administrator je prijavljen i prikazana je lista svih aranžmana

Osnovni scenario:

1. Administrator poziva sistem da prikaže sve polaske za izabrani aranžman (APSO)

2. Sistem obrađuje zahtev za prikazom svih polazaka za izabrani aranžman (SO)

3. Sistem prikazuje listu svih polazaka za izabrani aranžman (IA)

Alternativni scenario:

2.1. Sistem ne može da obradi zahtev za prikazom svih polazaka za izabrani aranžman, prikazuje administratoru odgovarajuću poruku (IA)

SK36: Dodavanje novog polaska za aranžman

Naziv: Dodavanje novog polaska za aranžman

Aktori: Administrator

Učesnici: Administrator i sistem

Preduslovi: Sistem je aktivan, administrator je prijavljen i prikazana je lista svih polazaka za aranžman

Osnovni scenario:

1. Administrator poziva sistem da prikaže formu za dodavanje novog polaska za aranžman (APSO)

2. Sistem otvara formu za dodavanje novog polaska za aranžman (SO)

3. Sistem prikazuje formu za dodavanje novog polaska za aranžman (IA)

4. Administrator upisuje podatke novog polaska (APUSO)

5. Administrator poziva sistem da evidentira novi polazak (APSO)

6. Sistem evidentira novi polazak aranžmana (SO)

7. Sistem prikazuje poruku o uspešnosti dodavanja novog polaska aranžmana (IA)

Alternativna scenarija:

2.1. Sistem ne može da otvori formu za dodavanje novog polaska za aranžman, prikazuje administratoru odgovarajuću poruku i prekida izvršavanje scenarija (IA)

5.1. Administrator poziva sistem da poništi proces dodavanja novog polaska aranžmana (APSO), sistem poništava proces (SO), prikazuje listu svih polazaka aranžmana i prekida izvršavanje scenarija (IA)

6.1. Sistem je utvrdio da je administrator uneo jedan ili više podataka u formatu koji nije validan, prikazuje administratoru odgovarajuću poruku (IA) i scenario se nastavlja od koraka 4

6.2. Sistem ne može da evidentira novi polazak aranžmana, prikazuje administratoru odgovarajuću poruku (IA)

SK37: Izmena podataka o polasku aranžmana

Naziv: Izmena podataka o polasku aranžmana

Aktori: Administrator

Učesnici: Administrator i sistem

Preduslovi: Sistem je aktivan, administrator je prijavljen i prikazana je forma za izmenu podataka o polasku aranžmana

Osnovni scenario:

1. Administrator menja podatke o polasku aranžmana (APUSO)
2. Administrator poziva sistem da evidentira izmenjene podatke o polasku aranžmana (APSO)
3. Sistem evidentira izmenjene podatke o polasku aranžmana (SO)
4. Sistem prikazuje poruku o uspešnosti izmene podataka o polasku aranžmana (IA)

Alternativna scenarija:

2.1. Administrator poziva sistem da poništi proces izmene podataka polaska aranžmana (APSO), sistem poništava proces (SO), prikazuje listu svih polazaka aranžmana i prekida izvršavanje scenarija (IA)

3.1. Sistem je utvrdio da je administrator uneo jedan ili više izmenjenih podataka u formatu koji nije validan, prikazuje administratoru odgovarajuću poruku i prekida izvršavanje scenarija (IA)

3.2. Sistem ne može da evidentira izmene za polazak aranžmana, prikazuje administratoru odgovarajuću poruku (IA)

SK38: Brisanje polaska aranžmana

Naziv: Brisanje polaska aranžmana

Aktori: Administrator

Učesnici: Administrator i sistem

Preduslovi: Sistem je aktivan, administrator je prijavljen i prikazana je lista svih polazaka za izabrani aranžman

Osnovni scenario:

1. Administrator poziva sistem da pokrene proces brisanja polaska aranžmana (APSO)
2. Sistem otvara dijalog za potvrdu o brisanju izabranog polaska (SO)
3. Sistem prikazuje dijalog za potvrdu o brisanju izabranog polaska (IA)
4. Administrator potvrđuje sistemu brisanje izabranog polaska (APSO)
5. Sistem briše izabrani polazak (SO)
6. Sistem prikazuje poruku o uspešnosti brisanja izabranog polaska aranžmana (IA)

Alternativna scenarija:

2.1. Sistem ne može da otvori dijalog za potvrdu o brisanju izabranog polaska, prikazuje administratoru odgovarajuću poruku i prekida izvršavanje scenarija (IA)

4.1. Administrator poziva sistem da poništi proces brisanja izabranog polaska (APSO), sistem poništava proces (SO), prikazuje listu svih polazaka za aranžman i prekida izvršavanje scenarija (IA)

5.1. Sistem ne može da obriše izabrani polazak, prikazuje odgovarajuću poruku (IA)

SK39: Pregled rezervacija za polazak aranžmana

Naziv: Pregled rezervacija za polazak aranžmana

Aktori: Administrator

Učesnici: Administrator i sistem

Preduslovi: Sistem je aktivan, administrator je prijavljen i prikazana je lista svih polazaka za izabrani aranžman

Osnovni scenario:

1. Administrator poziva sistem da prikaže rezervacije polaska za izabrani polazak aranžmana (APSO)

2. Sistem obrađuje zahtev za prikazom rezervacija (SO)

3. Sistem prikazuje listu svih rezervacija za odabrani polazak aranžmana (IA)

Alternativni scenario:

2.1. Sistem ne može da obradi zahtev za prikazom rezervacija, prikazuje administratoru odgovarajuću poruku (IA)

SK40: Dodavanje nove rezervacije za polazak aranžmana

Naziv: Dodavanje nove rezervacije za polazak aranžmana

Aktori: Administrator

Učesnici: Administrator i sistem

Preduslovi: Sistem je aktivan, administrator je prijavljen i prikazana je forma za dodavanje nove rezervacije u okviru izabranog polaska aranžmana

Osnovni scenario:

1. Administrator upisuje email korisnika za kojeg dodaje novu rezervaciju (APUSO)

2. Administrator poziva sistem da proveri email korisnika (APSO)

3. Sistem proverava email korisnika (SO)

4. Sistem prikazuje izmenjenu formu dodavanja nove rezervacije (IA)

5. Administrator poziva sistem da evidentira novu rezervaciju za korisnika (APSO)

6. Sistem evidentira novu rezervaciju polaska aranžmana (SO)

7. Sistem prikazuje poruku o uspešnosti dodavanja nove rezervacije polaska (IA)

Alternativna scenarija:

2.1. Administrator poziva sistem da poništi proces dodavanja nove rezervacije (APSO), sistem poništava proces (SO), prikazuje listu svih rezervacija za izabrani polazak i prekida izvršavanje scenarija (IA)

3.1. Sistem je utvrdio da je administrator upisao email korisnika u formatu koji nije validan, prikazuje odgovarajuću poruku administratoru i prekida izvršavanje scenarija (IA)

3.2. Sistem je utvrdio da već postoji rezervacija u okviru izabranog polaska za korisnika sa unetim email-om, prikazuje odgovarajuću poruku i prekida izvršavanje scenarija (IA)

3.3. Sistem je utvrdio da korisnik sa unetim email-om ne postoji u bazi podataka, prikazuje administratoru odgovarajuću poruku i prekida izvršavanje scenarija (IA)

3.4. Sistem je utvrdio da je izabrani polazak aranžmana dostigao maksimalan broj rezervacija, prikazuje administratoru odgovarajuću poruku i prekida izvršavanje scenarija (IA)

5.1. Administrator poziva sistem da poništi proces dodavanja nove rezervacije (APSO), sistem poništava proces (SO), prikazuje listu svih rezervacija za izabrani polazak i prekida izvršavanje scenarija (IA)

6.1. Sistem ne može da evidentira novu rezervaciju polaska aranžmana, prikazuje administratoru odgovarajuću poruku (IA)

SK41: Brisanje rezervacije za polazak aranžmana

Naziv: Brisanje rezervacije za polazak aranžmana

Aktori: Administrator

Učesnici: Administrator i sistem

Preduslovi: Sistem je aktivan, administrator je prijavljen i prikazana je lista svih rezervacija za izabrani polazak aranžmana

Osnovni scenario:

1. Administrator poziva sistem da pokrene proces brisanja izabrane rezervacije polaska (APSO)

2. Sistem otvara dijalog za potvrdu o brisanju izabrane rezervacije polaska (SO)

3. Sistem prikazuje dijalog za potvrdu o brisanju izabrane rezervacije polaska (IA)

4. Administrator potvrđuje sistemu brisanje izabrane rezervacije (APSO)

5. Sistem briše izabranu rezervaciju polaska (SO)

6. Sistem prikazuje poruku o uspešnosti brisanja izabrane rezervacije polaska aranžmana (IA)

Alternativna scenarija:

2.1. Sistem ne može da otvori dijalog za potvrdu o brisanju izabrane rezervacije polaska, prikazuje administratoru odgovarajuću poruku i prekida izvršavanje scenarija (IA)

4.1. Administrator poziva sistem da poništi proces brisanja izabrane rezervacije (APSO), sistem poništava proces (SO), prikazuje listu svih rezervacija za polazak i prekida izvršavanje scenarija (IA)

5.1. Sistem ne može da obriše izabranu rezervaciju, prikazuje odgovarajuću poruku (IA)

SK42: Pretraga korisnika

Naziv: Pretraga korisnika

Aktori: Administrator

Učesnici: Administrator i sistem

Preduslovi: Sistem je aktivan, administrator je prijavljen i prikazana je lista svih korisnika

Osnovni scenario:

1. Administrator upisuje neku od ključnih reči za pretragu korisnika (ime i prezime ili email) (APUSO)

2. Administrator poziva sistem da pretraži korisnike po ključnoj reči (APSO)

3. Sistem pretražuje korisnike (SO)

4. Sistem prikazuje jednog ili više korisnika vezanih za upisane ključne reči pretrage (IA)

Alternativna scenarija:

3.1. Sistem nije uspeo da pronađe nijednog korisnika sa zadatom ključnom reči, prikazuje administratoru odgovarajuću poruku (IA)

3.2. Sistem ne može da pretraži korisnike, prikazuje administratoru odgovarajuću poruku (IA)

SK43: Pretraga proizvoda

Naziv: Pretraga proizvoda

Aktori: Administrator

Učesnici: Administrator i sistem

Preduslovi: Sistem je aktivan, administrator je prijavljen i prikazana je lista svih proizvoda

Osnovni scenario:

1. Administrator upisuje neku od ključnih reči za pretragu proizvoda (naziv proizvoda) (APUSO)

2. Administrator poziva sistem da pretraži proizvode po ključnoj reči (APSO)

3. Sistem pretražuje proizvode po ključnoj reči (SO)

4. Sistem prikazuje jedan ili više proizvoda vezanih za upisane ključne reči pretrage (IA)

Alternativna scenarija:

3.1. Sistem nije uspeo da pronađe nijedan proizvod za zadatu ključnu reč, prikazuje administratoru odgovarajuću poruku (IA)

3.2. Sistem ne može da pretraži proizvode, prikazuje administratoru odgovarajuću poruku (IA)

2.1.9. Opis slučajeva korišćenja za menadžera

Nakon predstavljenih dijagrama slučajeva korišćenja za svakog korisnika sistema (aktora), ovde će biti dat detaljan opis svakog pojedinačnog slučaja korišćenja (SK) koji se sastoji od naziva, aktora, učesnika, preduslova, osnovnog i alternativnog/ih scenarija. U ovom poglavlju predstavljeni su opisi slučajeva korišćenja za menadžera.

SK44: Pregled izveštaja o prodaji svih proizvoda

Naziv: Pregled izveštaja o prodaji svih proizvoda

Aktori: Menadžer

Učesnici: Menadžer i sistem

Preduslovi: Sistem je aktivan, menadžer je prijavljen

Osnovni scenario:

1. Menadžer poziva sistem da prikaže tabelu izveštaja o prodaji svih proizvoda (APSO)

2. Sistem obrađuje zahtev za prikazom tabele izveštaja o prodaji svih proizvoda (SO)

3. Sistem prikazuje tabelu izveštaja o prodaji proizvoda (IA)

Alternativni scenario:

2.1. Sistem ne može da obradi zahtev za prikazom tabele izveštaja, prikazuje menadžeru odgovarajuću poruku (IA)

SK45: Pregled izveštaja budućih putovanja

Naziv: Pregled izveštaja budućih putovanja

Aktori: Menadžer

Učesnici: Menadžer i sistem

Preduslovi: Sistem je aktivan, menadžer je prijavljen

Osnovni scenario:

1. Menadžer poziva sistem da prikaže tabelu izveštaja o predstojećim putovanjima (polascima aranžmana) (APSO)

2. Sistem obrađuje zahtev za prikazom tabele izveštaja o prestojećim putovanjima (SO)

3. Sistem prikazuje tabelu izveštaja predstojećih putovanja (IA)

Alternativni scenario:

2.1. Sistem ne može da obradi zahtev za prikazom tabele izveštaja o predstojećim putovanjima, prikazuje menadžeru odgovarajuću poruku (IA)

SK46: Pregled izveštaja prošlih putovanja

Naziv: Pregled izveštaja prošlih putovanja

Aktori: Menadžer

Učesnici: Menadžer i sistem

Preduslovi: Sistem je aktivan, menadžer je prijavljen

Osnovni scenario:

1. Menadžer poziva sistem da prikaže tabelu izveštaja prošlih putovanja (polazaka aranžmana) (APSO)

2. Sistem obrađuje zahtev za prikazom tabele izveštaja prošlih putovanja (SO)

3. Sistem prikazuje tabelu izveštaja prošlih putovanja (IA)

Alternativni scenario:

2.1. Sistem ne može da obradi zahtev za prikazom tabele izveštaja prošlih putovanja, prikazuje menadžeru odgovarajuću poruku (IA)

SK47: Pregled izveštaja svih aranžmana

Naziv: Pregled izveštaja svih aranžmana

Aktori: Menadžer

Učesnici: Menadžer i sistem

Preduslovi: Sistem je aktivan, menadžer je prijavljen

Osnovni scenario:

1. Menadžer poziva sistem da prikaže tabelu izveštaja svih aranžmana (APSO)
2. Sistem obrađuje zahtev za prikazom tabele izveštaja svih aranžmana (SO)
3. Sistem prikazuje tabelu izveštaja svih aranžmana (IA)

Alternativni scenario:

2.1. Sistem ne može da obradi zahtev za prikazom tabele izveštaja svih aranžmana, prikazuje menadžeru odgovarajuću poruku (IA)

SK48: Sortiranje tabele izveštaja prema odabranim parametrima

Naziv: Sortiranje tabele izveštaja prema odabranim parametrima

Aktori: Menadžer

Učesnici: Menadžer i sistem

Preduslovi: Sistem je aktivan, menadžer je prijavljen i prikazana je neka od tabele izveštaja poslovanja planinarskog kluba

Osnovni scenario:

1. Menadžer pregledava i bira neki od parametara (kolona tabele), prema kojem želi da sortira tabelu izveštaja (ANSO)
2. Menadžer poziva sistem da sortira tabelu izveštaja prema odabranom parametru (APSO)
3. Sistem sortira tabelu izveštaja prema odabranom parametru (SO)
4. Sistem prikazuje sortiranu tabelu izveštaja prema odabranom parametru (IA)

Alternativni scenario:

3.1. Sistem ne može da sortira tabelu izveštaja prema odabranom parametru, prikazuje menadžeru odgovarajuću poruku (IA)

2.2. FAZA ANALIZE

Faza analize Larmanove metode razvoja softvera je druga faza razvoja, koja treba da da opis logičke strukture aplikativnog softvera i ponašanje sistema.

Ponašanje sistema (aplikacije) predstavićemo kroz sistemske dijagrame sekvence (DSSK), koji se prave za svaki slučaj korišćenja (SK) opisan u prethodnoj fazi, kao i kroz ugovore o sistemskim operacijama, koje ćemo dobiti na osnovu sistemskih dijagrama sekvenci. Strukturu aplikativnog softvera predstavićemo pomoću konceptualnog i relacionog modela [1].

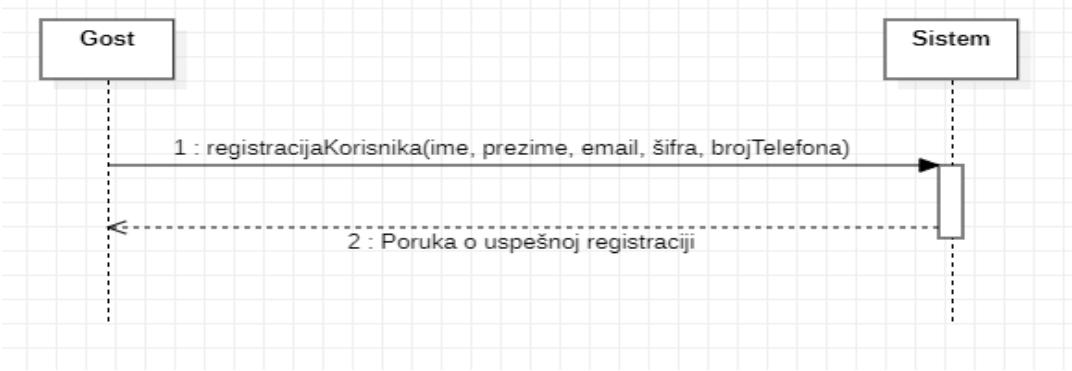
2.2.1. Sistemski dijagrami sekvenci za slučajeve korišćenja

Zbog ograničenog obima završnog rada, u ovom poglavlju će biti prikazano 8 karakterističnih slučajeva korišćenja. Dijagrami sekvenci će biti prikazani za sva osnovna scenarija i 1 alternativni scenario.

DSSK1: Registracija

Osnovni scenario:

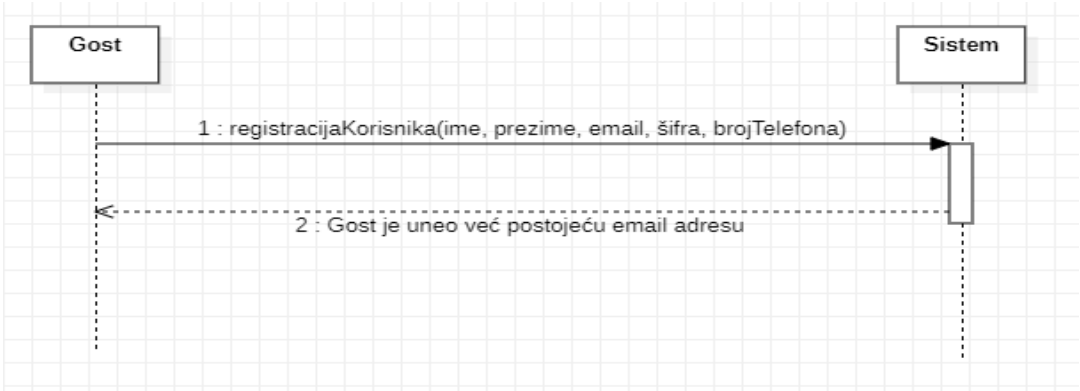
1. Gost poziva sistem da obradi podatke za registraciju (APSO)
2. Sistem prikazuje gostu poruku o uspešnoj registraciji (IA)



Slika 11. DSSK1 – Osnovni scenario

Alternativna scenarija:

2.1. Gost je uneo već postojeću email adresu, sistem prikazuje odgovarajuću poruku gostu (IA)



Slika 12. DSSK1 – Alternativni scenario 2.1.

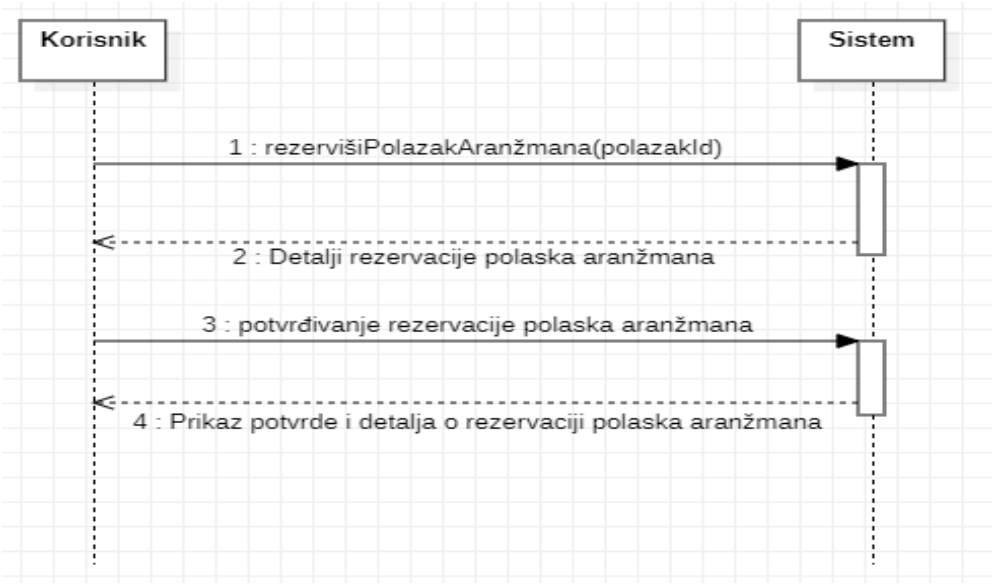
Uvedene su sistemske operacije:

1. registracijaKorisnika(ime, prezime, email, šifra, brojTelefona)

DSSK9: Rezervacija polaska aranžmana

Osnovni scenario:

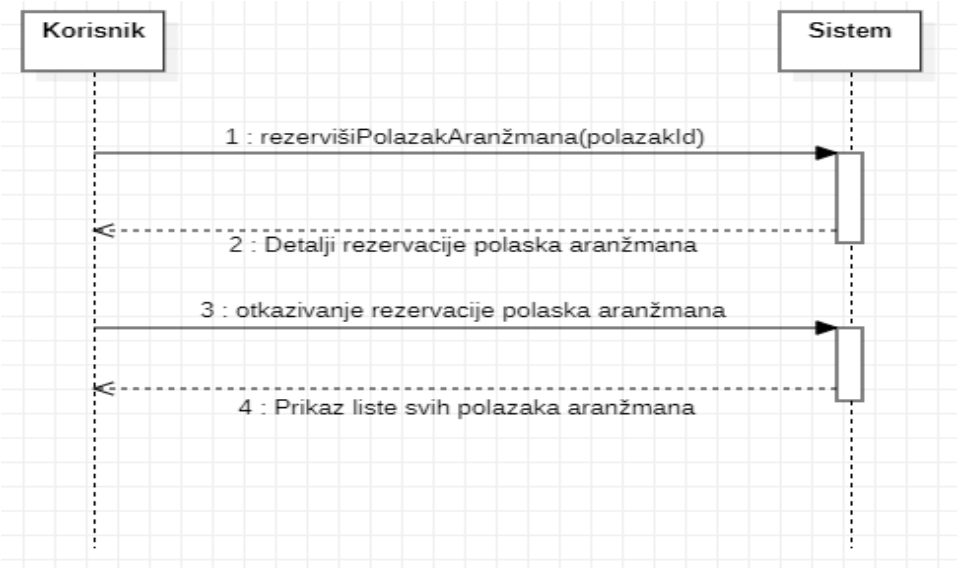
1. Korisnik poziva sistem da rezerviše odabrani polazak aranžmana (APSO)
2. Sistem prikazuje detalje rezervacije odabranog polaska aranžmana (IA)
3. Korisnik potvrđuje sistemu rezervaciju polaska (APSO)
4. Sistem prikazuje korisniku potvrdu rezervacije polaska aranžmana (IA)



Slika 13. DSSK9 – Osnovni scenario

Alternativna scenarija:

3.1. Korisnik otkazuje proces rezervacije (APSO), prikazuje korisniku listu svih polazaka aranžmana i prekida izvršavanje scenarija (IA)



Slika 14. DSSK9 – Alternativni scenario 3.1.

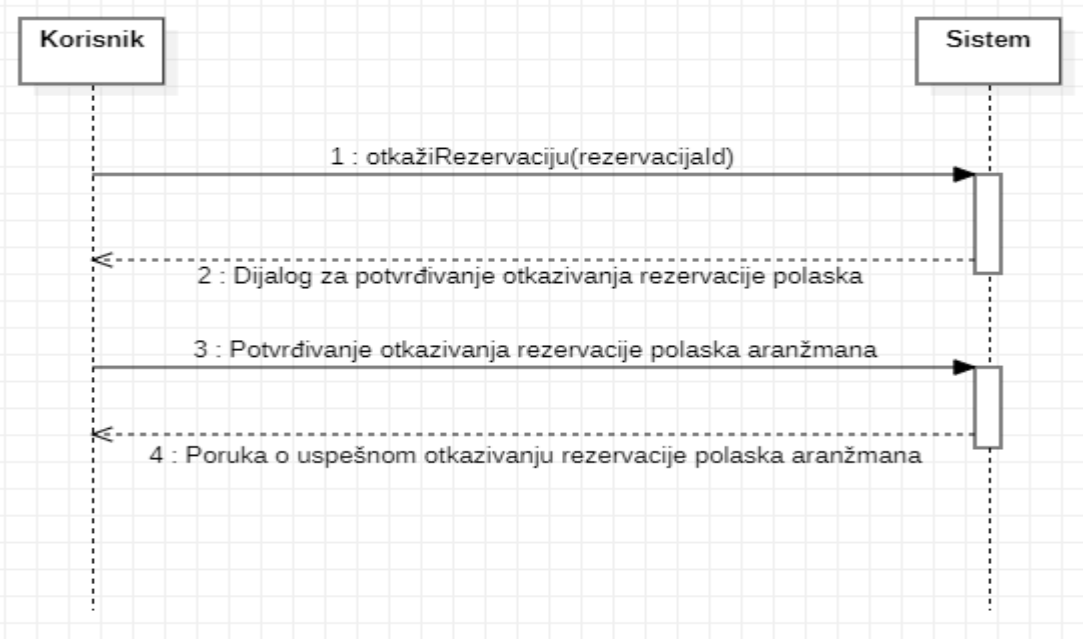
Uvedene su sistemske operacije:

1. rezervišiPolazakAranžmana(polazakId)

DSSK11: Otkazivanje rezervacije

Osnovni scenario:

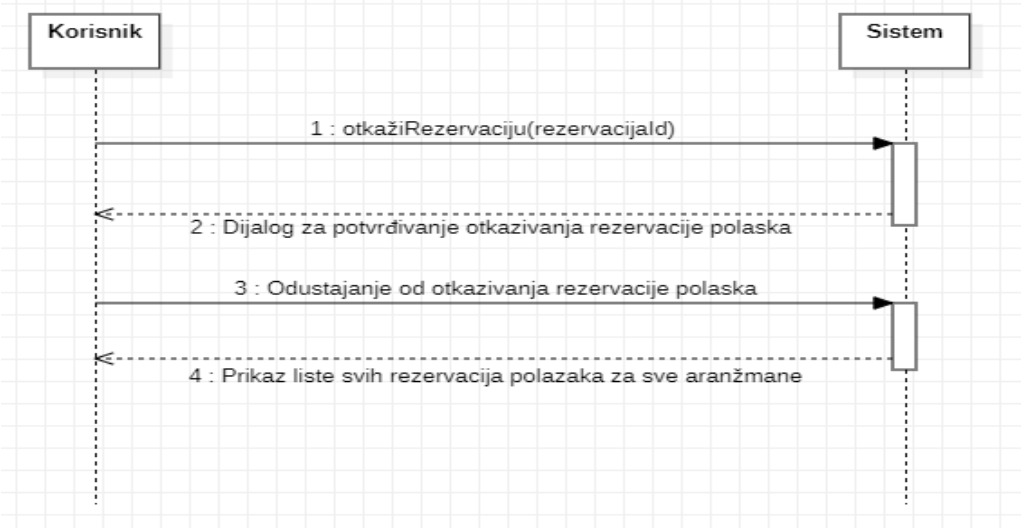
1. Korisnik poziva sistem da otkaže odabranu rezervaciju (APSO)
2. Sistem traži od korisnika da potvrdi otkazivanja odabrane rezervacije (IA)
3. Korisnik potvrđuje otkazivanje odabrane rezervacije (APSO)
4. Sistem prikazuje korisniku potvrdu o otkazivanju rezervacije (IA)



Slika 15. DSSK11 – Osnovni scenario

Alternativna scenarija:

3.1. Korisnik odustaje od otkazivanja rezervacije (APSO), prikazuje korisniku listu svih rezervacija polazaka i prekida izvršavanje scenarija (IA)



Slika 16. DSSK11 – Alternativni scenario 3.1.

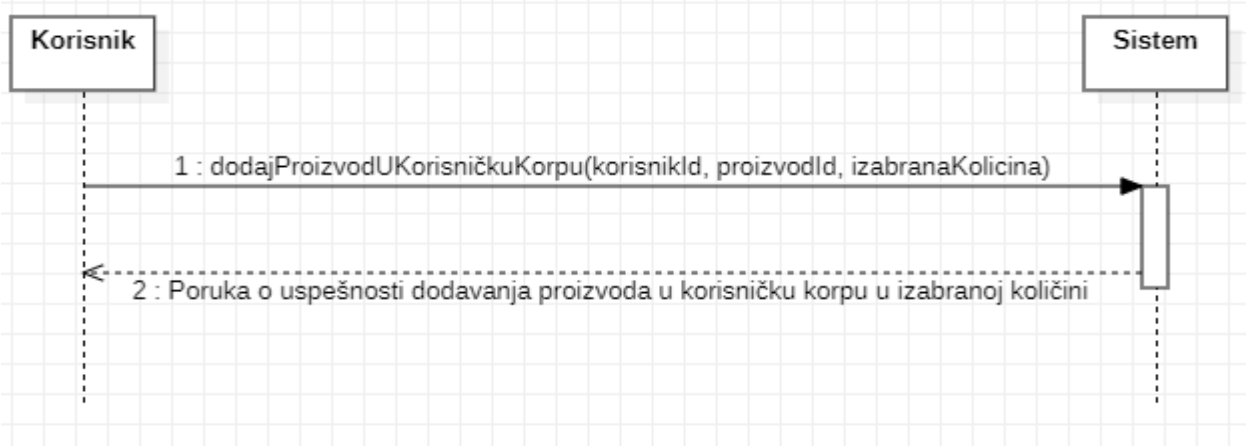
Uvedene su sistemske operacije:

1. otkažiRezervaciju(rezervacijaId)

DSSK13: Dodavanje proizvoda u korisničku korpu

Osnovni scenario:

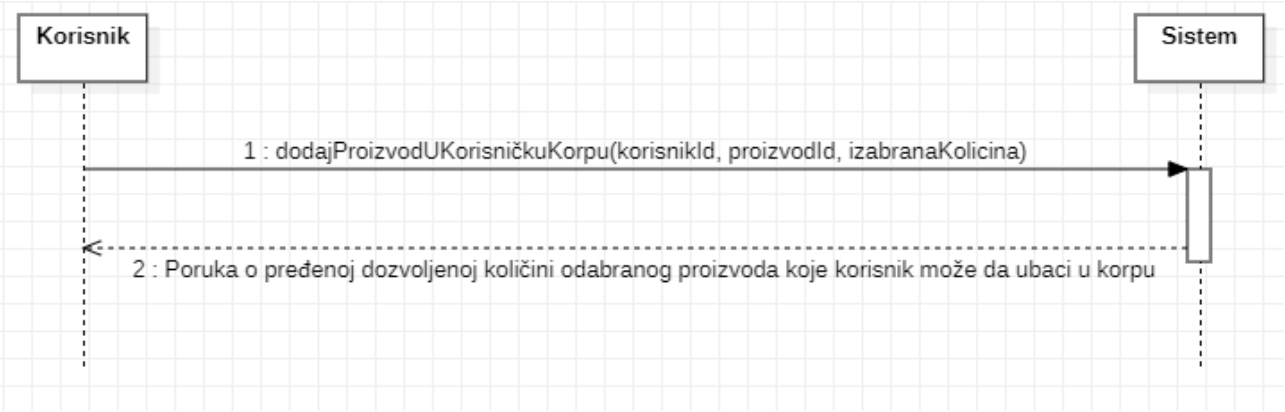
1. Korisnik poziva sistem da ubaci odabrani proizvod u korpu u biranoj količini (APSO)
2. Sistem prikazuje korisniku poruku o uspešnosti dodavanje proizvoda u korisničku korpu (IA)



Slika 17. DSSK13 – Osnovni scenario

Alternativna scenarija:

2.2. Sistem utvrđuje da je korisnik već dodao odabrani proizvod u korisničku korpu i da sa odabranom količinom prelazi limit količine proizvoda koju može da stavi u korpu, prikazuje odgovarajuću poruku korisniku (IA)



Slika 18. DSSK13 – Alternativni scenario 2.2.

Uvedene su sistemske operacije:

1. dodajProizvodUKorisničkuKorpu(korisnikId, proizvodId, izabranaKolicina)

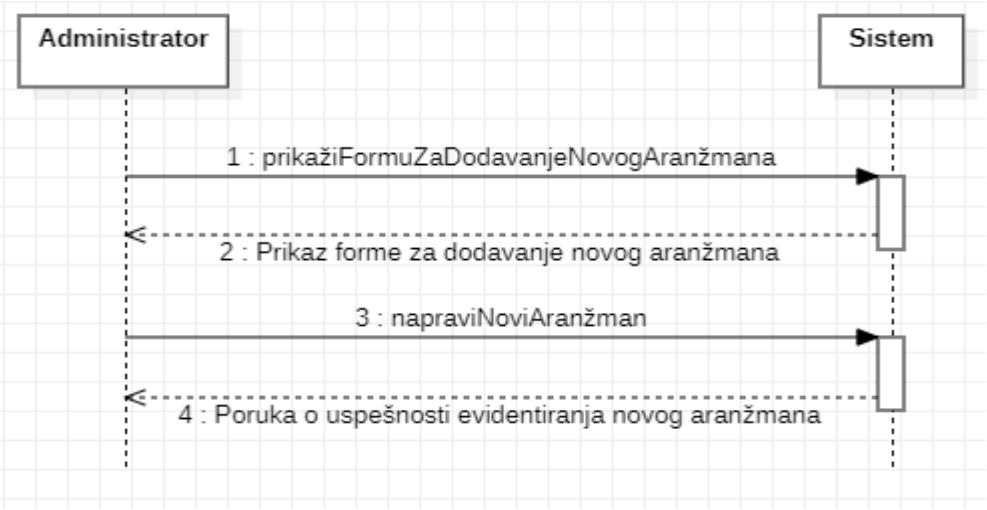
DSSK23: Dodavanje novog aranžmana

Osnovni scenario:

1. Administrator poziva sistem da mu prikaže formu za dodavanje novog aranžmana (APSO)
2. Sistem prikazuje formu za dodavanje novog aranžmana (IA)

3. Administrator poziva sistem da evidentira novi aranžman (APSO)

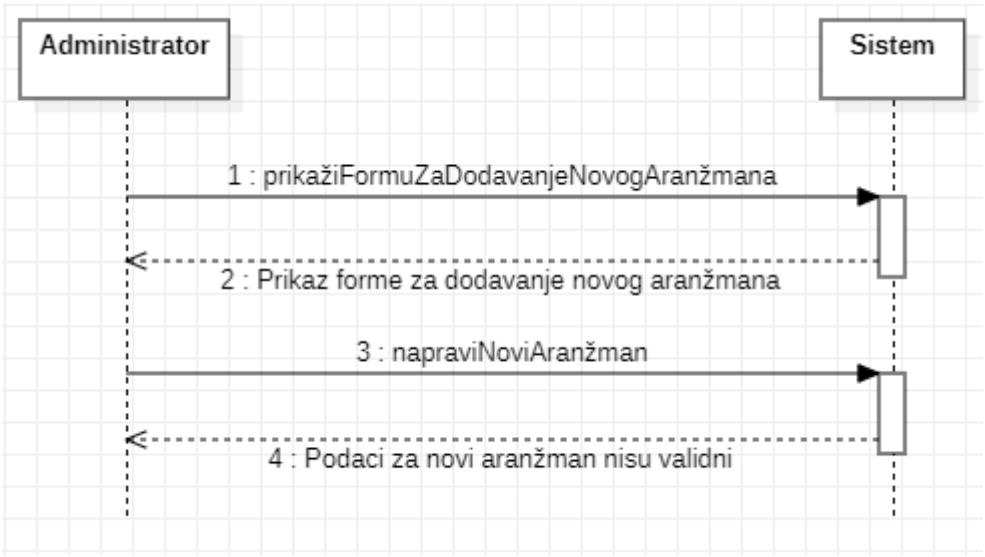
4. Sistem prikazuje poruku o uspešnom dodatom novom aranžmanu (IA)



Slika 19. DSSK23 – Osnovni scenario

Alternativna scenarija:

4.1. Sistem je utvrdio da administrator nije uneo jedan ili više podataka u validnom formatu, prikazuje administratoru odgovarajuću poruku (IA)



Slika 20. DSSK23 – Alternativni scenario 4.1.

Uvedene su sistemske operacije:

1. prikaziFormuZaDodavanjeNovogAranžmana
2. napraviNoviAranžman

DSSK27: Dodavanje novih slika za aranžman

Osnovni scenario:

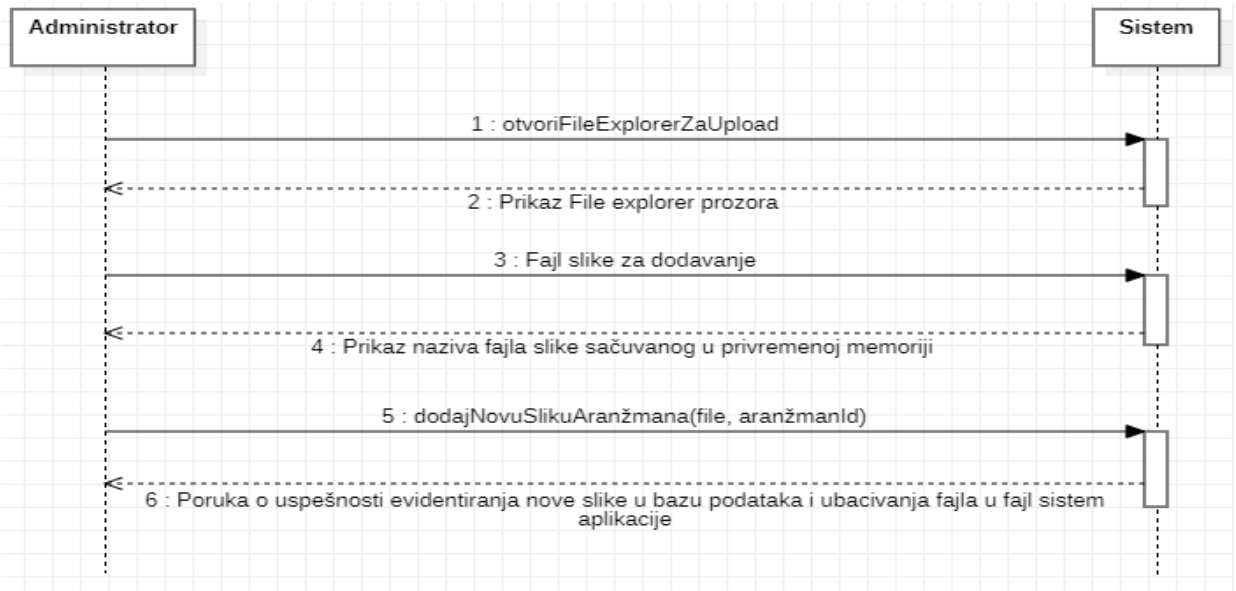
1. Administrator poziva sistem da otpočne proces dodavanja nove slike za aranžman (APSO)
2. Sistem prikazuje file explorer prozor za dodavanje nove slike za aranžman (IA)

3. Administrator odabira fajl slike koju želi da doda (APSO)

4. Sistem prikazuje naziv odabranog fajla slike (IA)

5. Administrator poziva sistem da doda fajl slike u fajl sistem aplikacije i doda novu sliku u galeriju slika aranžmana (APSO)

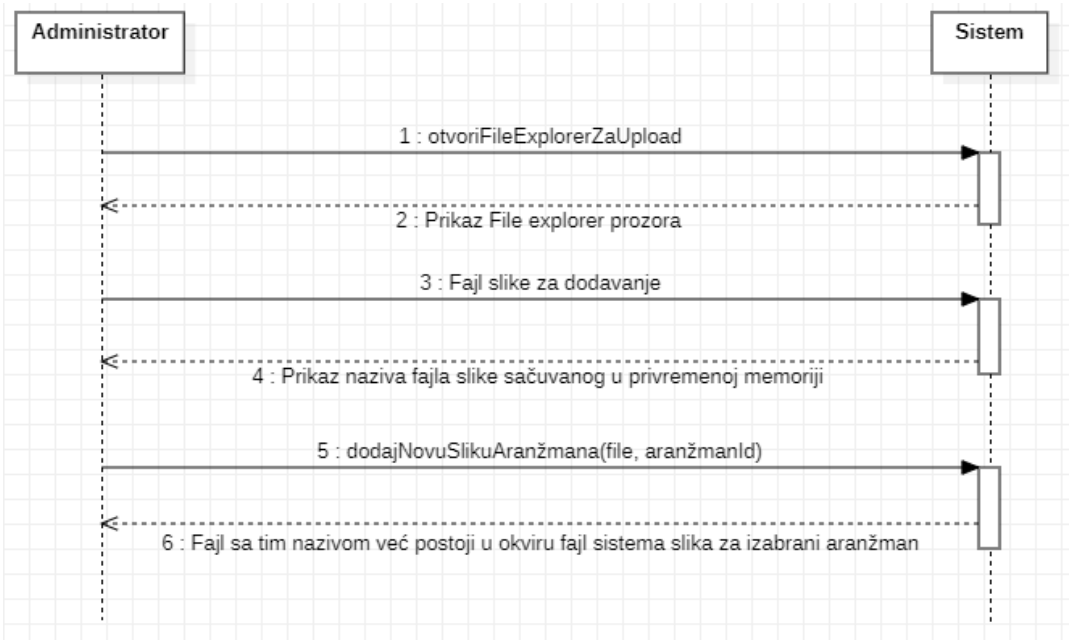
6. Sistem prikazuje poruku o uspešnosti dodavanja nove slike aranžmana (IA)



Slika 21. DSSK27 – Osnovni scenario

Alternativna scenarija:

6.1. Sistem je utvrdio da fajl slike sa tim nazivom već postoji u fajl sistemu slika za taj aranžman, prikazuje administratoru odgovarajuću poruku (IA)



Slika 22. DSSK27 – Alternativni scenario 6.1.

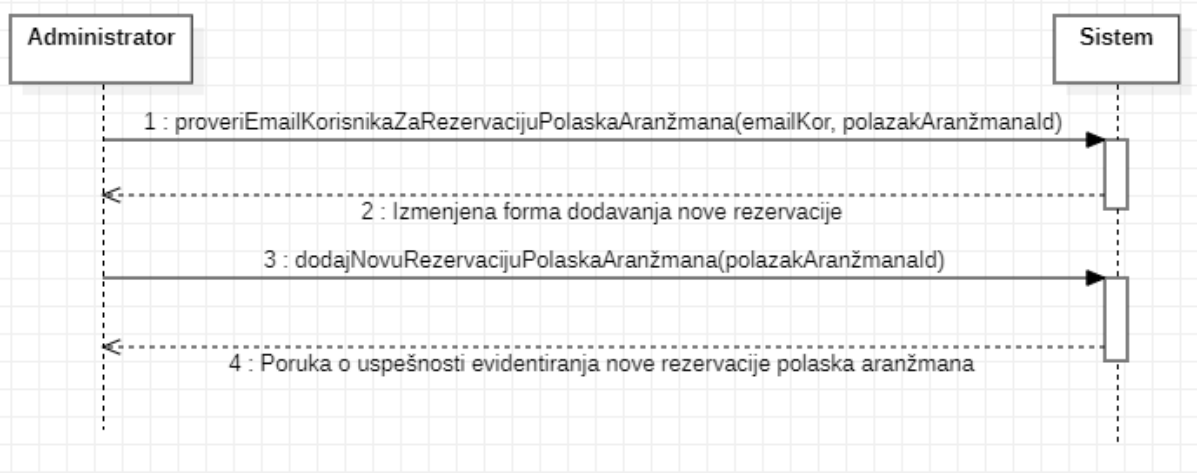
Uvedene su sistemske operacije:

1. otvoriFileExplorerZaUpload

2. dodajNovuSlikuAranžmana(file, aranžmanId)

DSSK40: Dodavanje nove rezervacije za polazak aranžmana

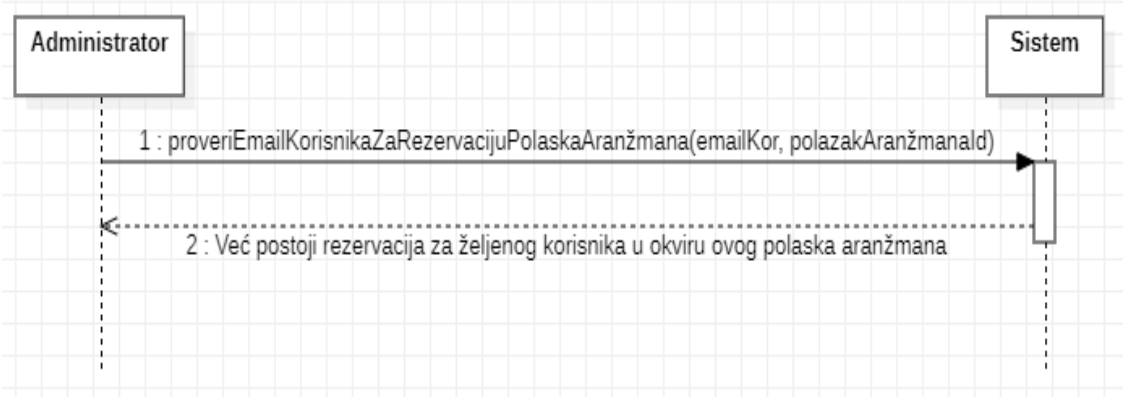
1. Administrator poziva sistem da proveri email korisnika (APSO)
2. Sistem prikazuje izmenjenu formu dodavanja nove rezervacije (IA)
3. Administrator poziva sistem da evidentira novu rezervaciju za korisnika (APSO)
4. Sistem prikazuje poruku o uspešnosti dodavanja nove rezervacije polaska (IA)



Slika 23. DSSK40 – Osnovni scenario

Alternativni scenario:

2.2. Sistem je utvrdio da već postoji rezervacija u okviru izabranog polaska za korisnika sa unetim email-om, prikazuje odgovarajuću poruku i prekida izvršavanje scenarija (IA)



Slika 24. DSSK40 – Alternativni scenario 2.2.

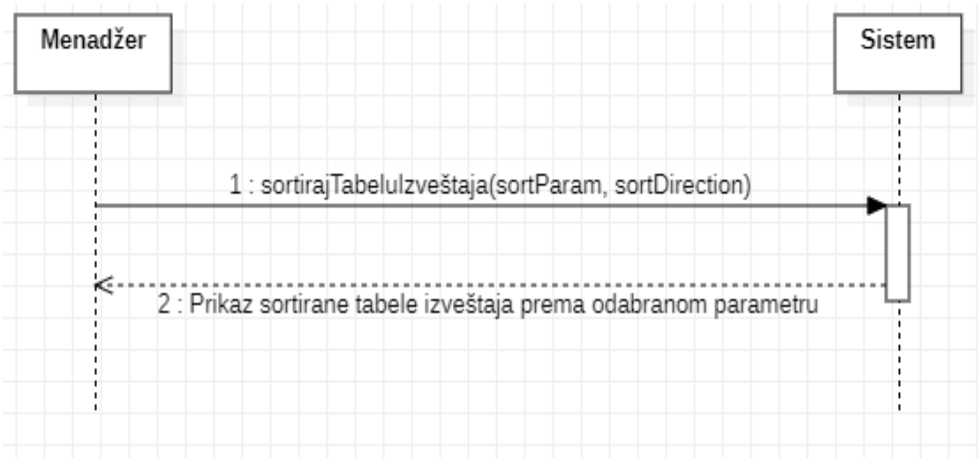
Uvedene su sistemske operacije:

1. proveriEmailKorisnikaZaRezervacijuPolaskaAranžmana(emailKor, polazakAranžmanaId)
2. dodajNovuRezervacijuPolaskaAranžmana(polazakAranžmanaId)

DSSK48: Sortiranje tabele izveštaja prema odabranim parametrima

Osnovni scenario:

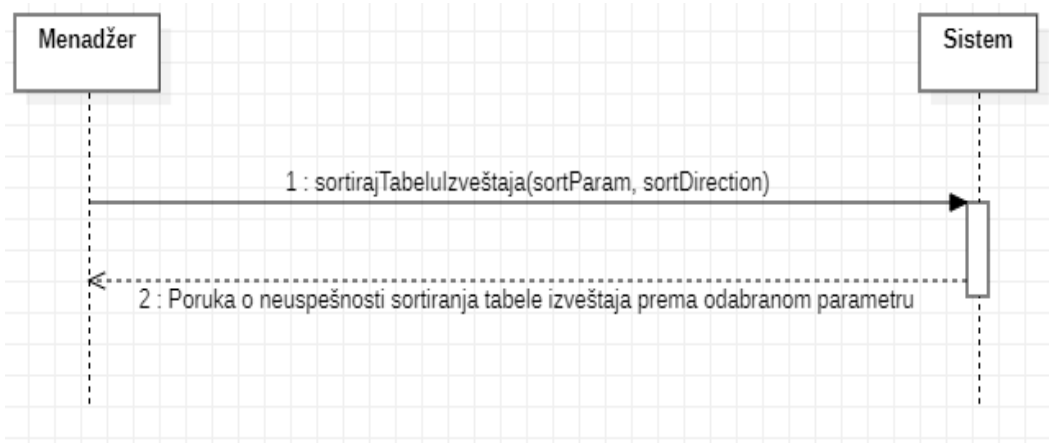
1. Menadžer poziva sistem da sortira tabelu izveštaja prema odabranom parametru (APSO)
2. Sistem prikazuje sortiranu tabelu izveštaja prema odabranom parametru (IA)



Slika 25. DSSK48 – Osnovni scenario

Alternativni scenario:

2.1. Sistem ne može da sortira tabelu izveštaja prema odabranom parametru, prikazuje menadžeru odgovarajuću poruku (IA)



Slika 26. DSSK48 – Alternativni scenario 2.1.

Uvedene su sistemske operacije:

1. sortirajTabeluIzveštaja(sortParam, sortDirection)

2.2.2. Definisanje ugovora o sistemskim operacijama

Tokom izrade sistemskih dijagrama sekvence slučajeva korišćenja (DSSK) uočene su određene sistemske operacije. Za svaku od njih prave se ugovori (eng. contracts). Ugovori opisuju šta sistemska operacija radi, ali ne i kako. Jedan ugovor odgovara jednoj sistemskoj operaciji [1].

Zbog ograničenog obima završnog rada, biće prikazano 20 ugovora za karakteristične sistemske operacije.

Uočene sistemske operacije koje treba projektovati su:

1. registracijaKorisnika(ime, prezime, email, šifra, brojTelefona)
2. prikažiAranžman(aranžmanId)
3. prikažiListuPolazakaAranžmana(aranžmanId)
4. prikažiProdavnicu
5. sortirajProizvodePoCeni(sortDirection)
6. prikažiDetaljeProizvoda(proizvodId)
7. prijavljivanje(email, šifra)
8. odjavljivanje
9. rezervišiPolazakAranžmana(polazakId)
10. prikažiRezervacijePolazaka(korisnikId)
11. otkažiRezervaciju(rezervacijaId)
12. prikažiKorisničkuKorpu(korisnikId)
13. dodajProizvodUKorisničkuKorpu(korisnikId, proizvodId, izabranaKolicina)
14. izbaciStavkuKorpe(stavkaKorpeId)
15. izračunajNovuCenuPorudžbine(clanKartPopustId)
16. prikažiFormuPodatakaODostavi(listaStavkiKorpe, korisnikId)
17. napraviNovuNarudžbenicu(listaStavkiKorpe, korisnikId)
18. prikažiLičnePodatke(korisnikId)
19. izmeniLičnePodatkeNaloga(korisnikId)
20. prikažiNarudžbenice(korisnikId)
21. prikažiListuSvihKorisnika
22. izmeniPodatkeČlanskeKarticeKorisnika(članKartId)
23. obrišiKorisnika(korisnikId)
24. prikažiListuSvihAranžmana
25. prikažiFormuZaDodavanjeNovogAranžmana
26. napraviNoviAranžman
27. sačuvajIzmenjenePodatkeAranžmana(aranžmanId)
28. obrišiAranžman(aranžmanId)
29. prikažiSlikeAranžmana(aranžmanId)
30. otvoriFileExplorerZaUpload

31. dodajNovuSlikuAranžmana(file, aranžmanId)
32. obrišiSlikuAranžmana(slikaAranžmanaId)
33. prikažiListuSvihProizvoda
34. izmeniPodatkeProizvoda(proizvodId)
35. obrišiProizvod(proizvodId)
36. prikažiListuSlikaProizvoda(proizvodId)
37. dodajNovuSlikuProizvoda(file, proizvodId)
38. obrišiSlikuProizvoda(slikaProizvodaId)
39. prikažiListuPolazakaAranžmana(aranžmanId)
40. prikažiFormuDodavanjaNovogPolaskaAranžmana(aranžmanId)
41. dodajNoviPolazakAranžmana(aranžmanId)
42. izmeniPodatkePolaskaAranžmana(polazakAranžmanaId)
43. obrišiPolazakAranžmana(polazakAranžmanaId)
44. prikažiRezervacijePolaskaAranžmana(polazakAranžmanaId)
45. proveriEmailKorisnikaZaRezervacijuPolaskaAranžmana(emailKor, polazakAranžmanaId)
46. dodajNovuRezervacijuPolaskaAranžmana(polazakAranžmanaId)
47. obrišiRezervacijuPolaskaAranžmana(rezervacijaId)
48. pretragaKorisnika(ključnaReč)
49. pretragaProizvoda(ključnaReč)
50. prikažiTabeluIzveštajaOProdajiProizvoda(listaProizvoda)
51. prikažiTabeluIzveštajaBudućihPutovanja
52. prikažiTabeluIzveštajaProšlihPutovanja
53. prikažiTabeluIzveštajaAranžmana(listaAranžmana)
54. sortirajTabeluIzveštaja(sortParam, sortDirection)

Ugovor **UG1**: registracijaKorisnika
Operacija: registracijaKorisnika(ime, prezime, email, šifra, brojTelefona)
Veza sa SK: SK1
Preduslovi: -
Postuslovi: Korisnik je registrovan i unet u bazu podataka

Ugovor **UG3**: prikažiListuPolazakaAranžmana
Operacija: prikažiListuPolazakaAranžmana(aranžmanId)
Veza sa SK: SK3
Preduslovi: Izabrani aranžman postoji u bazi podataka
Postuslovi: Prikazana je lista polazaka izabranog aranžmana

Ugovor **UG5**: sortirajProizvodePoCeni

Operacija: sortirajProizvodePoCeni(sortDirection)

Veza sa SK: SK5

Preduslovi: Proizvodi postoje u bazi podataka

Postuslovi: Proizvodi su sortirani po ceni u izabranom redosledu

Ugovor **UG7**: prijavljivanje

Operacija: prijavljivanje(email, šifra)

Veza sa SK: SK7

Preduslovi: Korisnik sa datim emailom i šifrom postoji u bazi podataka

Postuslovi: Korisnik je prijavljen i prikazana je početna stranica

Ugovor **UG9**: rezervišiPolazakAranžmana

Operacija: rezervišiPolazakAranžmana(polazakId)

Veza sa SK: SK9

Preduslovi: Korisnik je prijavljen i polazak aranžmana postoji u bazi podataka

Postuslovi: Napravljena je rezervacija korisnika u bazi podataka za odabrani polazak aranžmana

Ugovor **UG13**: dodajProizvodUKorisničkuKorpu

Operacija: dodajProizvodUKorisničkuKorpu(korisnikId, proizvodId, izabranaKolicina)

Veza sa SK: SK13

Preduslovi: Korisnik je prijavljen, proizvod postoji u bazi podataka, količina proizvoda u magacinu je veća ili jednaka izabranoj količine za dodavanje u korpu

Postuslovi: Proizvod je dodat u korisničku korpu

Ugovor **UG14**: izbaciStavkuKorpe

Operacija: izbaciStavkuKorpe(stavkaKorpeId)

Veza sa SK: SK14

Preduslovi: Stavka korpe postoji u bazi podataka

Postuslovi: Obrisana stavka korpe iz baze podataka

Ugovor **UG15**: izračunajNovuCenuPorudžbine

Operacija: izračunajNovuCenuPorudžbine(clanKartPopustId)

Veza sa SK: SK15

Preduslovi: Izabrani popust članske kartice postoji u bazi podataka I korisnik ima dovoljno skupljenih bodova na kartici

Postuslovi: Izračunata nova cena porudžbine

Ugovor **UG16**: prikažiFormuPodatakaODostavi

Operacija: prikažiFormuPodatakaODostavi(listaStavkiKorpe, korisnikId)

Veza sa SK: SK15

Preduslovi: Korisnik je prijavljen i postoji bar jedna stavka korpe u bazi podataka za prijavljenog korisnika

Postuslovi: Prikazana je forma za unos podataka o dostavi

Ugovor **UG17**: napraviNovuNarudžbenicu

Operacija: napraviNovuNarudžbenicu(listaStavkiKorpe, korisnikId)

Veza sa SK: SK15

Preduslovi: Korisnik je prijavljen i prikazana je forma za podatke o dostavi

Postuslovi: Napravljena nova narudžbenica u bazi podataka za prijavljenog korisnika i prikazana potvrda o tome

Ugovor **UG26**: napraviNoviAranžman

Operacija: napraviNoviAranžman

Veza sa SK: SK23

Preduslovi: Administrator je prijavljen

Postuslovi: Napravljen je novi aranžman u bazi podataka

Ugovor **UG27**: sačuvajIzmenjenePodatkeAranžmana

Operacija: sačuvajIzmenjenePodatkeAranžmana(aranžmanId)

Veza sa SK: SK24

Preduslovi: Administrator je prijavljen, izabrani aranžman postoji u bazi podataka

Postuslovi: Izmenjeni podaci izabranog aranžmana su evidentirani u bazi podataka

Ugovor **UG28**: obrišiAranžman

Operacija: obrišiAranžman(aranžmanId)

Veza sa SK: SK25

Preduslovi: Administrator je prijavljen, izabrani aranžman postoji u bazi podataka

Postuslovi: Izabrani aranžman je obrisan iz baze podataka

Ugovor **UG31**: dodajNovuSlikuAranžmana

Operacija: dodajNovuSlikuAranžmana(file, aranžmanId)

Veza sa SK: SK27

Preduslovi: Administrator je prijavljen, fajl slike postoji i aranžman postoji u bazi podataka

Postuslovi: Dodata je nova slika aranžmana u bazu podataka i sačuvan fajl slike u fajl sistem aplikacije

Ugovor **UG32**: obrišiSlikuAranžmana

Operacija: obrišiSlikuAranžmana(slikaAranžmanaId)

Veza sa SK: SK28

Preduslovi: Administrator je prijavljen, slika aranžmana postoji u bazi podataka i fajl slike aranžmana postoji u fajl sistemu aplikacije

Postuslovi: Slika aranžmana je obrisana iz baze podataka i fajl slike aranžmana je obrisan iz fajl sistema aplikacije

Ugovor **UG39**: prikažiListuPolazakaAranžmana

Operacija: prikažiListuPolazakaAranžmana(aranžmanId)

Veza sa SK: SK35

Preduslovi: Administrator je prijavljen, aranžman postoji u bazi podataka

Postuslovi: Pročitana je iz baze podataka i prikazana lista polazaka izabranog aranžmana

Ugovor **UG45**: proveriEmailKorisnikaZaRezervacijuPolaskaAranžmana

Operacija: proveriEmailKorisnikaZaRezervacijuPolaskaAranžmana(emailKor, polazakAranžmanaId)

Veza sa SK: SK40

Preduslovi: Administrator je prijavljen, unešen je email korisnika, polazak aranžmana postoji u bazi podataka

Postuslovi: Korisnik sa zadatom email adresom nema rezervaciju za izabrani polazak aranžmana u bazi podataka

Ugovor **UG48**: pretragaKorisnika

Operacija: pretragaKorisnika(ključnaReč)

Veza sa SK: SK42

Preduslovi: Administrator je prijavljen, parametar ključne reči je definisan

Postuslovi: Pročitani su iz baze podataka i prikazani korisnik/ci koji odgovaraju zadatoj ključnoj reči

Ugovor **UG50**: prikažiTabeluIzveštajaOProdajiProizvoda

Operacija: prikažiTabeluIzveštajaOProdajiProizvoda(listaProizvoda)

Veza sa SK: SK44

Preduslovi: Menadžer je prijavljen, tabela proizvod postoji u bazi podataka

Postuslovi: Pročitani su iz baze podataka proizvod/i i prikazana je tabela njihovih izveštaja o prodaji

Ugovor **UG54**: sortirajTabeluIzveštaja

Operacija: sortirajTabeluIzveštaja(sortParam, sortDirection)

Veza sa SK: SK48

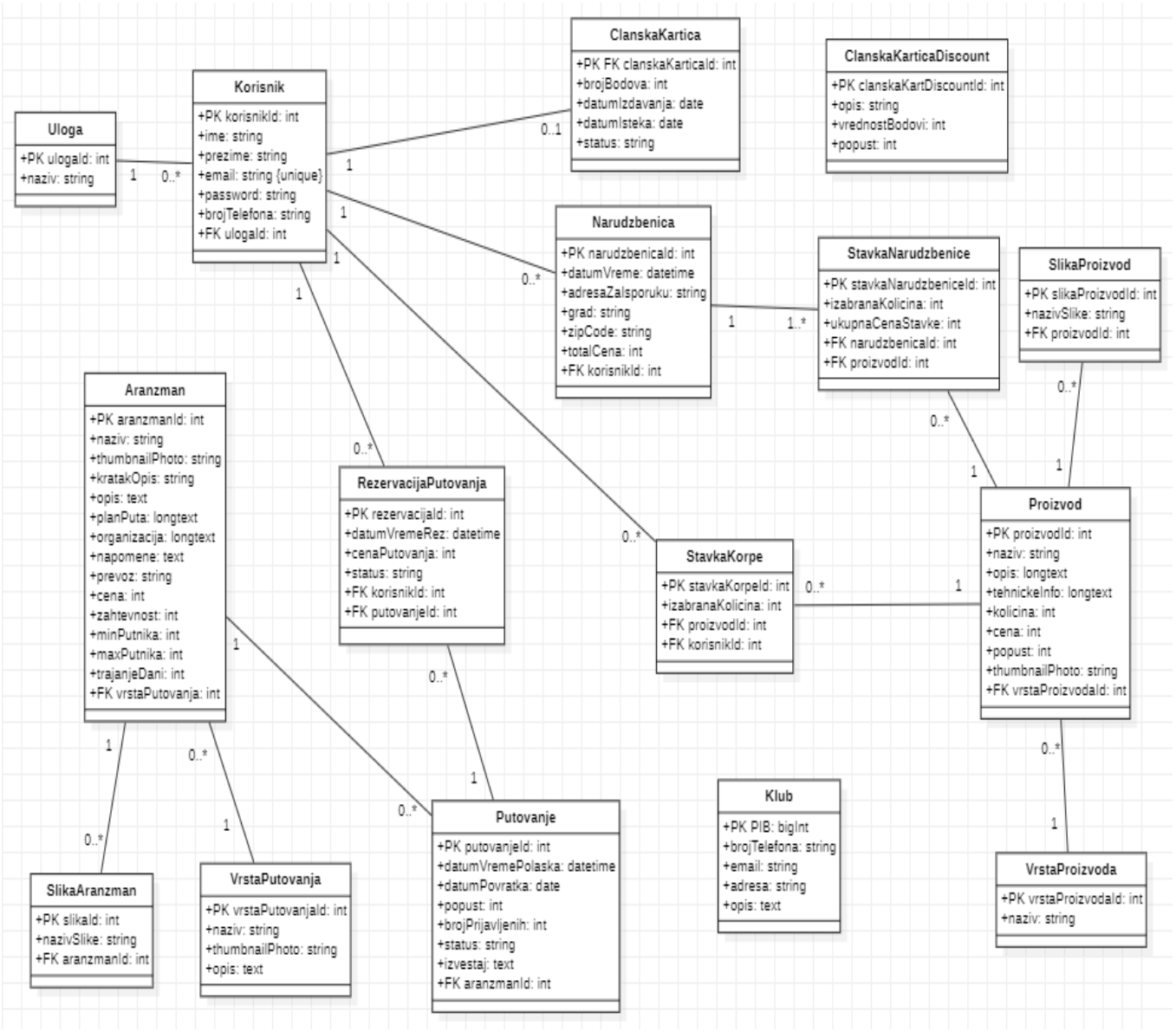
Preduslovi: Menadžer je prijavljen, parametrima sortParam i sortDirection je zadata vrednost

Postuslovi: Prikazana je sortirana tabela izveštaja po zadatim parametrima

2.2.3. Konceptualni (Domenski) model

Konceptualni (Domenski) model definiše se u okviru faze analize nakon definisanja ugovora, a na osnovu podataka iz funkcionalnog zahteva i slučajeva korišćenja.

Konceptualni model prikazan je na slici 27.



Slika 27. Konceptualni model

2.2.4. Relacioni model

Relacioni model se kreira na osnovu konceptualnog modela i predstavlja osnovu za projektovanje relacione baze podataka.

Klub(PIB, brojTelefona, email, adresa, opis)

ClanskaKarticaDiscount(#clanskaKartDiscountId, opis, vrednostBodovi, popust)

Uloga(#ulogaId, naziv)

Korisnik(#korisnikId, ime, prezime, email, password, brojTelefona, *ulogaId*)

Korisnik(*ulogaId*) referencira **Uloga**(#ulogaId)

ClanskaKartica(*clanskaKarticaId,* brojBodova, datumIzdavanja, datumIsteka, status)

ClanskaKartica(*clanskaKarticaId*) referencira **Korisnik**(#korisnikId)

VrstaPutovanja(#vrstaPutovanjaId, naziv, thumbnailPhoto, opis)

Aranzman(#aranzmanId, naziv, thumbnailPhoto, kratakOpis, opis, planPuta, organizacija, napomene, prevoz, cena zahtevnost, minPutnika, maxPutnika, trajanjeDani, *vrstaPutovanjaId*)

Aranzman(*vrstaPutovanjaId*) referencira **VrstaPutovanja**(#vrstaPutovanjaId)

SlikaAranzman(#slikaId, nazivSlike, *aranzmanId*)

SlikaAranzman(*aranzmanId*) referencira **Aranzman**(#aranzmanId)

Putovanje(#putovanjeId, datumVremePolaska, datumPovratka, popust, brojPrijavljenih, status, izvestaj, *aranzmanId*)

Putovanje(*aranzmanId*) referencira **Aranzman**(#aranzmanId)

RezervacijaPutovanja(#rezervacijaId, datumVremeRez, cenaPutovanja, status, *korisnikId*, *putovanjeId*)

RezervacijaPutovanja(*korisnikId*) referencira **Korisnik**(#korisnikId)

RezervacijaPutovanja(*putovanjeId*) referencira **Putovanje**(#putovanjeId)

VrstaProizvoda(#vrstaProizvodaId, naziv)

Proizvod(#proizvodId, naziv, opis, tehnickeInfo, kolicina, cena, popust, thumbnailPhoto, *vrstaProizvodaId*)

Proizvod(*vrstaProizvodaId*) referencira **VrstaProizvoda**(#vrstaProizvodaId)

SlikaProizvod(#slikaProizvodId, nazivSlike, *proizvodId*)

SlikaProizvod(*proizvodId*) referencira **Proizvod**(#proizvodId)

StavkaKorpe(#stavkaKorpeId, izabranaKolicina, *proizvodId*, *korisnikId*)

StavkaKorpe(*proizvodId*) referencira **Proizvod**(#proizvodId)

StavkaKorpe(*korisnikId*) referencira **Korisnik**(#korisnikId)

Narudzbenica(#narudzbenicaId, datumVreme, adresaZaIsporuku, grad, zipCode, totalCena, *korisnikId*)

Narudzbenica(*korisnikId*) referencira **Korisnik**(#korisnikId)

StavkaNarudzbenice(#stavkaNarudzbeniceId, izabranaKolicina, ukupnaCenaStavke, *narudzbenicaId*, *proizvodId*)

StavkaNarudzbenice(*narudzbenicaId*) referencira **Narudzbenica**(#narudzbenicaId)

StavkaNarudzbenice(*proizvodId*) referencira **Proizvod**(#proizvodId)

2.3. FAZA PROJEKTOVANJA

Faza projektovanja Larmanove metode razvoja softvera je treća faza razvoja, koja opisuje arhitekturu softverskog sistema.

Obuhvata:

- Projektovanje aplikacione logike
- Projektovanje skladišta podataka
- Projektovanje korisničkog interfejsa

U nastavku rada biće detaljno obrađene i prikazane svaka od ove tri stavke faze projektovanja.

2.3.1. Projektovanje aplikacione logike

U ovom segmentu faze projektovanja prave se dijagrami sekvenci i kolaboracioni dijagrami za ugovore o sistemskim operacijama opisanim u prethodnoj fazi analize.

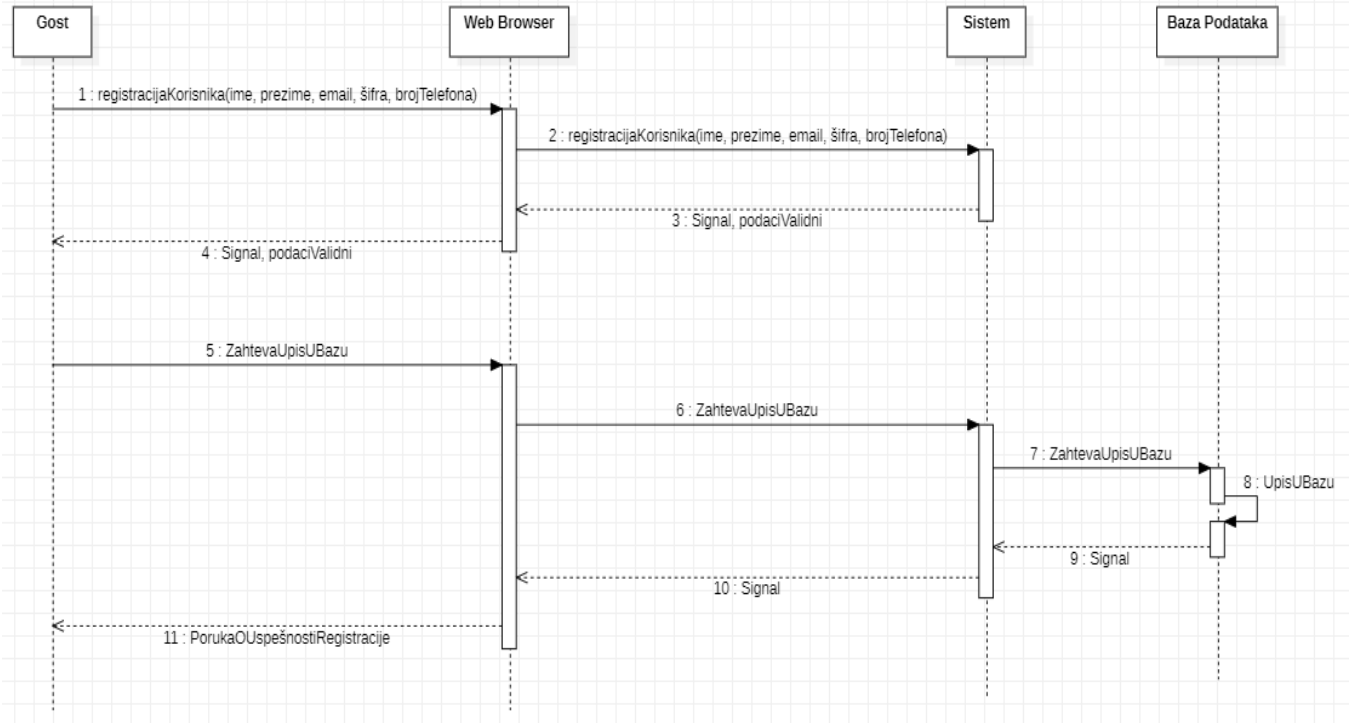
Zbog ograničenog obima završnog rada, u ovom poglavlju ću prikazati 12 karakterističnih ugovora o sistemskim operacijama i njihove dijagrame. Za neke od ugovora biće izostavljeni sekvencijalni ili kolaboracioni dijagram.

Ugovor **UG1**: registracijaKorisnika

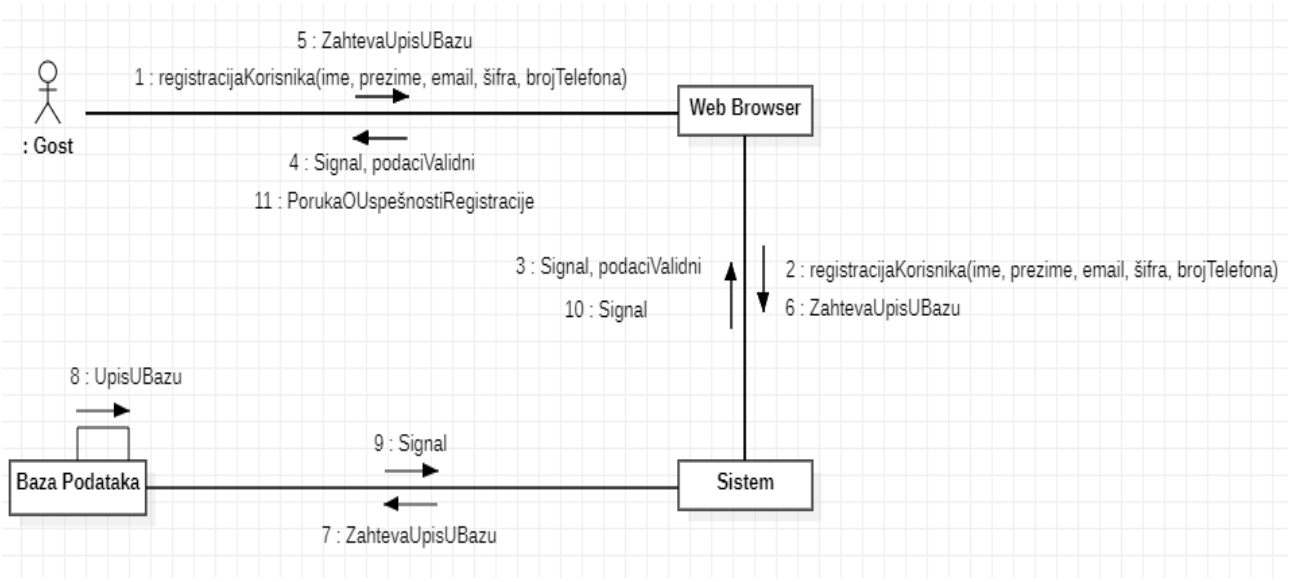
Operacija: registracijaKorisnika(ime, prezime, email, šifra, brojTelefona)

Preduslovi: -

Postuslovi: Korisnik je registrovan i unet u bazu podataka



Slika 28. Dijagram sekvenci UG1 – registracijaKorisnika



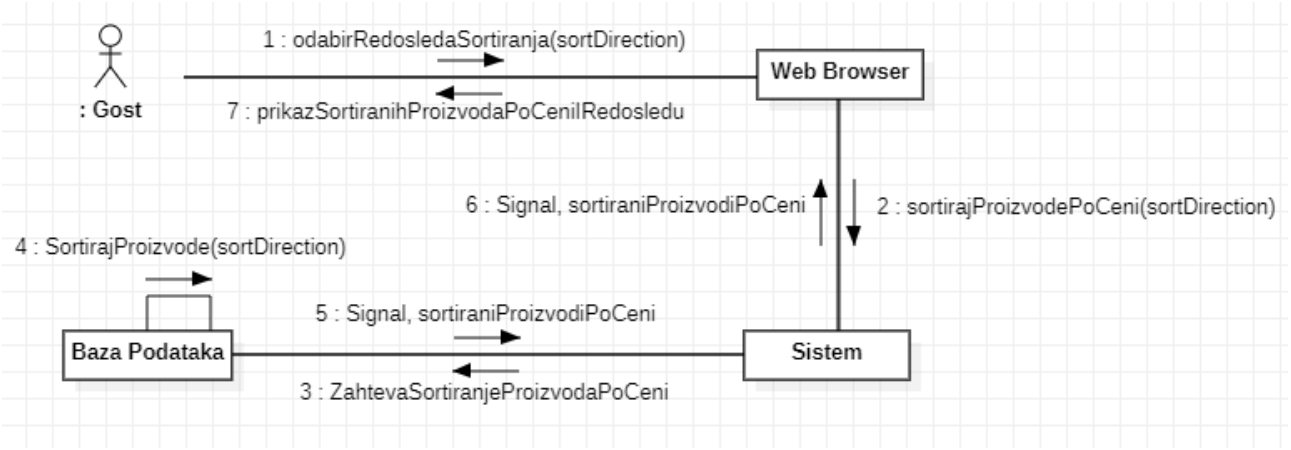
Slika 29. Kolaboracioni dijagram UG1 – registracijaKorisnika

Ugovor **UG5**: sortirajProizvodePoCeni

Operacija: sortirajProizvodePoCeni(sortDirection)

Preduslovi: Proizvodi postoje u bazi podataka

Postuslovi: Proizvodi su sortirani po ceni u izabranom redosledu



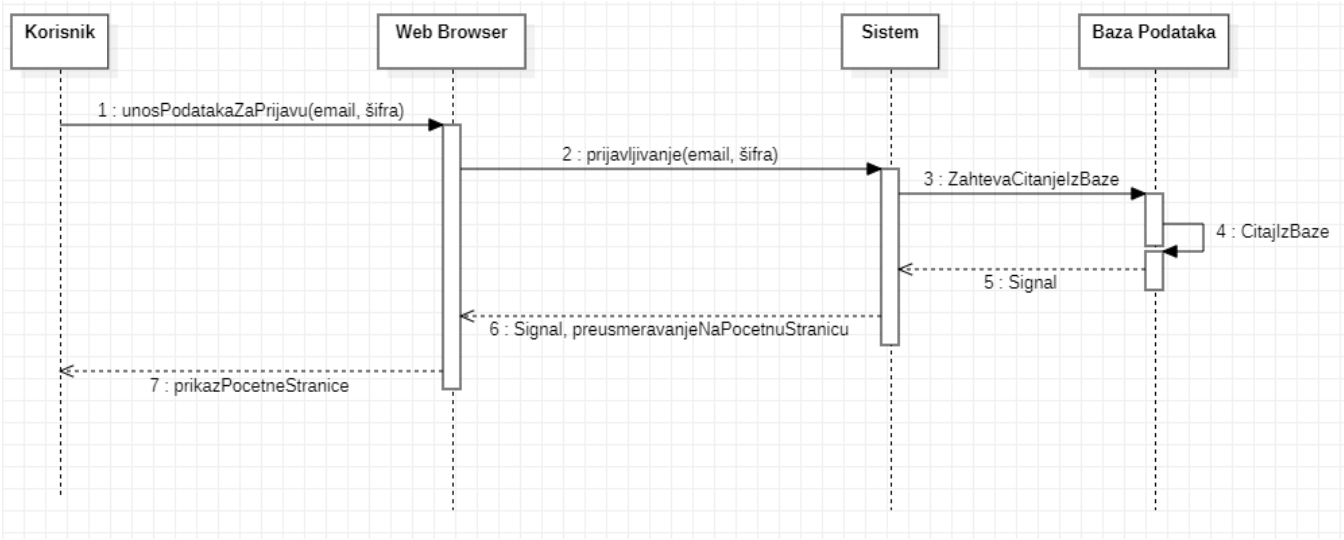
Slika 30. Kolaboracioni dijagram UG5 – sortirajProizvodePoCeni

Ugovor **UG7**: prijavljivanje

Operacija: prijavljivanje(email, šifra)

Preduslovi: Korisnik sa datim emailom i šifrom postoji u bazi podataka

Postuslovi: Korisnik je prijavljen i prikazana je početna stranica



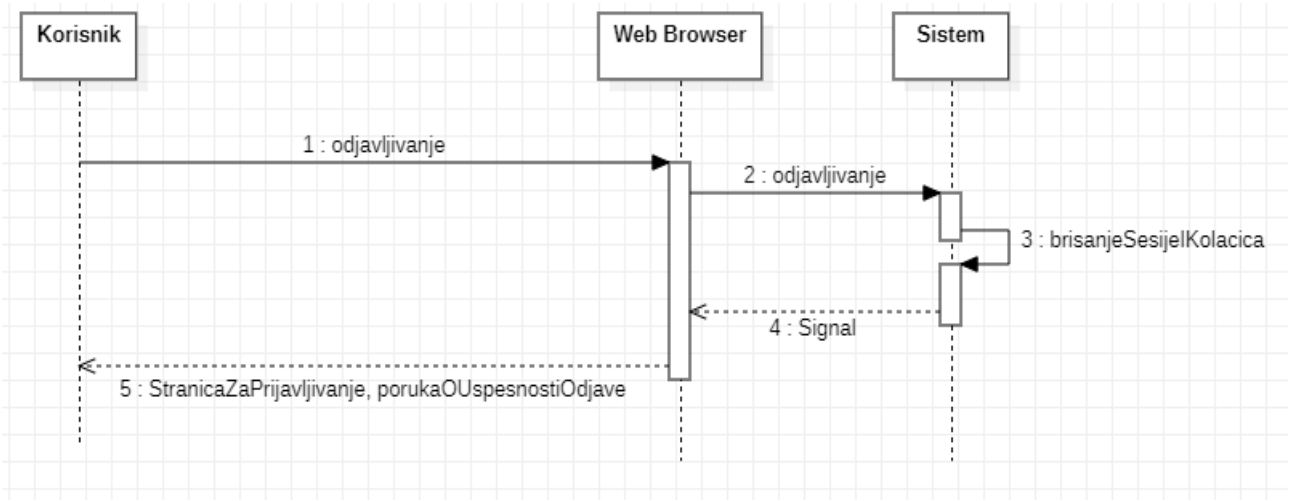
Slika 31. Dijagram sekvenci UG7 – prijavljivanje

Ugovor **UG8**: odjavljivanje

Operacija: odjavljivanje

Preduslovi: Korisnik je prijavljen

Postuslovi: Korisnik je odjavljen i prikazana je forma za prijavljivanje



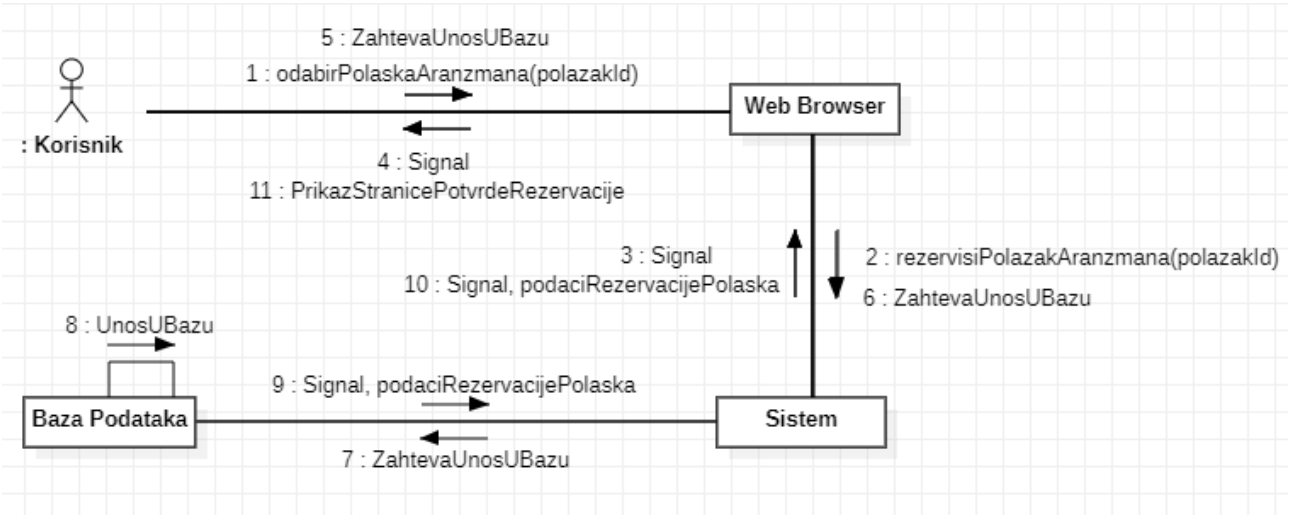
Slika 32. Dijagram sekvenci UG8 – odjavljivanje

Ugovor **UG9**: rezervišiPolazakAranžmana

Operacija: rezervišiPolazakAranžmana(polazakId)

Preduslovi: Korisnik je prijavljen i polazak aranžmana postoji u bazi podataka

Postuslovi: Napravljena je rezervacija korisnika u bazi podataka za odabrani polazak aranžmana



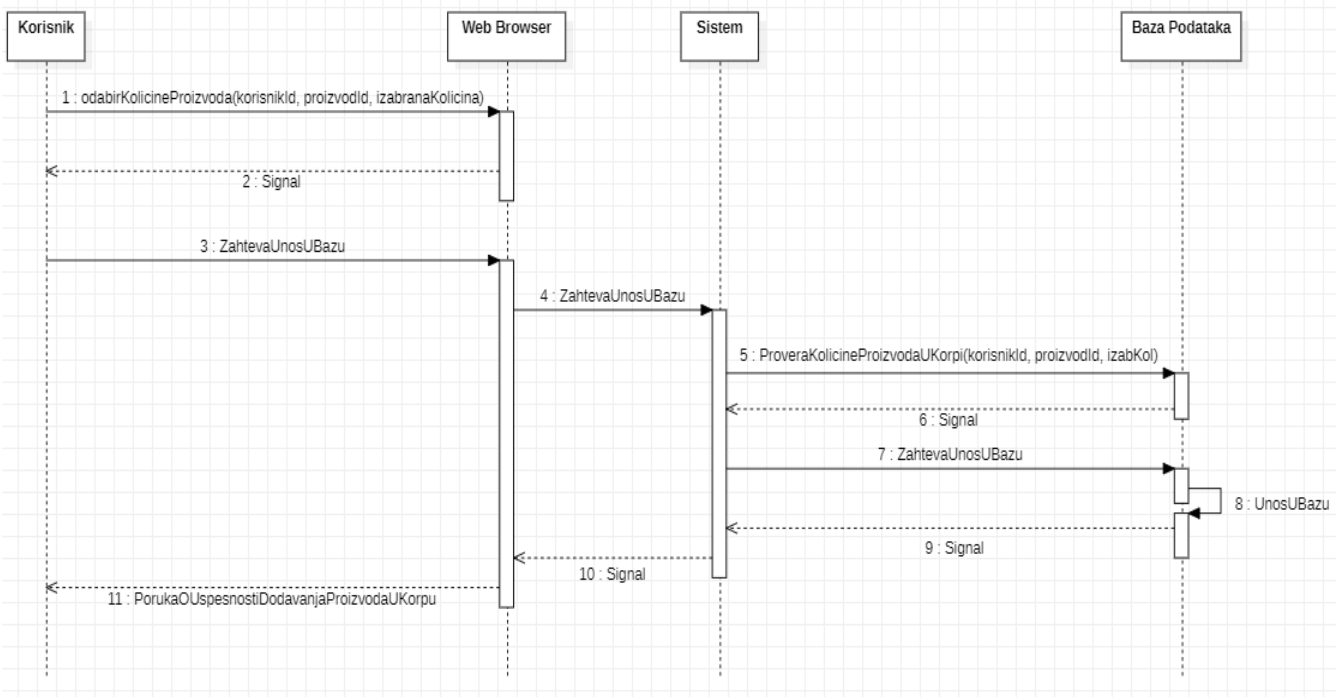
Slika 33. Kolaboracioni dijagram UG9 – rezervišiPolazakAranžmana

Ugovor **UG13**: dodajProizvodUKorisničkuKorpu

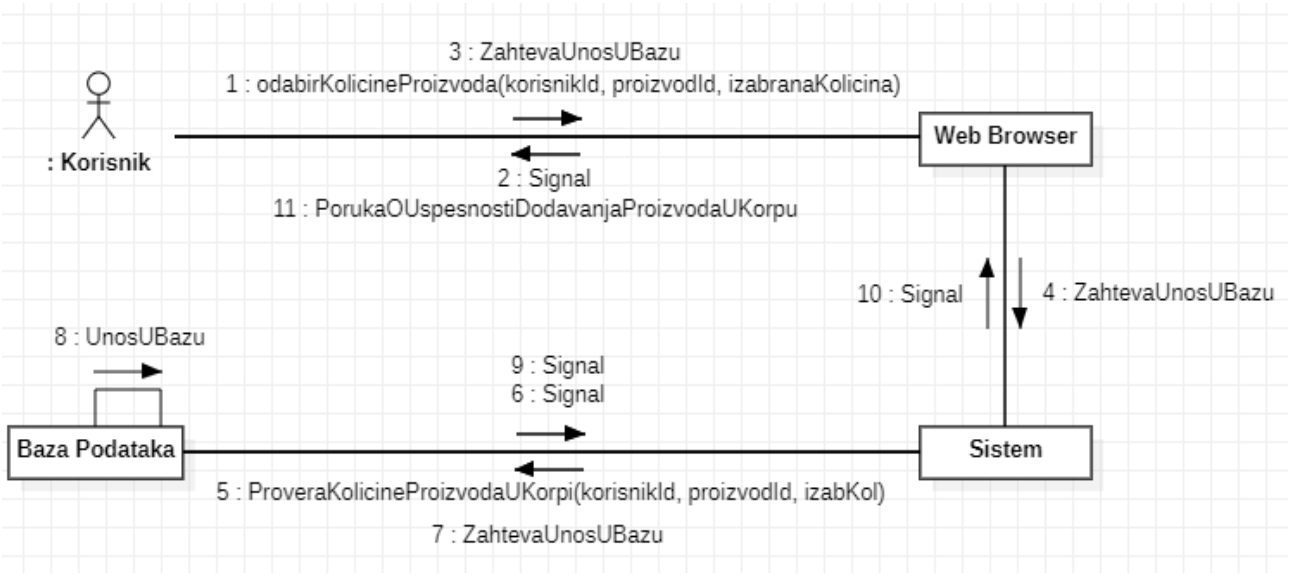
Operacija: dodajProizvodUKorisničkuKorpu(korisnikId, proizvodId, izabranaKolicina)

Preduslovi: Korisnik je prijavljen, proizvod postoji u bazi podataka, količina proizvoda u magacinu je veća ili jednaka izabranoj količine za dodavanje u korpu

Postuslovi: Proizvod je dodat u korisničku korpu



Slika 34. Dijagram sekvenci UG13 – dodajProizvodUKorisničkuKorpu



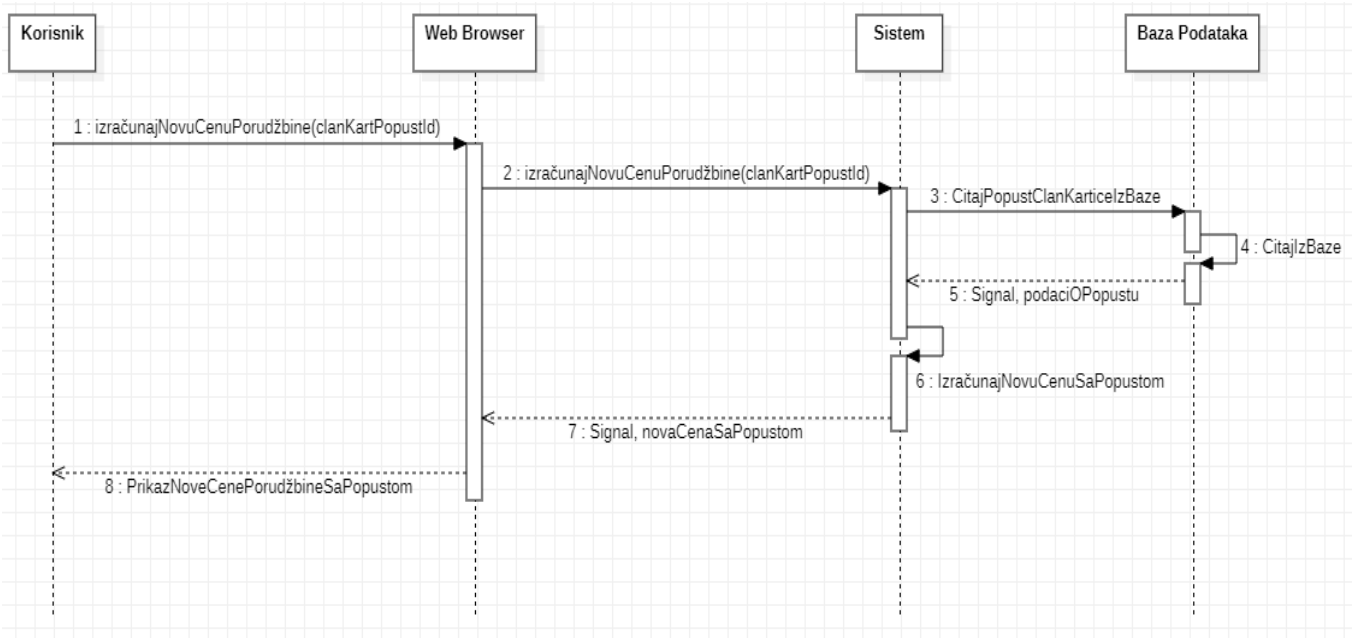
Slika 35. Kolaboracioni dijagram UG13 – dodajProizvodUKorisničkuKorpu

Ugovor **UG15**: izračunajNovuCenuPorudžbine

Operacija: izračunajNovuCenuPorudžbine(clanKartPopustId)

Preduslovi: Izabrani popust članske kartice postoji u bazi podataka

Postuslovi: Izračunata nova cena porudžbine



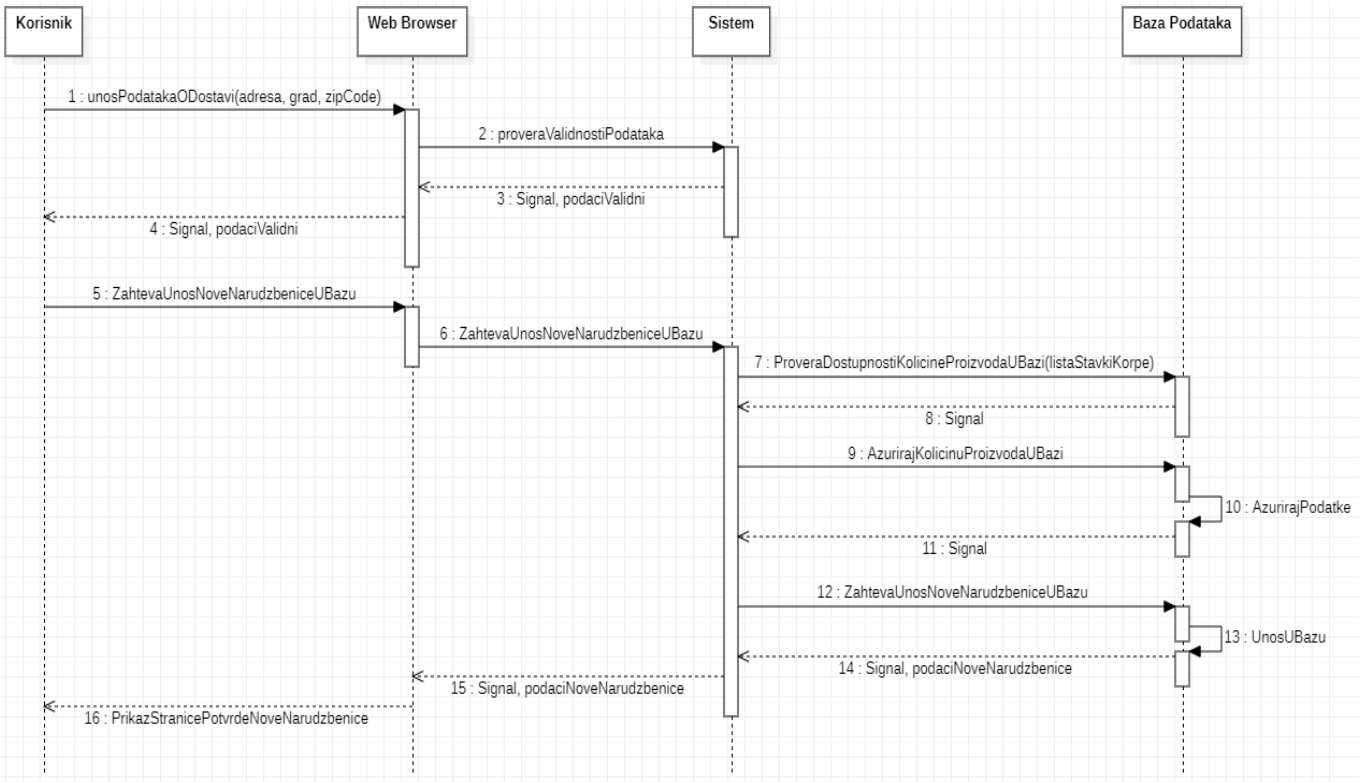
Slika 36. Dijagram sekvenci UG15 – izračunajNovuCenuPorudžbine

Ugovor **UG17**: napraviNovuNarudžbenicu

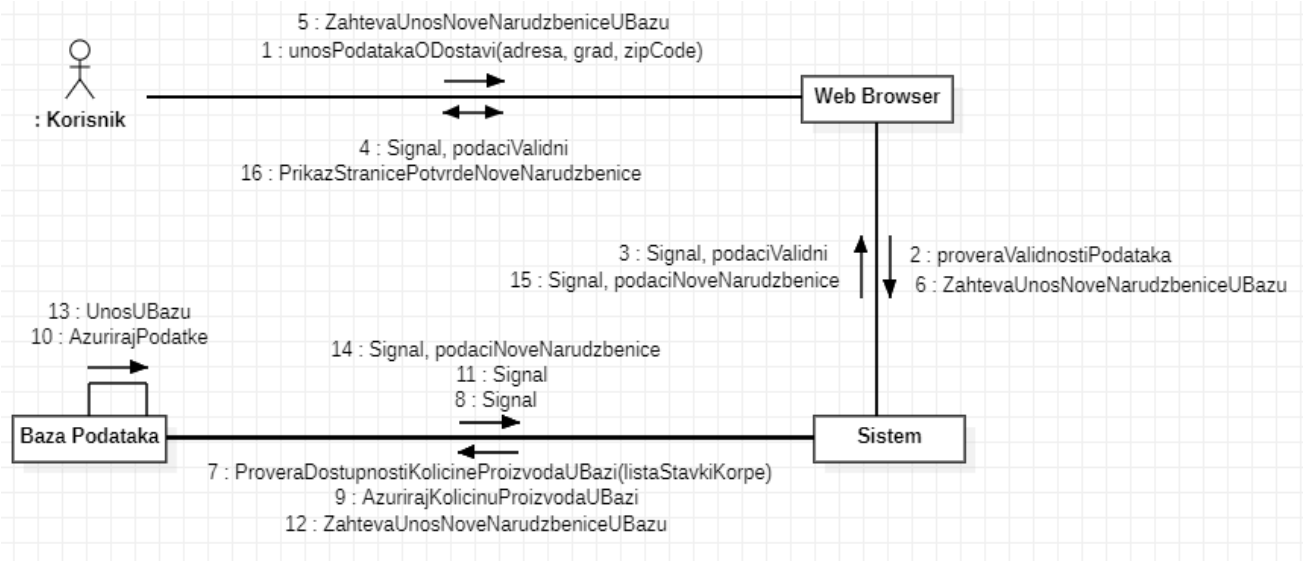
Operacija: napraviNovuNarudžbenicu(listaStavkiKorpe, korisnikId)

Preduslovi: Korisnik je prijavljen i prikazana je forma za podatke o dostavi

Postuslovi: Napravljena nova narudžbenica u bazi podataka za prijavljenog korisnika i prikazana potvrda o tome



Slika 37. Dijagram sekvenci UG17 – napraviNovuNarudžbenicu



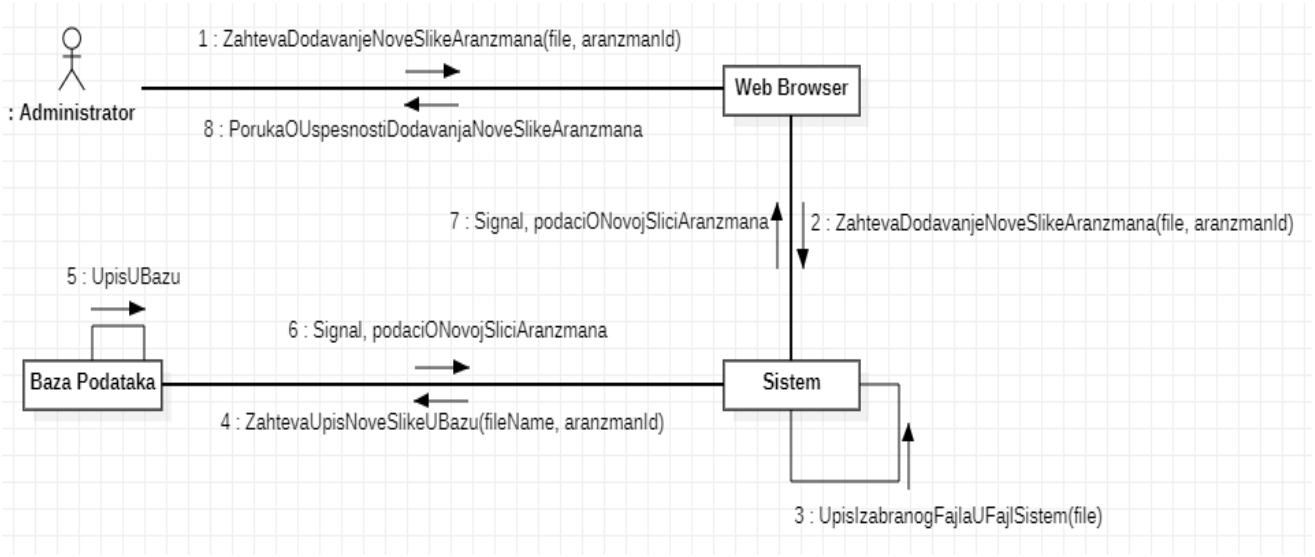
Slika 38. Kolaboracioni dijagram UG17 – napraviNovuNarudžbenicu

Ugovor **UG31**: dodajNovuSlikuAranžmana

Operacija: dodajNovuSlikuAranžmana(file, aranžmanId)

Preduslovi: Administrator je prijavljen, file slike postoji i aranžman postoji u bazi podataka

Postuslovi: Dodata je nova slika aranžmana u bazu podataka i sačuvan file slike u file sistem aplikacije



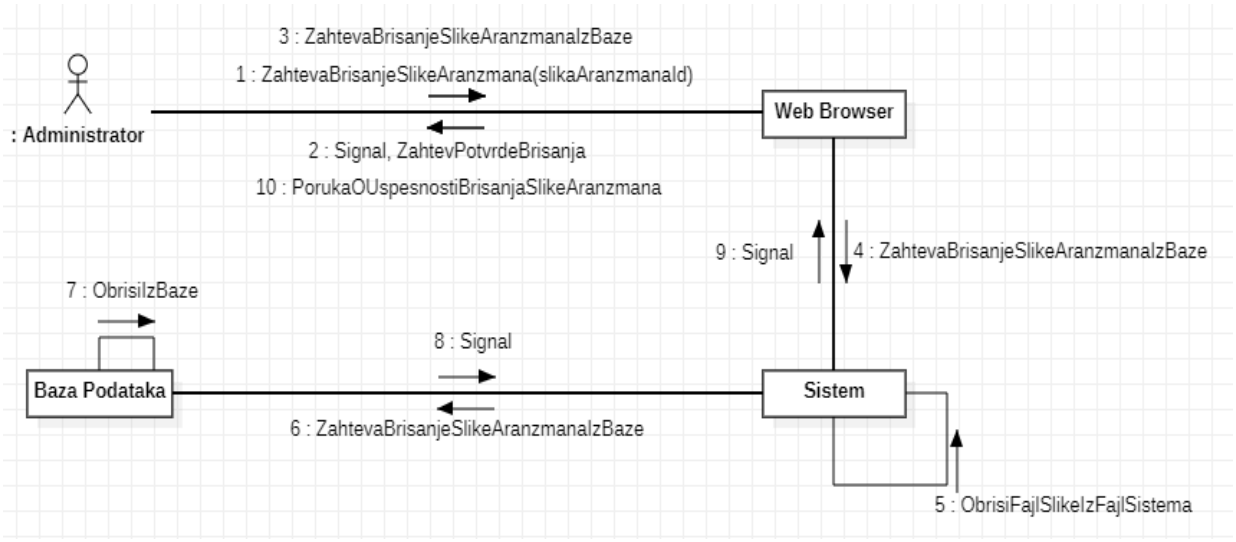
Slika 39. Kolaboracioni dijagram UG31 – dodajNovuSlikuAranžmana

Ugovor **UG32**: obrišiSlikuAranžmana

Operacija: obrišiSlikuAranžmana(slikaAranžmanaId)

Preduslovi: Administrator je prijavljen, slika aranžmana postoji u bazi podataka i file slike aranžmana postoji u file sistemu aplikacije

Postuslovi: Slika aranžmana je obrisana iz baze podataka i file slike aranžmana je obrisan iz file sistema aplikacije



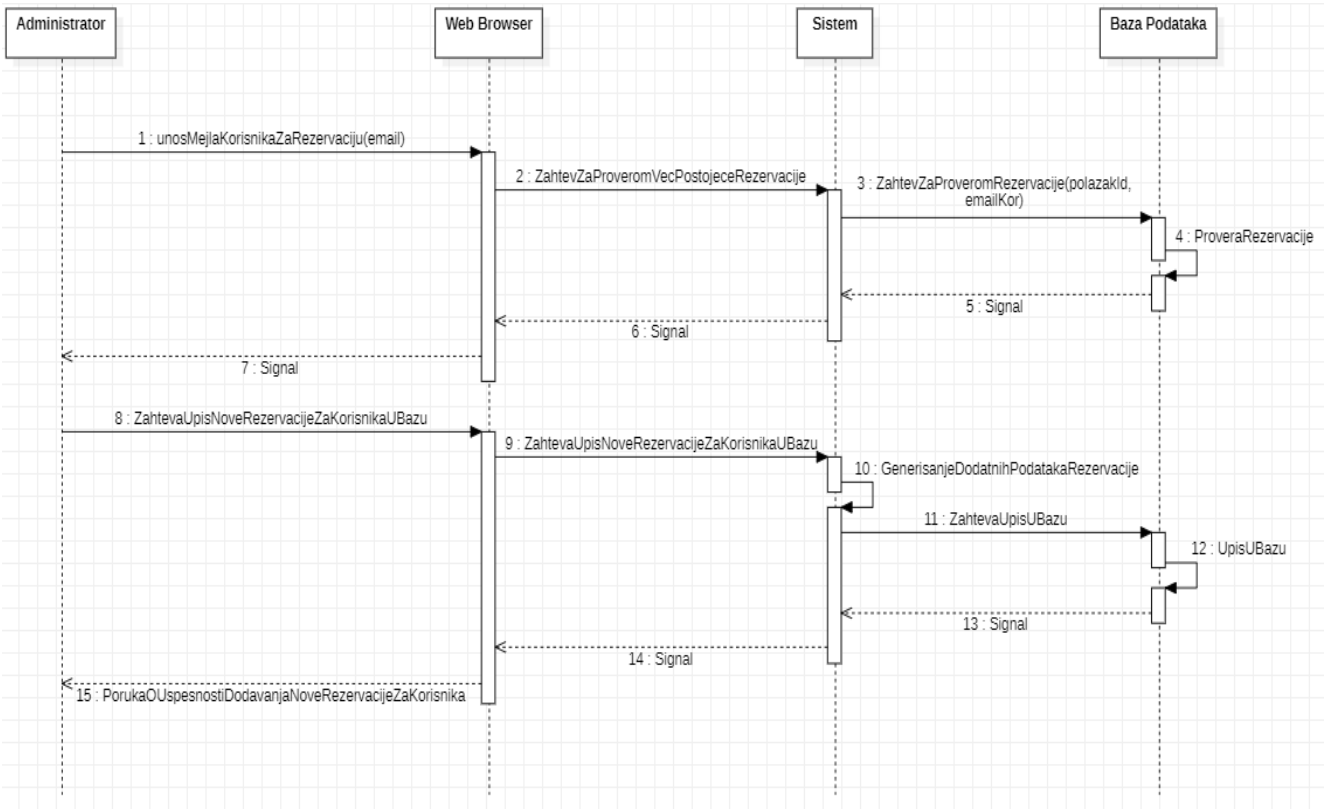
Slika 40. Kolaboracioni dijagram UG32 – obrišiSlikuAranžmana

Ugovor **UG45**: dodajNovuRezervacijuPolaskaAranžmana

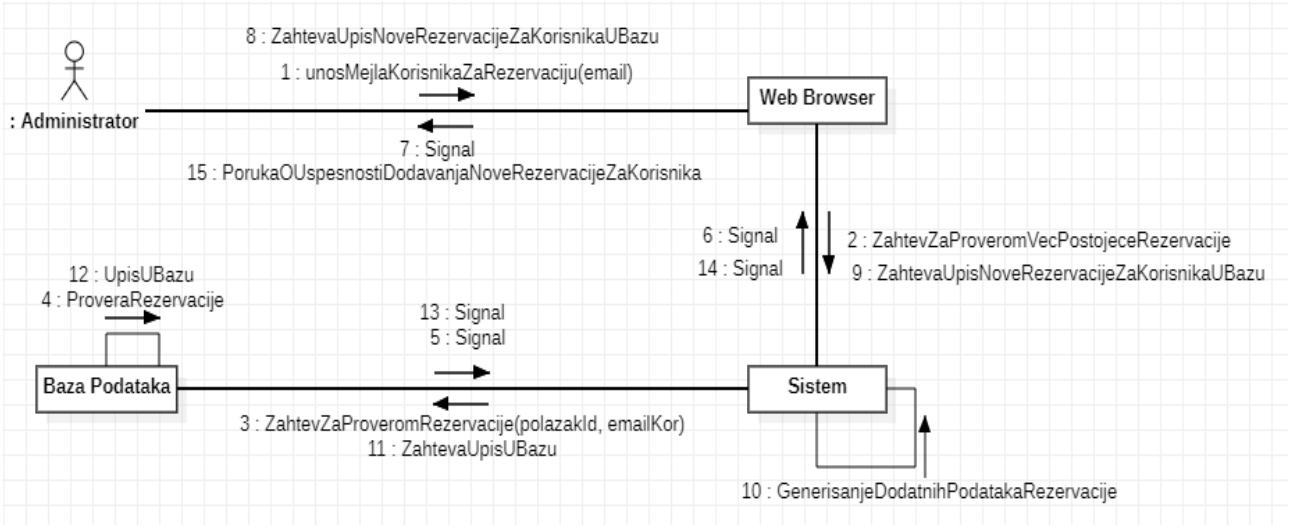
Operacija: dodajNovuRezervacijuPolaskaAranžmana(polazakAranžmanaId)

Preduslovi: Administrator je prijavljen, polazak aranžmana postoji u bazi podataka

Postuslovi: Nova rezervacija polaska aranžmana je sačuvana u bazi podataka



Slika 41. Dijagram sekvenci UG45 – dodajNovuRezervacijuPolaskaAranžmana



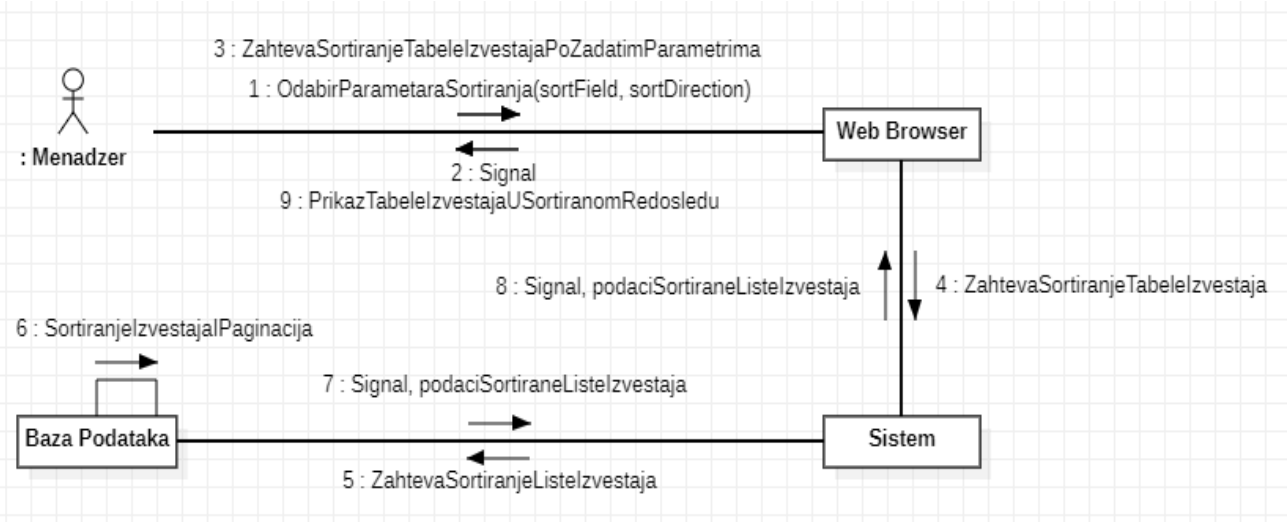
Slika 42. Kolaboracioni dijagram UG45 – dodajNovuRezervacijuPolaskaAranžmana

Ugovor **UG53**: sortirajTabeluIzveštaja

Operacija: sortirajTabeluIzveštaja(sortParam, sortDirection)

Preduslovi: Menadžer je prijavljen, parametrima sortParam i sortDirection je zadata vrednost

Postuslovi: Prikazana je sortirana tabela izveštaja po zadatim parametrima



Slika 43. Kolaboracioni dijagram UG53 – sortirajTabeluIzveštaja

2.3.2. Projektovanje skladišta podataka

Na osnovu prethodno definisanog konceptualnog (domenskog) i relacionog modela, projektuje se skladište podataka. U nastavku su prikazane tabele relacionog sistema za upravljanje bazom podataka, koje nastaju kao rezultat ovog projektovanja.

Tabela *klub*, prikazana je na slici 44.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	**PIB**	bigint(20)			No	*None*		
2	**brojTelefona**	varchar(100)	utf8mb4_general_ci		No	*None*		
3	**email**	varchar(100)	utf8mb4_general_ci		No	*None*		
4	**adresa**	varchar(100)	utf8mb4_general_ci		No	*None*		
5	**opis**	text	utf8mb4_general_ci		Yes	*NULL*		

Slika 44. Tabela – klub

Tabela *clanska_kartica_discount* prikazana je na slici 45.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	clanskaKartDiscountId	int(11)			No	None		AUTO_INCREMENT
2	opis	varchar(150)	utf8mb4_general_ci		No	None		
3	vrednostBodovi	int(11)			No	None		
4	popust	int(11)			No	None		

Slika 45. Tabela – clanska_kartica_discount

Tabela *uloga* prikazana je na slici 46.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	ulogaId	int(11)			No	None		AUTO_INCREMENT
2	naziv	varchar(120)	utf8mb4_general_ci		Yes	NULL		

Slika 46. Tabela – uloga

Tabela *korisnik* prikazana je na slici 47.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	korisnikId	int(11)			No	None		AUTO_INCREMENT
2	ime	varchar(80)	utf8mb4_general_ci		No	None		
3	prezime	varchar(100)	utf8mb4_general_ci		No	None		
4	email	varchar(110)	utf8mb4_general_ci		No	None		
5	password	varchar(80)	utf8mb4_general_ci		No	None		
6	brojTelefona	varchar(60)	utf8mb4_general_ci		No	None		
7	ulogaId	int(11)			No	None		

Slika 47. Tabela – korisnik

Tabela *clanska_kartica* prikazana je na slici 48.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	clanskaKarticaId	int(11)			No	None		
2	brojBodova	int(11)			No	None		
3	datumIzdavanja	date			No	None		
4	datumIsteka	date			Yes	NULL		
5	status	varchar(30)	utf8mb4_general_ci		No	None		

Slika 48. Tabela – clanska_kartica

Tabela *vrsta_putovanja* prikazana je na slici 49.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	**vrstaPutovanjaId**	int(11)			No	*None*		AUTO_INCREMENT
2	**naziv**	varchar(90)	utf8mb4_general_ci		No	*None*		
3	**thumbnailPhoto**	varchar(100)	utf8mb4_general_ci		Yes	*NULL*		
4	**opis**	longtext	utf8mb4_general_ci		No	*None*		

Slika 49. Tabela – vrsta_putovanja

Tabela *aranzman* prikazana je na slici 50.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	**aranzmanId**	int(11)			No	*None*		AUTO_INCREMENT
2	**naziv**	varchar(60)	utf8mb4_general_ci		No	*None*		
3	**thumbnailPhoto**	varchar(100)	utf8mb4_general_ci		Yes	*NULL*		
4	**kratakOpis**	varchar(200)	utf8mb4_general_ci		No	*None*		
5	**opis**	longtext	utf8mb4_general_ci		No	*None*		
6	**planPuta**	longtext	utf8mb4_general_ci		No	*None*		
7	**organizacija**	longtext	utf8mb4_general_ci		No	*None*		
8	**napomene**	text	utf8mb4_general_ci		Yes	*NULL*		
9	**prevoz**	varchar(90)	utf8mb4_general_ci		No	*None*		
10	**cena**	int(11)			No	*None*		
11	**zahtevnost**	varchar(1)	utf8mb4_general_ci		No	*None*		
12	**min_putnika**	int(11)			No	*None*		
13	**max_putnika**	int(11)			No	*None*		
14	**trajanjeDani**	int(11)			No	*None*		
15	**vrstaPutovanjaId**	int(11)			Yes	*NULL*		

Slika 50. Tabela – aranzman

Tabela *slika_aranzman* prikazana je na slici 51.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	**slikaId**	int(11)			No	*None*		AUTO_INCREMENT
2	**nazivSlike**	varchar(150)	utf8mb4_general_ci		No	*None*		
3	**aranzmanId**	int(11)			No	*None*		

Slika 51. Tabela – slika_aranzman

Tabela *putovanje* prikazana je na slici 52.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	**putovanjeId**	int(11)			No	*None*		AUTO_INCREMENT
2	**datumVremePolaska**	datetime			No	*None*		
3	**datumPovratka**	date			No	*None*		
4	**popust**	int(11)			No	*None*		
5	**brojPrijavljenih**	int(11)			No	*None*		
6	**status**	varchar(50)	utf8mb4_general_ci		No	*None*		
7	**izvestaj**	text	utf8mb4_general_ci		Yes	*NULL*		
8	**aranzmanId**	int(11)			No	*None*		

Slika 52. Tabela – putovanje

Tabela *rezervacija_putovanja* prikazana je na slici 53.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	**rezervacijaId**	int(11)			No	*None*		AUTO_INCREMENT
2	**datumVremeRez**	datetime			No	*None*		
3	**cenaPutovanja**	int(11)			Yes	*NULL*		
4	**status**	varchar(40)	utf8mb4_general_ci		No	*None*		
5	**korisnikId**	int(11)			No	*None*		
6	**putovanjeId**	int(11)			No	*None*		

Slika 53. Tabela – rezervacija_putovanja

Tabela *vrsta_proizvoda* prikazana je na slici 54.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	vrstaProizvodaId	int(11)			No	None		AUTO_INCREMENT
2	naziv	varchar(50)	utf8mb4_general_ci		No	None		

Slika 54. Tabela – vrsta_proizvoda

Tabela *proizvod* prikazana je na slici 55.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	proizvodId	int(11)			No	None		AUTO_INCREMENT
2	naziv	varchar(150)	utf8mb4_general_ci		No	None		
3	opis	longtext	utf8mb4_general_ci		No	None		
4	tehnickeInfo	longtext	utf8mb4_general_ci		Yes	NULL		
5	kolicina	int(11)			No	None		
6	cena	int(11)			No	None		
7	popust	int(11)			No	None		
8	thumbnailPhoto	varchar(150)	utf8mb4_general_ci		Yes	NULL		
9	vrstaProizvodaId	int(11)			Yes	NULL		

Slika 55. Tabela – proizvod

Tabela *slika_proizvod* prikazana je na slici 56.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	slikaProizvodId	int(11)			No	None		AUTO_INCREMENT
2	nazivSlike	varchar(110)	utf8mb4_general_ci		No	None		
3	proizvodId	int(11)			No	None		

Slika 56. Tabela – slika_proizvod

Tabela *stavka_korpe* prikazana je na slici 57.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	stavkaKorpeId	int(11)			No	None		AUTO_INCREMENT
2	izabranaKolicina	int(11)			No	None		
3	proizvodId	int(11)			No	None		
4	korisnikId	int(11)			No	None		

Slika 57. Tabela – stavka_korpe

Tabela *stavka_narudzbenice* prikazana je na slici 58.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	stavkaNarudzbeniceId	int(11)			No	None		AUTO_INCREMENT
2	izabranaKolicina	int(11)			No	None		
3	ukupnaCenaStavke	int(11)			No	None		
4	narudzbenicaId	int(11)			No	None		
5	proizvodId	int(11)			No	None		

Slika 58. Tabela – stavka_narudzbenice

Tabela *narudzbenica* prikazana je na slici 59.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	narudzbenicaId	int(11)			No	None		AUTO_INCREMENT
2	datumVreme	datetime			No	None		
3	adresa_za_isporuku	varchar(100)	utf8mb4_general_ci		No	None		
4	grad	varchar(60)	utf8mb4_general_ci		No	None		
5	zip_code	varchar(20)	utf8mb4_general_ci		No	None		
6	totalCena	int(11)			No	None		
7	korisnikId	int(11)			No	None		

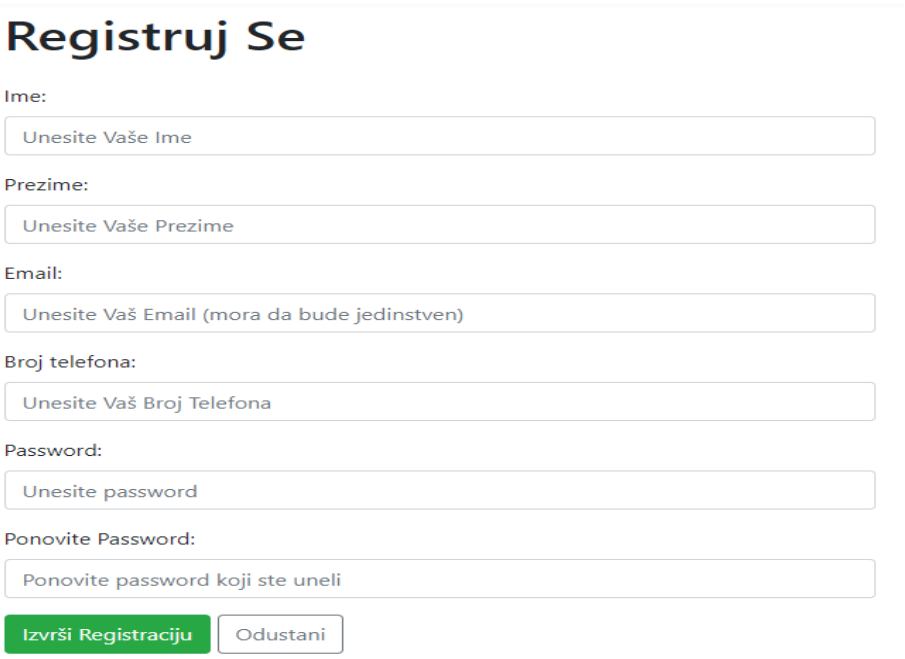
Slika 59. Tabela – narudzbenica

2.3.3. Projektovanje korisničkog interfejsa

Korisnički interfejs predstavlja grafičke, tekstualne i audio informacije koje program predstavlja korisniku. Podrazumeva i kontrolne sekvence koje korisnik koristi tokom upotrebe programa (npr. taster sa tastature, pokreti mišem, selekcija kod ekrana osetljivih na dodir). U nastavku će biti prikazan korisnički interfejs predstavljen kroz ekranske forme za neke od slučajeva korišćenja softvera, kao i opis akcija. Zbog ograničenog obima završnog rada, biće prikazani korisnički interfejsi za nekoliko alternativnih scenarija.

SK1: Registracija

Preduslovi: Sistem je aktivan i prikazana je veb strana forme za registraciju



Slika 60. Projektovanje korisničkog intefejsa SK1 – Registracija

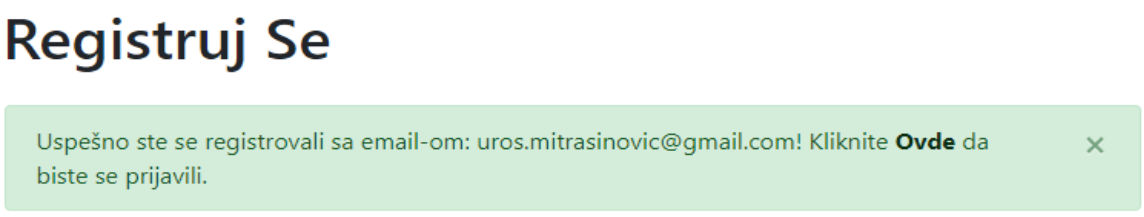
Osnovni scenario:

1. Gost unosi podatke neophodne za registraciju (APUSO)
2. Gost poziva sistem da obradi podatke za registraciju (APSO)

Opis akcije: Gost pritiska dugme 

, čime poziva sistem da pokrene sistemsku operaciju *registracijaKorisnika(ime, prezime, email, šifra, brojTelefona)*.

3. Sistem obrađuje unete podatke i proverava validnost (SO)
4. Sistem prikazuje gostu poruku o uspešnoj registraciji (IA)



Slika 61. Projektovanje korisničkog intefejsa SK1 – Poruka o uspešnoj registraciji

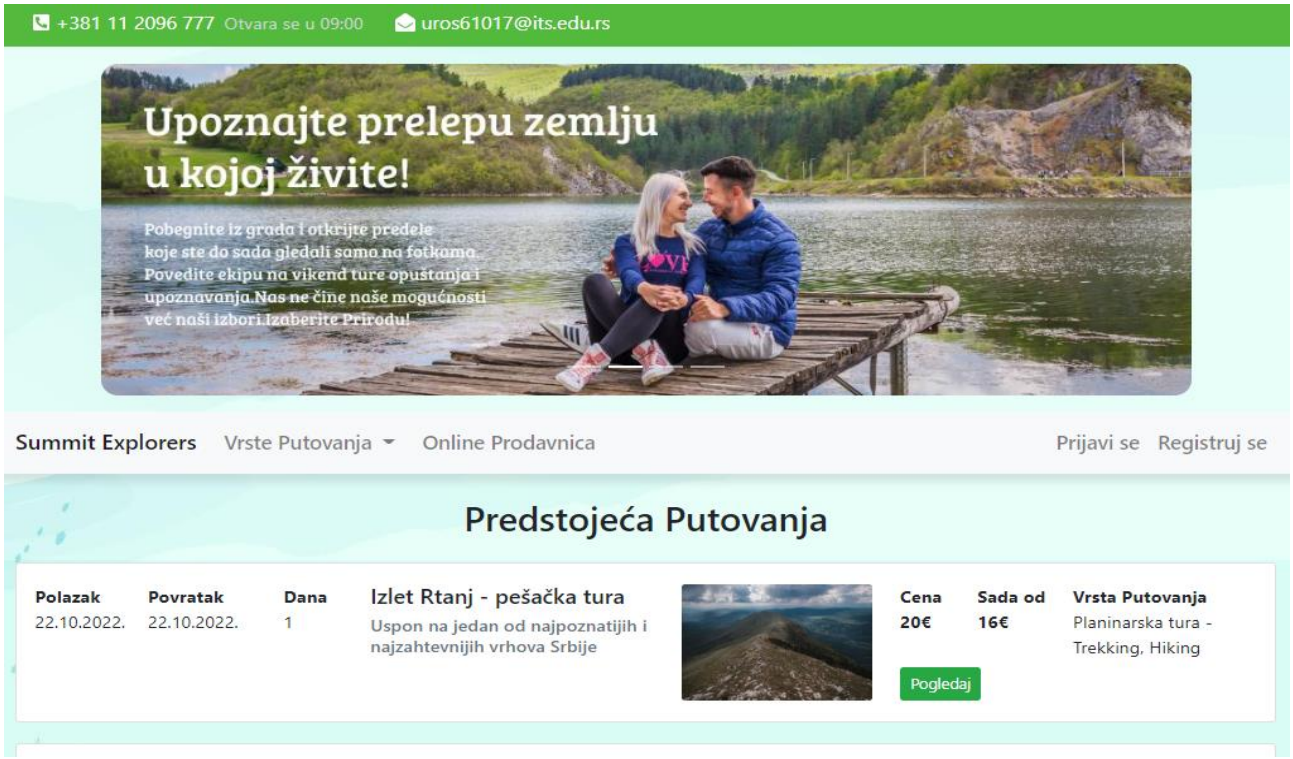
Alternativni scenario:

3.2. Sistem je utvrdio da gost nije uneo validne podatke za registraciju i prikazuje odgovarajuću poruku gostu (IA)

Slika 62. Projektovanje korisničkog intefejsa SK1 – Neuspešna validacija podataka

SK4: Pregled prodavnice

Preduslovi: Sistem je aktivan, prikazana je početna stranica (slika 63)



Slika 63. Projektovanje korisničkog intefejsa SK4 – Početna stranica

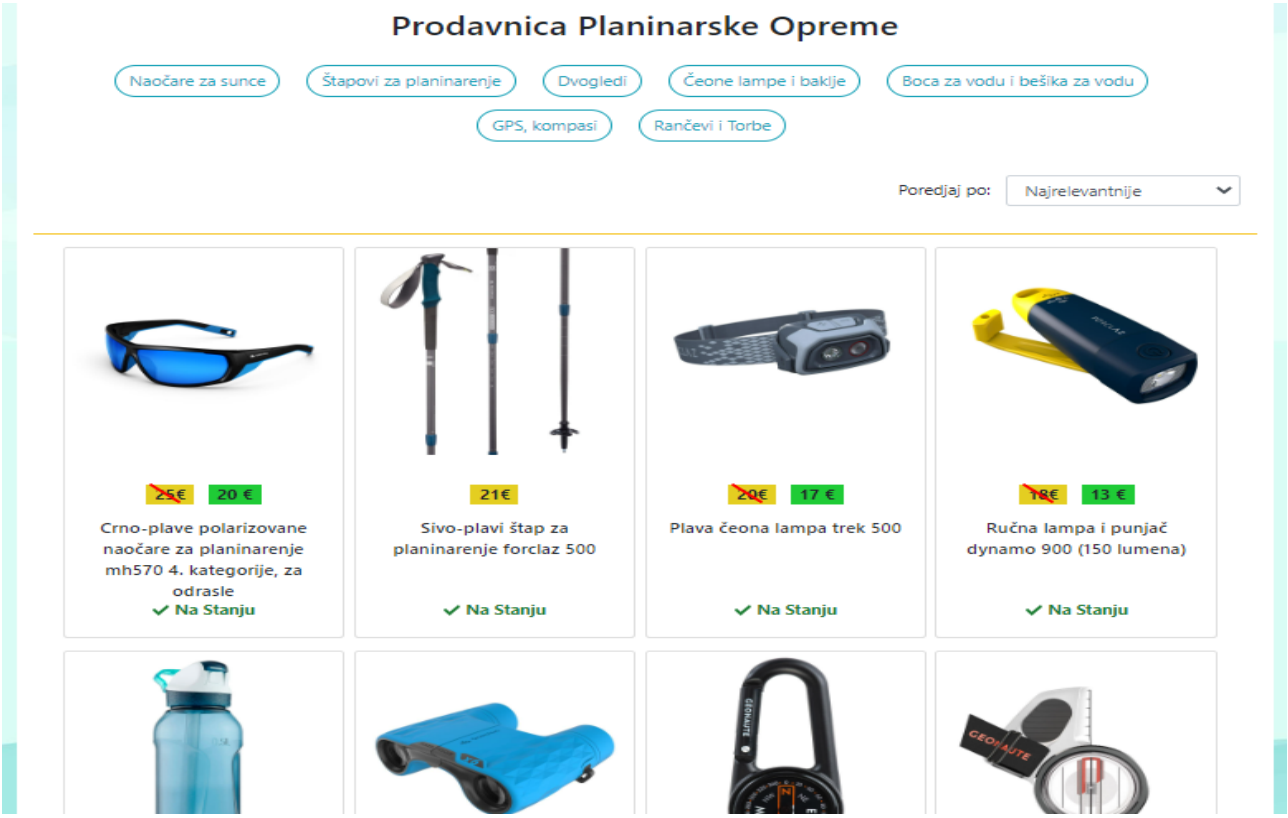
Osnovni scenario:

1. Gost poziva sistem da prikaže prodavnicu (APSO)

Opis akcije: Gost u navigacionom baru pritiska opciju 

, čime poziva sistem da pokrene sistemsku operaciju *prikažiProdavnicu*.

2. Sistem obrađuje zahtev za prikazom prodavnice (SO)
3. Sistem prikazuje gostu prodavnicu (IA)



Slika 64. Projektovanje korisničkog interfejsa SK4 – Prikaz prodavnice

SK7: Prijavljivanje

Preduslovi: Sistem je aktivan i prikazana je forma za prijavljivanje

Osnovni scenario:

1. Korisnik unosi lične podatke vezane za prijavu na sistem (email, password) (APUSO)

Slika 65. Projektovanje korisničkog interfejsa SK7 – Forma za prijavu, unos kredencijala

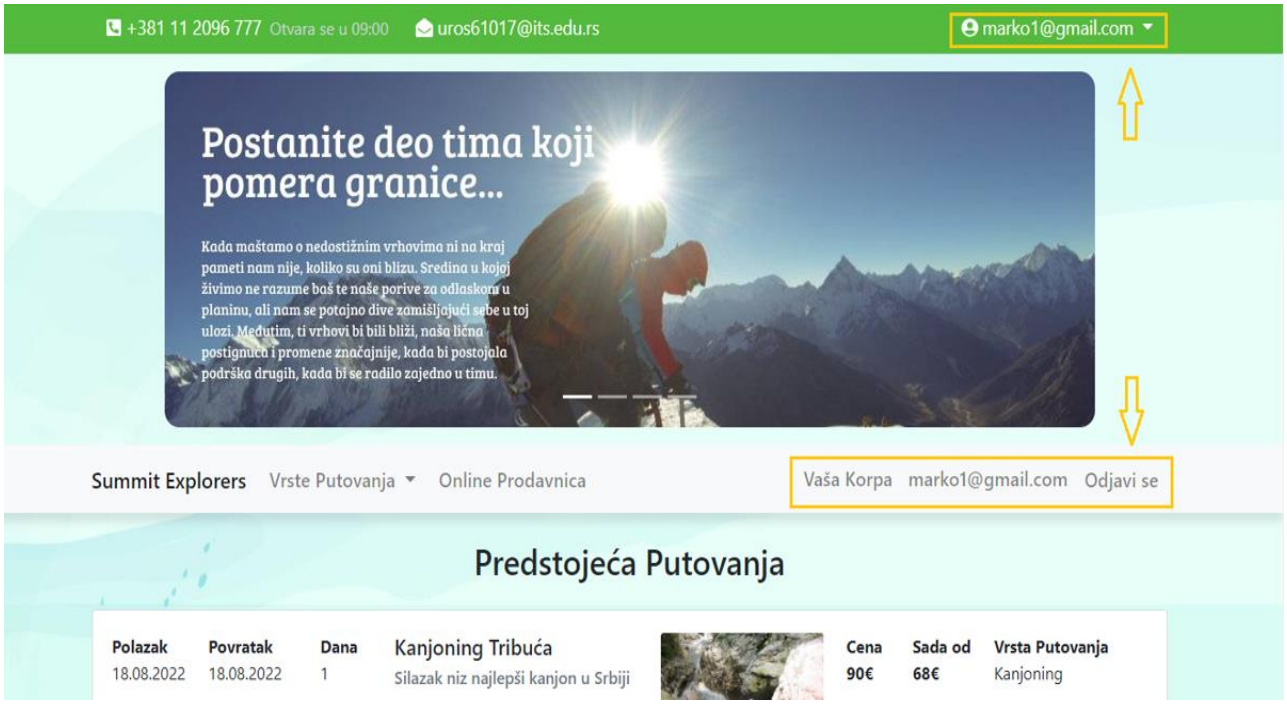
2. Korisnik poziva sistem da izvrši prijavljivanje sa unetim podacima (APSO)

Opis akcije: Korisnik pritiska dugme 

, čime poziva sistem da pokrene sistemsku operaciju *prijavljivanje(email, šifra)*.

3. Sistem izvršava prijavljivanje korisnika (SO)

4. Sistem prikazuje korisniku početnu stranicu (IA)



Slika 66. Projektovanje korisničkog interfejsa SK7 – Početna stranica prijavljenog korisnika

SK9: Rezervacija polaska aranžmana

Preduslovi: Sistem je aktivan, korisnik je prijavljen i prikazani su detalji aranžmana sa listom polazaka

Slika 67. Projektovanje korisničkog interfejsa SK9 – Stranica detalja aranžmana, sekcija Polasci i cene

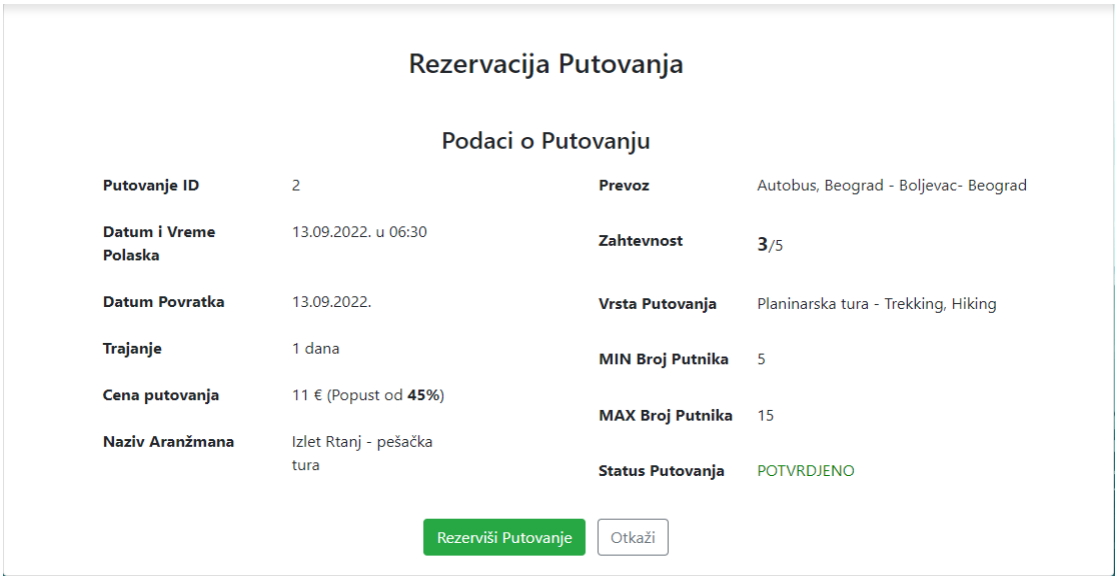
Osnovni scenario:

1. Korisnik poziva sistem da rezerviše odabrani polazak aranžmana (APSO)

Opis akcije: Korisnik pritiska dugme 

, čime poziva sistem da pokrene sistemsku operaciju *rezervišiPolazakAranžmana(polazakId)*.

2. Sistem obrađuje zahtev za rezervacijom odabranog polaska aranžmana (SO)
3. Sistem prikazuje detalje rezervacije odabranog polaska aranžmana (IA)



Slika 68. Projektovanje korisničkog interfejsa SK9 – Detalji putovanja odabranog za rezervaciju

4. Korisnik pregledava detalje svoje rezervacije (ANSO)
5. Korisnik potvrđuje sistemu rezervaciju polaska (APSO)

Opis akcije: Korisnik nakon pregleda detalja rezervacije pritiska dugme 

, čime potvrđuje svoju rezervaciju i nastavak sistemske operacije *rezervišiPolazakAranžmana(polazakId)*.

6. Sistem kreira novu rezervaciju polaska aranžmana (SO)

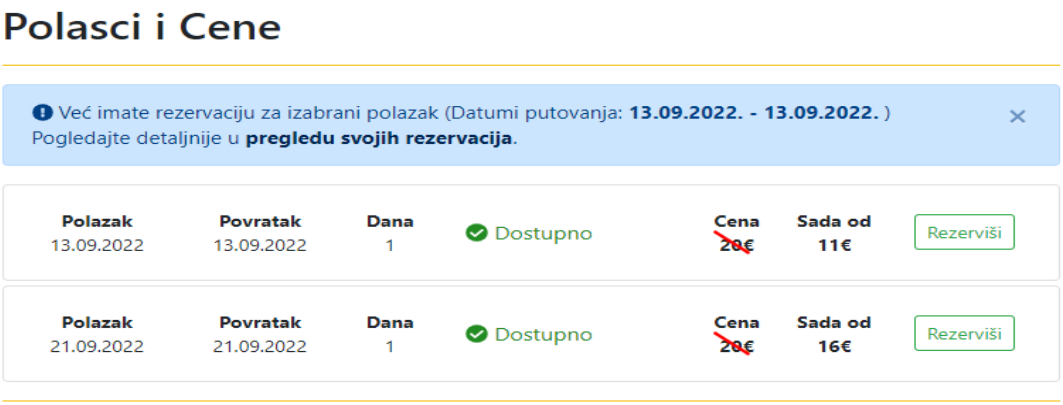
7. Sistem prikazuje korisniku potvrdu rezervacije polaska aranžmana (IA)

Alternativni scenario:

2.1. Sistem je utvrdio da korisnik ima već rezervaciju za odabrani polazak aranžmana, prikazuje odgovarajuću poruku korisniku i prekida izvršavanje scenarija (IA)

Opis akcije: Korisnik pritiska dugme 

, nakon čega sistem proverava da li postoji rezervacija ulogovanog korisnika za izabrano putovanje. Nakon što utvrdi da postoji, prekida izvršavanje sistemske operacije *rezervišiPolazakAranžmana(polazakId)* i prikazuje odgovarajuću poruku koja se može videti na slici 69.



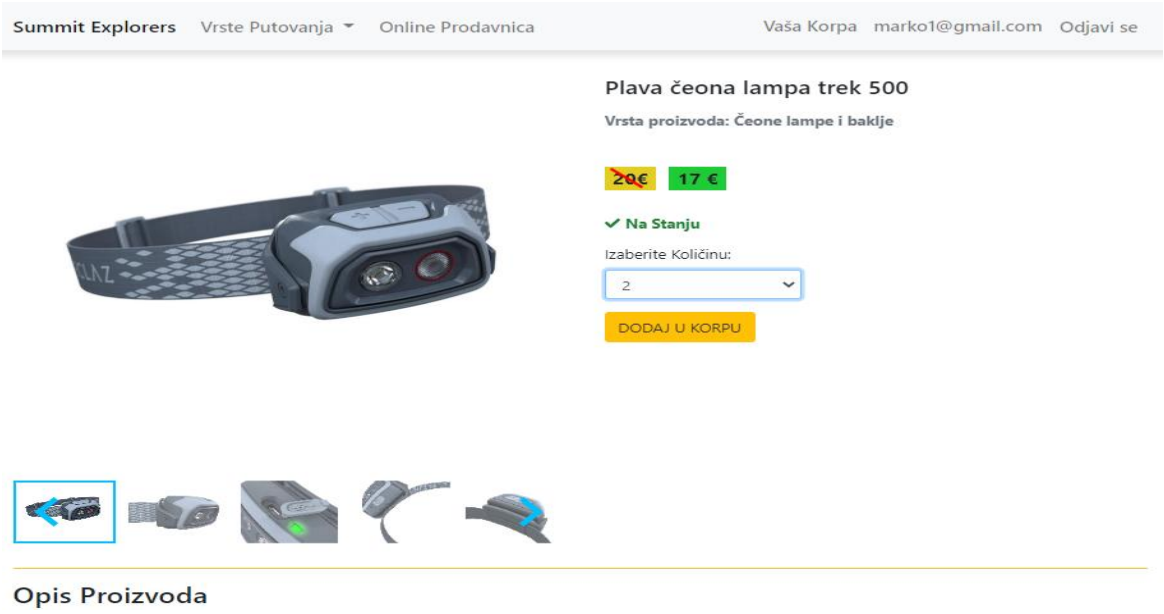
Slika 69. Projektovanje korisničkog interfejsa SK9 – Već postojeća rezervacija za izabrani polazak

SK13: Dodavanje proizvoda u korisničku korpu

Preduslovi: Sistem je aktivan, korisnik je prijavljen i prikazani su detalji odabranog proizvoda iz prodavnice (slika 70)

Osnovni scenario:

1. Korisnik bira količinu odabranog proizvoda (APUSO)



Slika 70. Projektovanje korisničkog interfejsa SK13 – Pregled detalja proizvoda, odabir količine

2. Korisnik poziva sistem da ubaci odabrani proizvod u korpu u biranoj količini (APSO)

Opis akcije: Korisnik pritiska dugme DODAJ U KORPU, čime poziva sistem da pokrene sistemsku operaciju *dodajProizvodUKorisničkuKorpu(korisnikId, proizvodId, izabranaKolicina)*.

3. Sistem pokreće proces ubacivanja proizvoda u korpu u odabranoj količini (SO)

4. Sistem prikazuje korisniku poruku o uspešnosti dodavanja proizvoda u korisničku korpu (IA)

Alternativna scenarija:

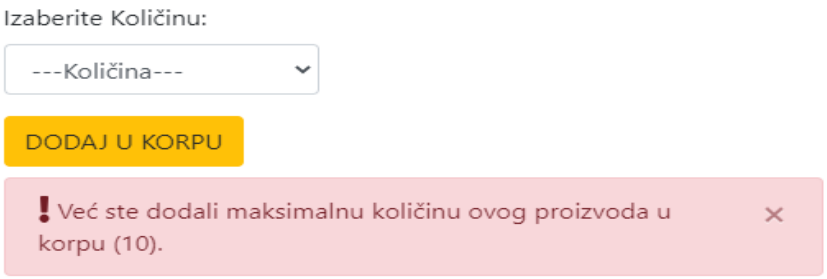
3.1. Sistem utvrđuje da je korisnik već dodao odabrani proizvod u korisničku korpu i da sa odabranom količinom prelazi limit količine proizvoda koju može da stavi u korpu, prikazuje odgovarajuću poruku korisniku (IA)

Slika 71. Projektovanje korisničkog interfejsa SK13 – Alternativni scenario 3.1.

Opis akcije: Korisnik je odabrao željenu količinu 10, pritiska dugme DODAJ U KORPU,

čime poziva sistem da pokrene sistemsku operaciju *dodajProizvodUKorisničkuKorpu(korisnikId, proizvodId, izabranaKolicina)*. Međutim, u korisničku korpu se dodaje samo količina 8, odnosno količina do limita količine jednog proizvoda koji može da stane u korisničku korpu.

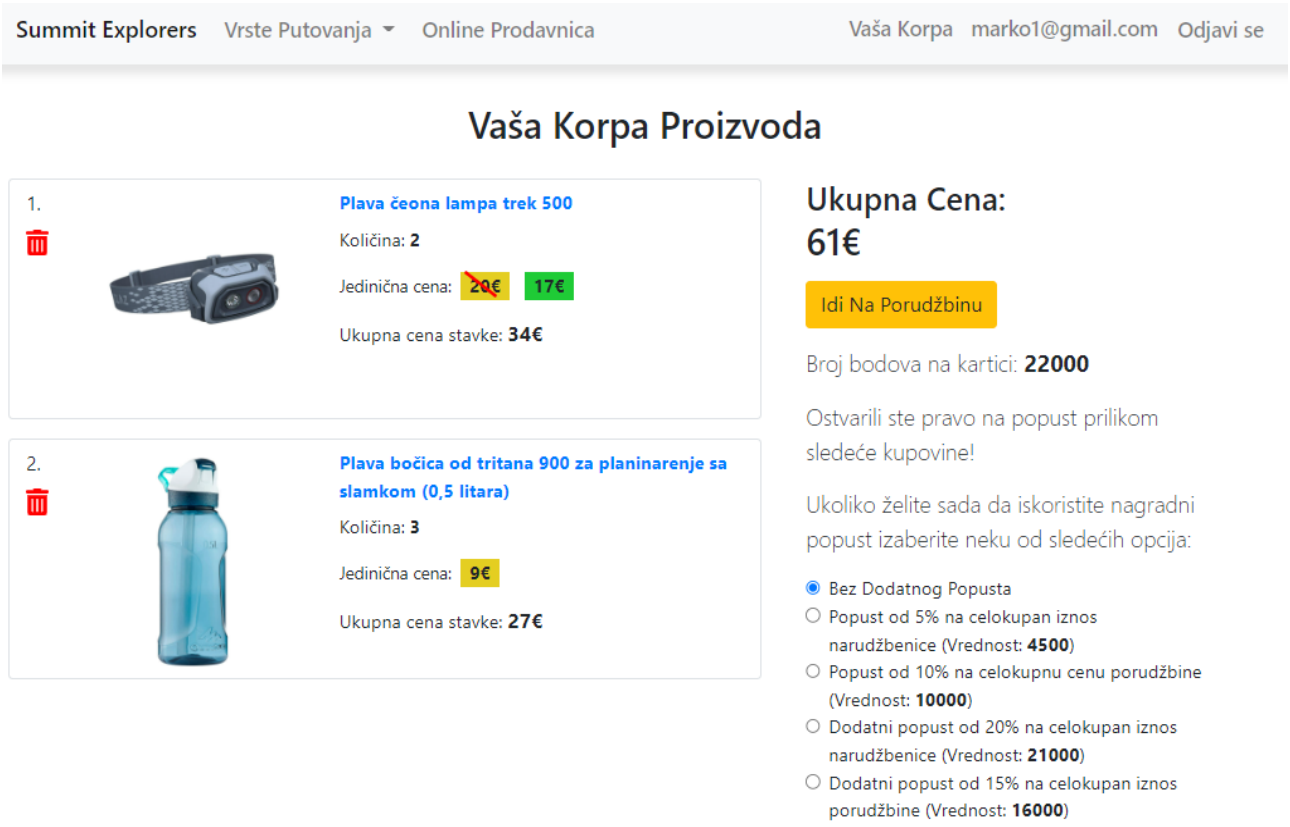
3.2. Sistem utvrđuje da je korisnik već dodao odabrani proizvod u korisničku korpu i da je već dostigao limit količine proizvoda koju može da stavi u korpu, prikazuje korisniku odgovarajuću poruku (IA)



Slika 72. Projektovanje korisničkog interfejsa SK13 – Pređen limit količine proizvoda u korpi

SK15: Poručivanje proizvoda

Preduslovi: Sistem je aktivan, korisnik je prijavljen i prikazana je korisnička korpa sa njenim stavkama



Slika 73. Projektovanje korisničkog interfejsa SK15 – Korisnička korpa

Osnovni scenario:

1. Korisnik bira jednu od ponuđenih opcija za popust prilikom naručivanja, u skladu sa bodovima na članskoj kartici (APSO)

Opis akcije: Odabirom neke od opcija ponuđenih popusta u grupisanoj listi grafičkog kontrolnog elementa RadioButton (slika 73, donji desni ugao), korisnik poziva sistem da pokrene sistemsku operaciju *izračunajNovuCenuPorudžbine(clanKartPopustId).*

2. Sistem izračunava novu cenu moguće porudžbine (SO)

3. Sistem prikazuje korisniku novu cenu moguće porudžbine (IA)

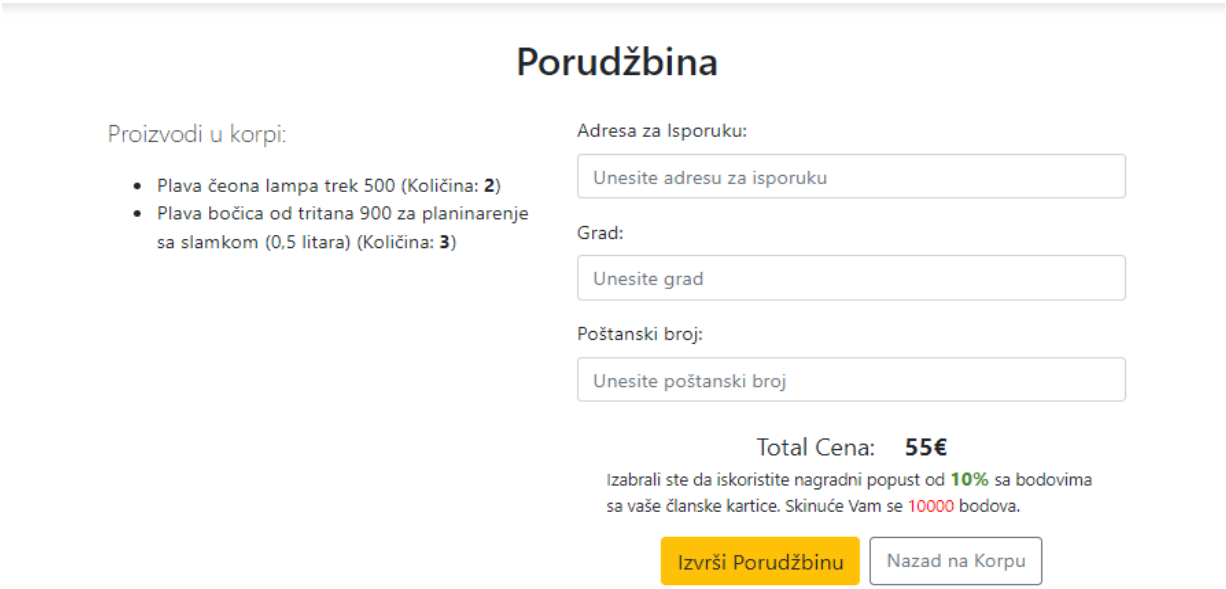
4. Korisnik potvrđuje sistemu poručivanje proizvoda iz korisničke korpe (APSO)

Opis akcije: Korisnik pritiska dugme 

, čime poziva sistem da pokrene sistemsku operaciju *prikažiFormuPodatakaODostavi(listaStavkiKorpe, korisnikId)*.

5. Sistem proverava dostupnost proizvoda nakon potvrde korisnika o poručivanju (SO)

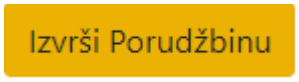
6. Sistem prikazuje korisniku formu za unos podataka o dostavi (adresa, grad, poštanski broj) (IA)



Slika 74. Projektovanje korisničkog interfejsa SK15 – Forma za unos podataka dostave

7. Korisnik unosi podatke o dostavi (APUSO)

8. Korisnik poziva sistem da obradi podatke o dostavi i izvrši porudžbinu (APSO)

Opis akcije: Korisnik pritiska dugme 

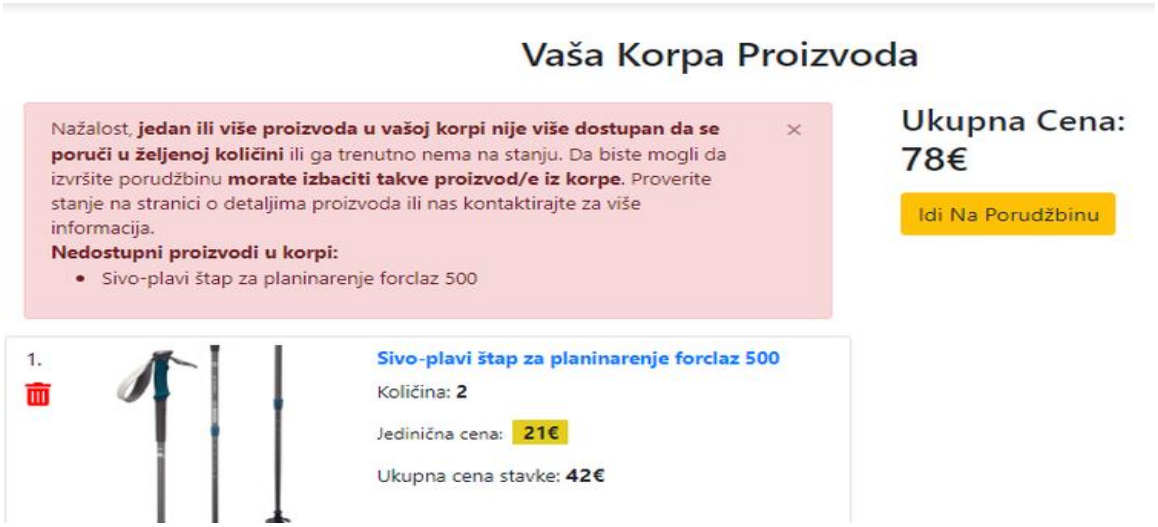
, čime poziva sistem da pokrene sistemsku operaciju *napraviNovuNarudžbenicu(listaStavkiKorpe, korisnikId)*.

9. Sistem evidentira novu narudžbenicu za korisnika (SO)

10. Sistem prikazuje korisniku poruku o uspešno izvršenoj porudžbini (IA)

Alternativni scenario:

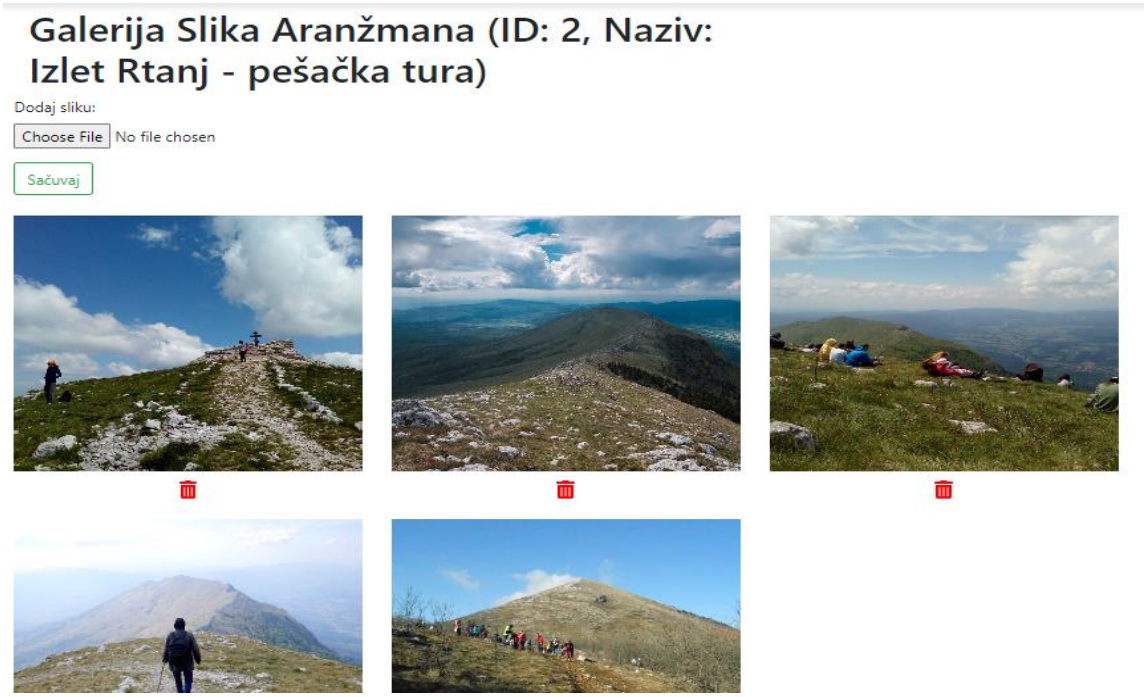
5.1. Sistem je utvrdio da jednog ili više proizvoda iz stavki korpe nema u dovoljnoj količini na stanju, prikazuje korisniku odgovarajuću poruku i prekida izvršavanje scenarija (IA)



Slika 75. Projektovanje korisničkog interfejsa SK15 – Proizvoda u stavki korpe nema u dovoljnoj količini

SK28: Brisanje slika aranžmana

Preduslovi: Sistem je aktivan, administrator je prijavljen i prikazana je stranica pregleda slika izabranog aranžmana (slika 76)



Slika 76. Projektovanje korisničkog interfejsa SK28 – Pregled slika aranžmana

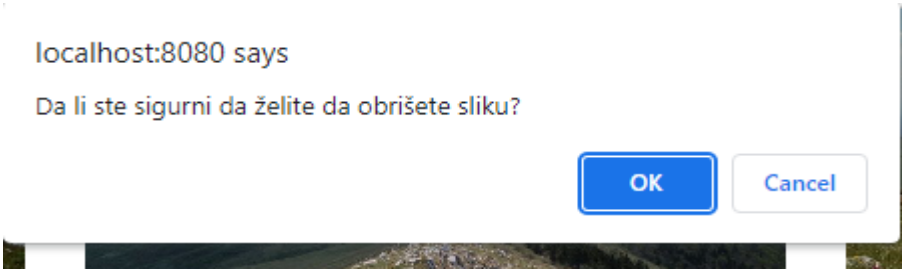
Osnovni scenario:

1. Administrator poziva sistem da započne proces brisanja izabrane slike aranžmana (APSO)

Opis akcije: Administrator pritiska dugme  ispod slike aranžmana koju je izabrao da obriše, čime poziva sistem da pokrene sistemsku operaciju *obrišiSlikuAranžmana(slikaAranžmanaId)*.

2. Sistem otvara dijalog za potvrdu brisanja izabrane slike (SO)

3. Sistem prikazuje dijalog za potvrdu brisanja izabrane slike (IA)



Slika 77. Projektovanje korisničkog interfejsa SK28 – Dijalog za potvrdu brisanja slike aranžmana

4. Administrator potvrđuje sistemu brisanje izabrane slike aranžmana (APSO)

Opis akcije: Administrator pritiska dugme 

, čime poziva sistem da nastavi sistemsku operaciju *obrišiSlikuAranžmana(slikaAranžmanaId).*

5. Sistem briše sliku iz baze podataka i iz fajl sistema aplikacije (SO)

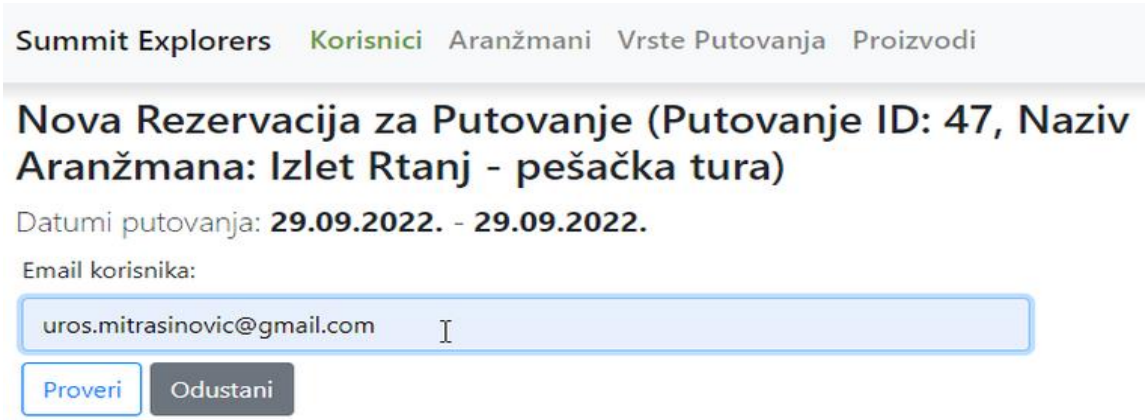
6. Sistem prikazuje poruku o uspešnosti brisanja izabrane slike aranžmana (IA)

SK40: Dodavanje nove rezervacije za polazak aranžmana

Preduslovi: Sistem je aktivan, administrator je prijavljen i prikazana je forma za dodavanje nove rezervacije u okviru izabranog polaska aranžmana (slika 78)

Osnovni scenario:

1. Administrator upisuje email korisnika za kojeg dodaje novu rezervaciju (APUSO)



Slika 78. Projektovanje korisničkog interfejsa SK40 – Forma nove rezervacije, unos email-a korisnika

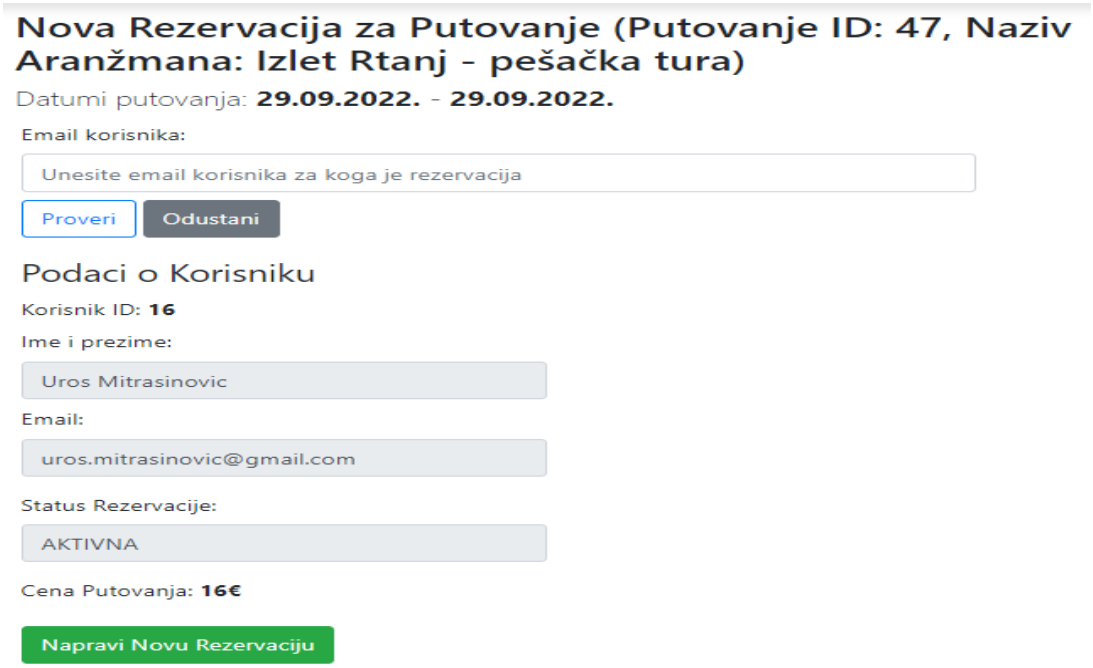
2. Administrator poziva sistem da proveri email korisnika (APSO)

Opis akcije: Administrator pritiska dugme 

, čime poziva sistem da pokrene sistemsku operaciju *proveriEmailKorisnikaZaRezervacijuPolaskaAranžmana(emailKor, polazakAranžmanaId)*.

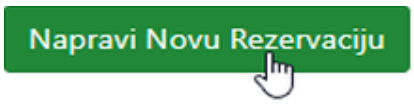
3. Sistem proverava email korisnika (SO)

4. Sistem prikazuje izmenjenu formu dodavanja nove rezervacije (IA)



Slika 79. Projektovanje korisničkog interfejsa SK40 – Izmenjena forma za dodavanje nove rezervacije

5. Administrator poziva sistem da evidentira novu rezervaciju za korisnika (APSO)

Opis akcije: Administrator pritiska dugme 

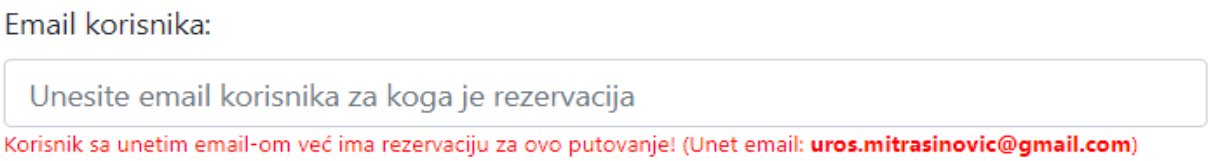
, čime poziva sistem da pokrene sistemsku operaciju *dodajNovuRezervacijuPolaskaAranžmana(polazakAranžmanaId)*.

6. Sistem evidentira novu rezervaciju polaska aranžmana (SO)

7. Sistem prikazuje poruku o uspešnosti dodavanja nove rezervacije polaska (IA)

Alternativni scenario:

3.2. Sistem je utvrdio da već postoji rezervacija u okviru izabranog polaska za korisnika sa unetim email-om, prikazuje administratoru odgovarajuću poruku (IA)



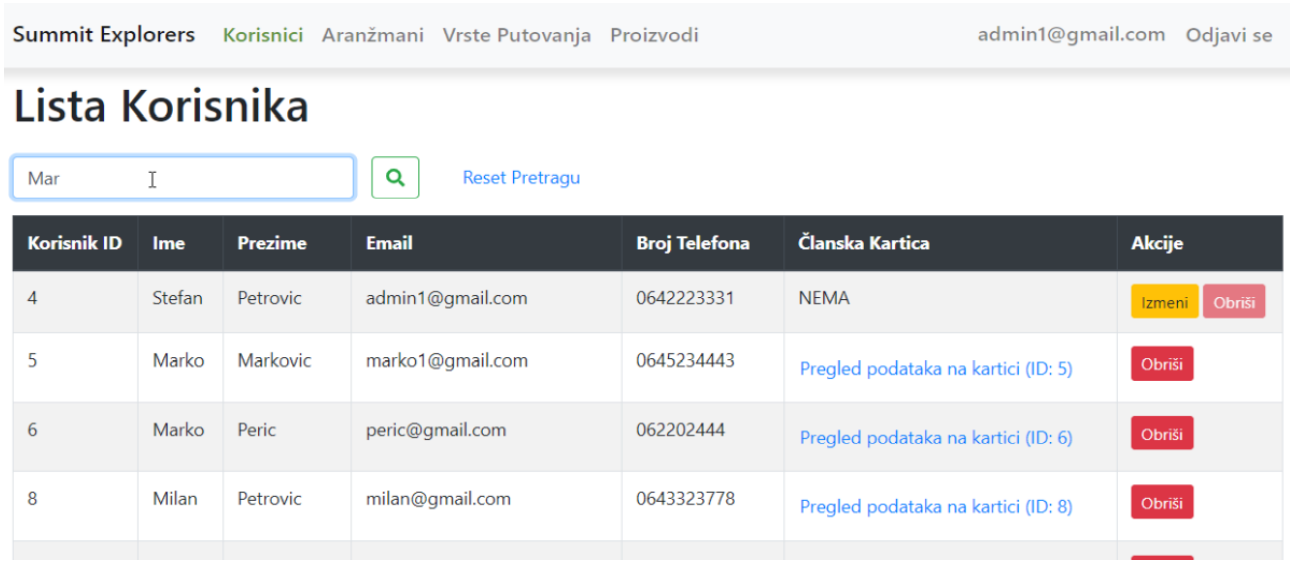
Slika 80. Projektovanje korisničkog interfejsa SK40 – Rezervacija za unetog korisnika već postoji

SK42: Pretraga korisnika

Preduslovi: Sistem je aktivan, administrator je prijavljen i prikazana je lista svih korisnika (početna stranica administratora, slika 81)

Osnovni scenario:

1. Administrator upisuje neku od ključnih reči za pretragu korisnika (ime, prezime ili email) (APUSO)



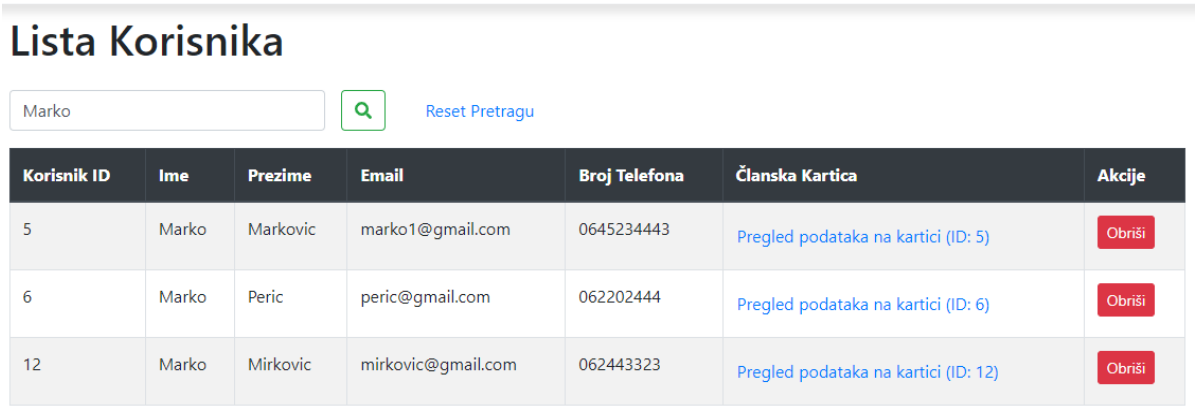
Slika 81. Projektovanje korisničkog interfejsa SK42 – Početna stranica administratora, unos ključne reči za pretragu

2. Administrator poziva sistem da pretraži korisnike po ključnoj reči (APSO)

Opis akcije: Administrator pritiska dugme  , čime poziva sistem da pokrene sistemsku operaciju *pretragaKorisnika(ključnaReč)*.

3. Sistem pretražuje korisnike (SO)

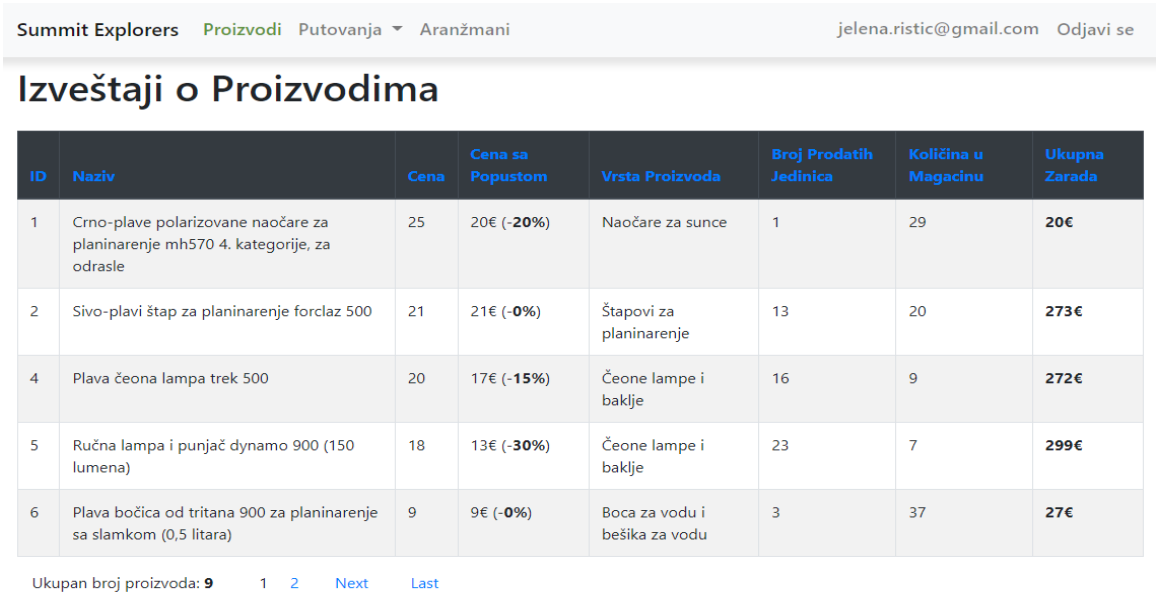
4. Sistem prikazuje jednog ili više korisnika vezanih za upisane ključne reči pretrage (IA)



Slika 82. Projektovanje korisničkog interfejsa SK42 – Rezultat pretrage

SK48: Sortiranje tabele izveštaja prema odabranim parametrima

Preduslovi: Sistem je aktivan, menadžer je prijavljen i prikazana je neka od tabele izveštaja poslovanja planinarskog kluba (slika 83)



Slika 83. Projektovanje korisničkog interfejsa SK48 – Početna stranica menadžera, pregled izveštaja o prodaji proizvoda

Osnovni scenario:

1. Menadžer pregledava i bira neki od parametara (kolona tabele), prema kojem želi da sortira tabelu izveštaja (ANSO)

Opis akcije: U pregledu tabele izveštaja, na primer prodaje proizvoda (slika 83), korisnik bira kolonu tabele po kojoj želi da je sortira, na primer Cena.

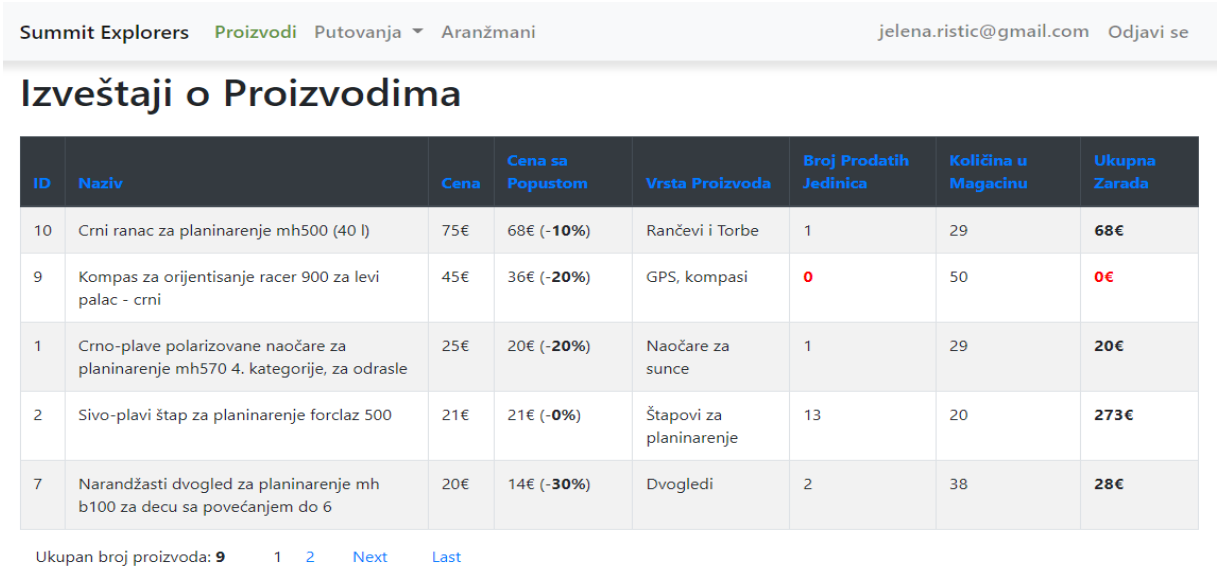
2. Menadžer poziva sistem da sortira tabelu izveštaja prema odabranom parametru (APSO)

Opis akcije: Menadžer pritiska na naziv kolone koju je izabrao 

, čime poziva sistem da pokrene sistemsku operaciju *sortirajTabeluIzveštaja(sortParam, sortDirection)*.

3. Sistem sortira tabelu izveštaja prema odabranom parametru (SO)

4. Sistem prikazuje sortiranu tabelu izveštaja prema odabranom parametru (IA)



Slika 84. Projektovanje korisničkog interfejsa SK48 – Sortirana tabela izveštaja proizvoda po ceni opadajuće

2.4. FAZA IMPLEMENTACIJE

Faza implementacije Larmanove metode razvoja softvera je četvrta faza razvoja, u kojoj se vrši kodiranje sistema na osnovu podataka iz prethodne tri faze, primenom određenih tehnologija. Za izradu ove Veb aplikacije korišćene su sledeće tehnologije:

Sa serverske strane:

- Spring Framework i u okviru njega niz njegovih modula koji pripadaju spring core sloju (spring-core, spring-beans, …), data access sloju (sloju podataka) kao što je JDBC (eng. Java Database Connectivity), Spring Data JPA, Web sloju itd.
- Hibernate ORM (objektno-relacioni maper)
- Thymeleaf template engine tehnologija – Java template engine (šablonski mehanizam) tehnologija za pisanje prirodnih (HTML) šablona stranica
- Maven alat za upravljanje projektom i njegovom konfiguracijom

Sa klijentske strane:

- HTML, CSS, JS, JQuery, Bootstrap biblioteka klasa (verzija 4.5.)

Kao razvojno okruženje korišćen je IntelliJ IDEA Ultimate.

Kao sistem za upravljanje relacionom bazom podataka korišćen je MySQL.

Ova veb aplikacija podignuta je uz pomoć Springovog projekta Spring Boot, koji u sebi već sadrži početnu konfiguraciju i osnovne module.

Spring Boot olakšava kreiranje samostalnih (eng. stand-alone), spremnih za produkciju (eng. production-grade) aplikacija zasnovanih na Springu, koje možemo odmah da pokrenemo [10].

Za autentifikaciju i autorizaciju korišćen je Spring Security radni okvir.

Rezultat faze implementacije je izrađena Veb aplikacija koja je prilog ovog rada.

ZAKLJUČAK

Ova veb aplikacija, kada gledamo striktno sa klijentske (web browser) strane, je izgrađena sa IT podrškom koja omogućava da se proces rezervacija putovanja i kupovine u prodavnici obavlja potpuno onlajn, i u samo nekoliko koraka. Takođe, olakšan je pregled rezervacija putovanja i pregled korisnikovih porudžbina u prodavnici, kojima lako klijent može da pristupi na svom nalogu. Što se tiče administrativne uloge, korisnički interfejs (user interface) je jednostavan, intuitivan i lak za upotrebu, i samim tim je kompletna kontrola i uvid nad podacima znatno olakšan. Menadžer, kao poslednji tip korisnika ove aplikacije, ima jasan uvid u poslovanje planinarskog kluba kroz tabelaran prikaz i mogućnost sortiranja sadržaja.

Spring radni okvir kao tehnologiju izrade za veb aplikaciju ovog završnog rada izabrao sam zbog njene velike popularnosti i traženosti, ali i moje lične zainteresovanosti za nju. Cilj mi je bio, između ostalog, i da proširim svoje znanje o Java tehnologijama, ali i kompletnom procesu razvoja softvera. Pošto se radi o tehnologiji koja je meni do pre početka rada na ovom završnom radu bila skoro potpuno nepoznata, cilj mi je bio i da unapredim svoje istraživačke sposobnosti i testiram svoju volju i snalaženje u novom tehnološkom okruženju.

Spring radni okvir nudi veliki nivo apstrakcije nad svim slojevima razvoja aplikacije, čime razvoj postaje brži i produktivnost veća. Sadrži veliki broj gotovih rešenja, smanjujući potrebu za pisanjem upravljačkog koda, omogućavaju developerima da se fokusiraju više na pisanje poslovne logike. Nudi veliku podršku integracije sa različitim tehnologijama, zbog svoje modularne arhitekture, čineći ga izuzetno kompatibilnim u različitim okruženjima. Performansi ovog radnog okvira su na zavidnom nivou, kao i bezbednost koja se implementira sa lakoćom. Takođe, inženjeri aktivno rade na njegovom unapređenju, pružajući garanciju još kvalitetnijih, efikasnijih i pouzdanijih rešenja u budućnosti. Svi ovi parametri su igrali veliku ulogu u odabiru ove tehnologije za razvoj veb aplikacije.

Aplikacija koju sam izgradio je naravno studentski primer, i postoji dosta prostora za njeno unapređenje i proširenje. Jedna od glavnih stvari bi bila uvođenje sistema za online plaćanje karticom, odnosno transakcijski prenos sredstava, tako što bi se aplikacija povezala sa spoljnim bankarskim informacionim sistemom. Dalje, mogla bi da se uvede komunikacija preko email-a između sistema i korisnika, kao i zaposlenih u planinarskom klubu koji koriste sistem. Slanje raznih promocija usled, na primer, popusta na određene proizvode u prodavnici, slanje potvrda rezervacija putovanja preko mail-a, kao i slanje mail-a za konfirmaciju na email unet prilikom registracije, što bi predstavljalo dodatan vid zaštite.

LITERATURA

[1] Anđelić, S. *WPF I ASP.NET Framework - projektovanje i implementacija softvera*. Beograd: ITS, 2016.

[2] Enríquez, R., Salazar, A. *Software Architecture with Spring 5. 0 : Design and Architect Highly Scalable, Robust, and High-Performance Java Applications.* Birmingham: Packt Publishing, Limited; 2018.

[3] Introduction to Spring Framework. Available at: https://docs.spring.io/spring-framework/docs/4.0.x/spring-framework-reference/html/overview.html (pristupano 13.09.2022)

[4] Introduction to the POM. The Apache Software Foundation. Available at: https://maven.apache.org/guides/introduction/introduction-to-the-pom.html (pristupano 13.09.2022)

[5] Introduction to the Spring IoC Container and Beans. Available at: https://docs.spring.io/spring-framework/docs/current/reference/html/core.html#beans-introduction (pristupano 13.09.2022)

[6] John, T., *Hands-On Spring Security 5 for Reactive Applications : Learn Effective Ways to Secure Your Applications with Spring and Spring WebFlux.* Birmingham: Packt Publishing, Limited; 2018.

[7] Karanam, RR. *Naučite Spring 5.* Beograd: Kompjuterska biblioteka, 2017.

[8] Mehta, C., Shah, S., Shah, P., Goswami, P., Radadiya, D. *Hands-On High Performance with Spring 5 : Techniques for Scaling and Optimizing Spring and Spring Boot Applications.* Birmingham: Packt Publishing, Limited; 2018.

[9] Patel, N., Patel, K. *Java 9 Dependency Injection : Write Loosely Coupled Code with Spring 5 and Guice.* Birmingham: Packt Publishing, Limited; 2018.

[10] Spring Boot. Available at: https://spring.io/projects/spring-boot (pristupano 13.09.2022)

[11] Spring Data. Available at: https://spring.io/projects/spring-data (pristupano 13.09.2022)

[12] Spring Security. Available at: https://spring.io/projects/spring-security (pristupano 13.09.2022)

[13] The IoC Container. Available at: https://docs.spring.io/spring-framework/docs/4.0.x/spring-framework-reference/html/beans.html (pristupano 13.09.2022)

[14] The Security Filter Chain. Available at: https://docs.spring.io/spring-security/site/docs/3.0.x/reference/security-filter-chain.html (pristupano 14.09.2022)