

Programsko inženjerstvo ak.god 2024./2025

SkyDancers

Grupa: TG12.4

Tim:

Ime člana	Područje rada
Barbara Glavina	Frontend
Katarina Bubalo	Dokumentacija
Fani Jurak	Dokumentacija
Luka Malešević	Frontend
Antonio Šimić	Frontend
Leonardo Klišanić	Dokumentacija
Mario Vukoja	Full stack

Nastavnik: Vlado Sruk

1. Opis projektnog zadatka

Opis projektnog zadatka

SkyDancers je platforma za povezivanje plesača i direktora u plesnoj industriji. Dizajnirana kako bi olakšala proces audicija, SkyDancers omogućava plesnim direktorima da jednostavno pretražuju i pronađe talentirane plesače za raznovrsne poslove i projekte. S druge strane, plesači imaju pristup sveobuhvatnom katalogu audicija i projekata, na kojem mogu pregledavati sve otvorene mogućnosti i prijaviti se na one koje im odgovaraju. Zbog nedostatka ovakvih aplikacija u regiji, SkyDancers bi mogao biti centralno mjesto koje bi povezivalo plesače i plesne direktore i pomoglo popularnosti i razvoju plesne industrije.



Korisnici u aplikaciji

Kao što je navedeno, korisnici u aplikaciji su:

- Plesači
 - Imaju svoje javne profile na koji mogu postaviti vlastiti portfolio koji uključuje slike ili snimke svojih nastupa. Na profilu, plesači također mogu postaviti oznake vrsta plesa u kojima su iskusni (npr. tango, valcer, jazz, breakdance...), te postaviti osnovne informacije: lokacija (grad, država), dob i spol.
 - Mogu pretraživati objavljene audicije po filtrima: vrijeme, plaća, lokacija, vrsta plesa.
 - Mogu se prijaviti na odabranu audiciju.
- Plesni direktori
 - Na platformi objavljaju audicije.
 - Imaju javne profile s portfolijom snimki i slika svojih projekata i poslova.
 - Mogu koristiti tražilicu za pretraživanje plesača korištenjem različitih filtera poput dobi, spola i vrsti plesa.
- Administratori
 - Postavljaju cijenu članstva, te upravljaju korisnicima.

Korištenje platforme

Za korištenje platforme potrebno je registrirati se kao plesač ili plesni direktor. Plesači koriste aplikaciju besplatno, dok plesni direktori plaćaju godišnju članarinu za korištenje platforme što mogu napraviti plaćanjem kreditnim karticama korištenjem vanjskog servisa.

Korištenje SkyDancers platforme je jednostavno. Plesači započinju tako što kreiraju svoj profil, gdje mogu unijeti sve relevantne informacije o sebi – uključujući plesni stil, razinu iskustva, specifične vještine, videozapise i slike koji prikazuju njihov talent i audicije na kojima su nastupali. Platforma također omogućava plesačima da navedu svoje preferencije za projekte, područja i lokacije na kojima žele raditi i vrste angažmana koji ih najviše zanimaju. Kada profil bude postavljen, plesači mogu pregledavati otvorene audicije i projekte pomoću filtera, kao što su stil plesa, lokacija ili vrsta angažmana. Plesači se na audicije prijavljuju direktno putem platforme, čime njihova prijava odmah postaje dostupna odgovarajućim plesnim direktorima. Korištenje platforme za plesače je besplatno.

Plesni direktori započinju korištenje platforme kreiranjem projektnog opisa za angažman ili audiciju koju nude. U okviru projektnog opisa, direktori mogu specificirati detalje poput zahtijevanih plesnih stilova, tehničkih vještina, nivoa iskustva, trajanja projekta i lokacije. Nakon što se oglas objavi, plesači se mogu prijavljivati, a direktor ima pregledan prikaz svih prijava i profila plesača koji su pokazali interes. Direktori također imaju mogućnost korištenja dodatnih filtera i pretraživača kako bi pronašli plesače koji najviše odgovaraju specifičnim potrebama projekta. Korištenje platforme za plesne direktore bi se plaćalo godišnjom članarinom.

Komunikacija između korisnika

U platformi je omogućena opcija komunikacije između svih korisnika. Plesači se mogu javljati drugim plesačima putem razmjene poruka (čavrjanjem). Plesači i direktori, mogu putem platforme komunicirati i dogоворити detalje audicija, eventualnih proba ili daljnjih koraka u procesu zapošljavanja. SkyDancers time olakšava i ubrzava sve faze od prvotne prijave do konačnog nastupa, čineći cjelokupan proces jednostavnijim i bržim za sve uključene. Plesni direktori mogu također plesačima direktno slati ponude za poslove (neovisno o audicijama).

Opseg projekta

- Osnovna funkcionalnost platforme
 - Registracija i prijava korisnika
 - Korisnici se mogu registrirati koristeći google račun.
 - Autentifikacija i sigurnost korisničkih računa.
 - Profili
 - Plesači imaju mogućnost unosa osobnih podataka, iskustva, plesnih stilova, vještina, video i foto sadržaja.
 - Plesni direktori imaju mogućnost unosa osobnih podataka, te projekata i audicija na kojima su sudjelovali.
 - Administratori imaju mogućnost upravljanja profilima primjerice brisanje ili sakrivanje.
 - Kreiranje audicija
 - Plesni direktori mogu objavljivati nove audicije ili projekte s detaljnim opisima i preferencama.
 - Pretraživanje audicija
 - Plesači mogu pretraživati audicije koristeći različite filtere poput stila plesa, razine iskustva, lokacije.
 - Komunikacija
 - Svaki korisnik imaj mogućnost komuniciranja sa svim ostalim korisnicima.
- Napredne funkcionalnosti platforme
 - Servis za plaćanje
 - Platforma će integrirati vanjski servis za plaćanje.
 - Servis za prijavu i registraciju
 - Platforma će integrirati vanjski servis za registraciju i prijavu korisnika.

Slična rješenja

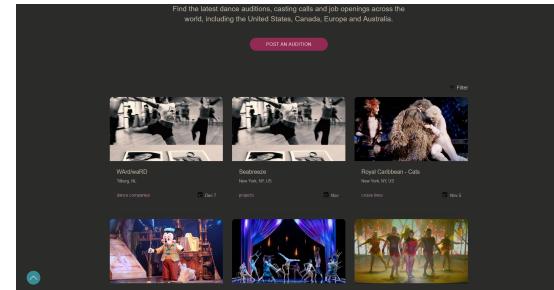
Jedno slično rješenje je aplikacija DancePlug u kojoj korisnici mogu isto objavljivati audicije i prijavljivati se na njih. Sličnosti:

- DancePlug aplikacije i naše su to što korisnici mogu objavljivati audicije i prijavljivati se na njih.

- Aplikacija ima mogućnost pretplate.

Razlike:

- Bilo koji neregistrirani korisnik može objaviti audiciju sa svim potrebnim podacima.
- Aplikacija nudi online tečajeve plesa.
- Korisnici koji se hoće prijaviti imaju sedam dana besplatnog korištenja aplikacije, a zatim moraju odabratи neku od opcija plaćanja.
- Aplikacija nema implementiranu komunikaciju između korisnika, nego se korisnici javljaju na kontakt ostavljen u opisu audicije.
- Aplikacija ima novosti i članke.



Moguće nadogradnje

- "Native" aplikacija
 - Platforma bi se mogla postaviti i na mobilne uređaje, te bi bila još dostupnija korisnicima.
- Edukativni sadržaj
 - Platforma bi se mogla proširiti edukativni sadržajem na kojem bi se moglo učiti plesati. Jedan dio sadržaja bi mogao biti besplatan da korisnici vide sviđa li im se sadržaj, a onda bi dio sadržaja bio na plaćanje ako se korisnicima sadržaj svidi.
- Mentorstvo i privatni treninzi
 - Aplikacija bi mogla dodati još jednu vrstu korisnika "mentor" ili proširiti mogućnost plesača. Mentor bi mogao organizirati privatne treninge uživo ili online. Mentor bi također mogao davati savjete za razvoj plesača.

2. Analiza zahtjeva

Funkcionalni zahtjevi

ID zahtjeva	Opis	Prioritet	Izvor	Kriteriji prihvaćanja
-------------	------	-----------	-------	-----------------------

ID zahtjeva	Opis	Prioritet	Izvor	Kriteriji prihvatanja
F-1	Aplikacija omogućuje registraciju korisnika kao plesača ili plesnih direktora	Visok	Zahtjev dionika	Korisnik se može registrirati e-mailom, primiti e-mail za potvrdu i uspješno se prijaviti.
F-2	Prilikom registracije, plesači se registriraju besplatno, a plesni direktori uz plaćanje godišnje članarine nakon ispunjavanja osobnih podataka	Srednji	Zahtjev dionika	Plesni direktori mogu preko servisa za plaćanje platiti članarinu.
F-2.1	Aplikacija omogućuje plaćanje članarine plesnim direktorima putem vanjskog servisa za plaćanje	Srednji	Postojeći sustav	Plesni direktori mogu dobiti potvrdu o plaćanju.
F-3	Aplikacija omogućuje plesnim direktorima objavljivanje audicija, s detaljima poput vremena, lokacije, opisa posla, vrsta traženih plesača, broja otvorenih pozicija i plaće	Visok	Dokument zahtjeva	Plesni direktori mogu ispuniti formu sa svim navedenim uvjetima.
F-4	Plesni direktori moraju moći pretraživati plesače koristeći filtre poput dobi, spola, vrste plesa.	Visok	Povratne informacije korisnika	Plesni direktori pomoći filtera vide odgovarajuće plesače koji zadovoljavaju uvjete.
F-5	Plesači moraju moći pretraživati dostupne audicije pomoći filtera kao što su vrijeme, plaća, lokacija i vrsta plesa.	Srednji	Dokument zahtjeva	Plesači pomoći filtera vide odgovarajuće audicije koje zadovoljavaju uvjete.

ID zahtjeva	Opis	Prioritet	Izvor	Kriteriji prihvaćanja
F-6	Plesači moraju moći prijaviti se na odabране audicije izravno putem aplikacije.	Visok	Dokument zahtjeva	Plesači mogu pogledati jesu li uspešno prijavljeni na audiciju.
F-7	Plesači moraju moći kreirati javni profil s osnovnim podacima (ime, prezime, lokacija, dob, spol, vještine) te portfolio koji uključuje slike ili snimke nastupa	Visok	Dokument zahtjeva	Plesači mogu vidjeti podatke na svome profilu.
F-8	Plesni direktori moraju moći kreirati javni profil s osnovnim podacima (ime, prezime, adresa, kontakt informacije) te portfolio svojih projekata i poslova.	Visok	Dokument zahtjeva	Plesni direktori mogu vidjeti podatke na svome profilu.
F-9	Plesni direktori moraju imati mogućnost slanja izravnih ponuda za poslove plesačima, neovisno o audicijama.	Srednji	Dokument zahtjeva	Plesači primaju ponudu na svojoj aplikaciji, dok plesni direktori vide potvrdu o primitku.
F-10	Aplikacija mora integrirati postojeći sustav za razmjenu poruka (npr. FreeChat) kako bi omogućila komunikaciju među korisnicima.	Visok	Dokument zahtjeva	Korisnici vide poruke koje su poslali ili koje su primili.
F-10.1	Aplikacija mora omogućiti plesačima razmjenu poruka (čavrjanje) za komunikaciju.	Visok	Zahtjev dionika	Plesači vide poruke koje su poslali ili koje su im poslane.
F-10.2	Aplikacija mora omogućiti plesačima javljanje plesnim direktorima preko poruka za više informacija o audiciji	Visok	Dokument zahtjeva	Plesači vide poruke koje su poslali.

ID zahtjeva	Opis	Prioritet	Izvor	Kriteriji prihvaćanja
F-11	Plesači moraju imati mogućnost postaviti svoj profil kao "neaktivan" na određeno razdoblje, tijekom kojeg drugi korisnici ne mogu komunicirati s njima.	Srednji	Dokument zahtjeva	Plesači primaju obavijest koja potvrđuje da neće više primati poruke od drugih korisnika.
F-12	Administratori sustava moraju imati mogućnost upravljanja korisničkim profilima plesača i plesnih direktora.	Visok	Dokument zahtjeva	Administratori mogu raditi određene akcije na korisničkim profilima (brisanje, uređivanje, ...)
F-13	Administratori moraju moći postaviti iznos godišnje članarine za plesne direktore.	Srednji	Dokument zahtjeva	Administratori dobivaju potvrdu o promjeni ili postavljanju članarine.
F-14	Svaka audicija mora jasno prikazivati rok prijave za plesače, koji mora biti vidljiv na stranici audicije.	Visok	Dokument zahtjeva	Korisnici koji kliknu na audiciju mogu vidjeti sve detalje o audiciji.
F-15	Plesni direktori moraju imati mogućnost pregledavanja statistika prijava na audicije, uključujući broj prijavljenih plesača, vrste plesa i status prijava.	Srednji	Dokument zahtjeva	Plesni direktori dobivaju vizualnu reprezentaciju statistika.
F-16	Aplikacija mora omogućiti slanje notifikacija plesačima o novim audicijama na temelju njihovih preferencija (filtr poput vrste plesa, lokacije).	Niski	Dokument zahtjeva	Plesači dobivaju periodične prijedloge o novim audicijama.

ID zahtjeva	Opis	Prioritet	Izvor	Kriteriji prihvaćanja
F-17	Aplikacija mora omogućiti korisnicima promjenu lozinke i oporavak lozinke putem e-mail adrese.	Visok	Povratne informacije korisnika	Korisnik na e-mail prima link za promjenu lozinke i može ju promjeniti.
F-18	Aplikacija mora omogućiti plesnim direktorima arhiviranje starih audicija i pregled arhiviranih podataka.	Srednji	Dokument zahtjeva	Plesni direktori mogu u povijesti vidjeti održane audicije i podatke.
F-19	Aplikacija mora omogućiti korisnicima upravljanje svojim profilom, uključujući promjenu osnovnih podataka, dodavanje portfolia i ažuriranje statusa (aktivan/neaktivran).	Visok	Dokument zahtjeva	Korisnik može promjeniti podatke o sebi.

Ostali zahtjevi

ID zahtjeva	Opis	Prioritet
NF-1	Aplikacija mora osigurati da su svi podaci korisnika zaštićeni.	Visok
NF-2	Aplikacija mora biti optimizirana kako bi podržala istovremeni rad par tisuća korisnika bez smanjenja performansi (čekanja na odgovor više od 1 sekunde).	Visok
NF-2.1	Aplikacija mora biti skalabilna kako bi podržala rast baze korisnika i broj audicija bez utjecaja na stabilnost sustava (preko 1000 korisnika).	Srednji
NF-3	Aplikacija mora biti u potpunosti responzivna i funkcionalna na različitim uređajima.	Visok

Dionici

1. Plesni direktori
2. Plesači
3. Administratori
4. Razvojni tim

Aktori i njihovi funkcionalni zahtjevi:

- Plesni direktor (sudionik) može:
 - registrirati se u aplikaciju (F-1)
 - platiti godišnju članarinu (F-2)
 - koristiti vanjski servis za plaćanje(F-2.1)
 - objavljivati audicije (F-3)
 - pretraživati plesače pomoću filtara (F-4)
 - kreirati javni profil s osnovnim podacima (F-8)
 - slati izravne ponude plesačima (F-9)
 - razmjenjivati poruke s plesačima (F-10.2)
 - pregledavati statistike prijava na audicije (F-15)
 - promijeniti lozinku (F-17)
 - arhiviranje i pregled starih audicija (F-18)
 - upravljati svojim profilom (F-19)
- Plesač (sudionik) može:
 - registrirati se u aplikaciju (F-1)
 - besplatno koristiti aplikaciju (F-2)
 - pretraživati audicije pomoću filtara (F-5)
 - prijaviti se na određene audicije (F-6)
 - kreirati javni profil s osnovnim podacima (F-7)
 - razmjenjivati poruke s ostalim korisnicima (F-10)
 - razmjenjivati poruke s drugim plesačima (F-10.1)
 - javiti se plesnim direktorima za informacije o audiciji (F-10.2)
 - postaviti profil kao neaktivn (F-11)
 - primati notifikacije o novim audicijama (F-16)
 - promijeniti lozinku (F-17)
 - upravljati svojim profilom (F-19)
- Administrator (inicijator) može:
 - upravljati korisničkim profilima (F-12)
 - postaviti iznos godišnje članarine plesnim direktorima (F-13)

3. Specifikacija zahtjeva sustava

Obrasci uporabe

Opis obrazaca uporabe

UC 1 - Registracija korisnika

- Glavni sudionik: Neprijavljeni korisnik
- Cilj: stvoriti korisnički račun
- Sudionici: Baza podataka

- Preduvjet: ako je odabrana izrada računa za plesnog direktora, plaćanje godišnje članarine
- Opis osnovnog tijeka:

1. Korisnik odabire opciju za registraciju
2. Odabir plesnog direktora ili plesača (F-1)
3. Korisnik unosi potrebne korisničke podatke (F-2, F-7, F-8)
4. Korisnik je uspješno registriran

- Opis mogućih odstupanja:

3. Odabir postojećeg korisničkog imena i/ili e-maila, unos korisničkog podatka u nedozvoljenom formatu ili unos neispravnog e-maila - sustav obavještava korisnika o neuspješnoj registraciji i vraća ga na stranicu za registraciju, gdje korisnik mijenja potrebne podatke ili odustaje od registracije
4. Odabran plesni direktor, a nije plaćena godišnja članarina - sustav obavještava korisnika o neuspješnoj registraciji i vraća ga na stranicu za plaćanje članarine

UC 2 - Prijava korisnika

- Glavni sudionik: Neprijavljeni korisnik
- Cilj: Prijava u sustav
- Sudionici: Baza podataka
- Preduvjet: Registracija korisnika
- Opis osnovnog tijeka:

1. Korisnik odabire opciju za prijavu u sustav
2. Korisnik unosi korisničko ime i lozinku
3. Korisnik se uspješno prijavljuje i dobiva pristup sustavu

- Opis mogućih odstupanja:

2. Neispravno korisničko ime i/ili lozinka - sustav obavještava korisnika o neuspješnoj prijavi, korisnik ponovo unosi podatke (ako je samo lozinka kriva moguć njen oporavak putem e-mail adrese) ili odustaje od prijave

UC 3 - Plaćanje godišnje članarine za plesne direktore

- Glavni sudionik: Neprijavljeni korisnik
- Cilj: Prijava u sustav
- Sudionici: Baza podataka, Vanjski servis za plaćanje
- Preduvjet: Odabrana opcija plesnog direktora i ispunjen obrazac za registraciju
- Opis osnovnog tijeka:

1. Korisnik preusmjeren na stranicu vanjskog servisa za plaćanje (F-2.1)
2. Korisnik uspješno platio

- Opis mogućih odstupanja:

2. Neispravna uplata - vraćanje na stranicu vanjskog servisa ili odustajanje od uplate

UC 4 - Objava audicije

- Glavni sudionik: Plesni direktor
- Cilj: Objava nove audicije s detaljima poput vremena, lokacije, opisa posla, vrsta traženih plesača, broja otvorenih pozicija i plaće
- Sudionici: Baza podataka
- Preduvjet: -
- Opis osnovnog tijeka:

1. Plesni direktor odabire opciju za objavu nove audicije
2. Plesni direktor unosi sve podatke o audiciji u obrascu (F-3, F-14)
3. Uspješno objavljena audicija

- Opis mogućih odstupanja:

2. Neispravno uneseni podaci/nije unesen rok prijave za plesače - ponovno unošenje podataka ili odustajanje

UC 5 - Pretraživanje plesača

- Glavni sudionik: Plesni direktor
- Cilj: Pretraživanje plesača koristeći filtre poput dobi, spola, vrste plesa
- Sudionici: Baza podataka
- Preduvjet: -
- Opis osnovnog tijeka:

1. Plesni direktor odabire opciju za pretraživanje plesača
2. Plesni direktor unosi filtre prema kojima želi pretraživati (F-4)
3. Sustav mu vraća listu plesača koji zadovoljavaju kriterije

UC 6 - Pretraživanje dostupnih audicija

- Glavni sudionik: Plesač
- Cilj: Pretraživanje dostupnih audicija koristeći filtre poput vrijeme, plaće, lokacija i vrsta plesa
- Sudionici: Baza podataka
- Preduvjet: -
- Opis osnovnog tijeka:

1. Plesač odabire opciju za pretraživanje dostupnih audicija
2. Plesač unosi filtre prema kojima želi pretraživati
3. Sustav mu vraća listu dostupnih audicija koji zadovoljavaju kriterije

UC 7 - Izravna prijava na audicije

- Glavni sudionik: Plesač
- Cilj: Prijava na odabrane audicije izravno putem aplikacije
- Sudionici: Baza podataka
- Preduvjet: -
- Opis osnovnog tijeka:

1. Plesač odabire audiciju
2. Plesač odabire opciju za prijavu na odabranu audiciju (F-6)

3. Sustav mu vraća poruku o uspješnoj prijavi

UC 8 - Slanje poruka

- Glavni sudionik: Plesač/Plesni direktor
- Cilj: Poslati poruku pomoću postojećeg sustava za razmjenu poruka
- Sudionici: Baza podataka, FreeChat
- Preduvjet: -
- Opis osnovnog tijeka:

1. Korisnik odabire osobu kojoj želi poslati poruku
2. Korisnik napiše sadržaj poruke te stisne gumb za slanje
3. Sustav tu poruku proslijeđuje vanjskom sustavu za razmjenu poruka koji ju proslijeđuje primatelju poruke (F-10)
4. Korisnik vidi poruku koju je poslao

UC 9 - Primanje poruka

- Glavni sudionik: Plesač/Plesni direktor
- Cilj: Primiti poruku pomoću postojećeg sustava za razmjenu poruka (F-10)
- Sudionici: Baza podataka, FreeChat
- Preduvjet: -
- Opis osnovnog tijeka:

1. Korisnik dobiva obavijest o primljenoj poruci te ju može pregledati

UC 10 - Upravljanje korisničkim profilima

- Glavni sudionik: Administrator
- Cilj: Administratori mogu raditi određene akcije na profilima plesača i plesnih direktora (brisanje, sakrivanje, ...)
- Sudionici: Baza podataka
- Preduvjet: -
- Opis osnovnog tijeka:

1. Administrator odabire opciju za uređivanje korisničkih profila
2. Administratoru je omogućeno uređivanje korisničkih profila

UC 11 - Promjena iznosa godišnje članarine

- Glavni sudionik: Administrator
- Cilj: Postavljanje iznosa godišnje članarine za plesne direktore
- Sudionici: Baza podataka
- Preduvjet: -
- Opis osnovnog tijeka:

1. Administrator odabire opciju za uređivanje iznosa godišnje članarine
2. Administrator odabir novi iznos
3. Administrator dobiva potvrdu o promjeni ili postavljanju članarine (F-13)

UC 12 - Pregledavanje statistika

- Glavni sudionik: Plesni direktor
- Cilj: Plesni direktovi imaju mogućnost pregledavanja statistika prijava na audicije, uključujući broj prijavljenih plesača, vrste plesa i status prijava
- Sudionici: Baza podataka
- Preduvjet: -
- Opis osnovnog tijeka:
 1. Plesni direktor odabire opciju za pregled statistika
 2. Plesni direktor dobiva vizualno reprezentaciju statistika (F-15)

UC 13 - Slanje obavijesti plesačima o novim audicijama

- Glavni sudionik: Plesač
- Cilj: Aplikacija šalje notifikacije plesačima o novim audicijama na temelju njihovih preferencija (filtrirajući vrste plesa, lokacije)
- Sudionici: Baza podataka
- Preduvjet: -
- Opis osnovnog tijeka:
 1. Sustav šalje plesaču obavijest o novim audicijama (F-16)

UC 14 - Oporavak lozinke putem e-mail adrese

- Glavni sudionik: Neprijavljeni korisnik
- Cilj:
- Sudionici: Baza podataka
- Preduvjet: -
- Opis osnovnog tijeka:
 1. Korisnik odabire opciju za oporavak lozinke
 2. Korisnik upisuje e-mail adresu
 3. Sustav šalje lozinku na e-mail adresu (F-17)

UC 15 - Pregled arhiviranih audicija

- Glavni sudionik: Plesni direktor
- Cilj: Plesni direktovi mogu u povijesti vidjeti održane audicije i podatke
- Sudionici: Baza podataka
- Preduvjet: -
- Opis osnovnog tijeka:
 1. Plesni direktor odabire opciju za pregled arhiviranih audicija
 2. Sustav omogućuje plesnom direktoru pregled arhive audicija

UC 16 - Upravljanje profilom i uređivanje podataka

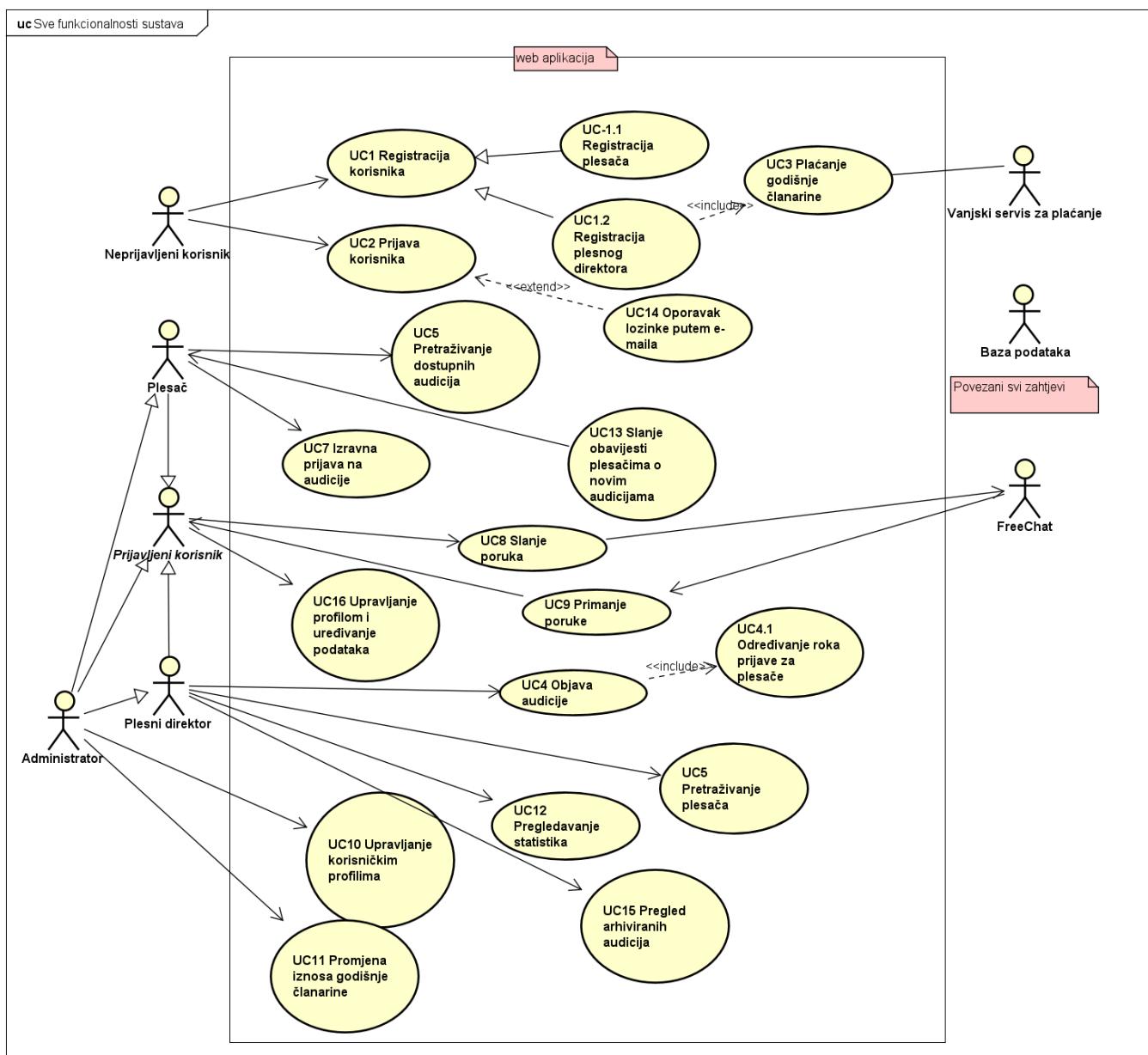
- Glavni sudionik: Plesač/Plesni direktor

- Cilj: Omogućiti upravljanje profilom, uključujući promjenu osobnih podataka, dodavanje portfolia i ažuriranje statusa (aktivan/neaktivran)
 - Sudionici: Baza podataka
 - Preduvjet: -
 - Opis osnovnog tijeka:

1. Korisnik odabire opciju za uređivanje profila
 2. Korisnik radi promjene u postavkama profila i podacima
 3. Korisnik pohranjuje promjene (F-11, F-19)

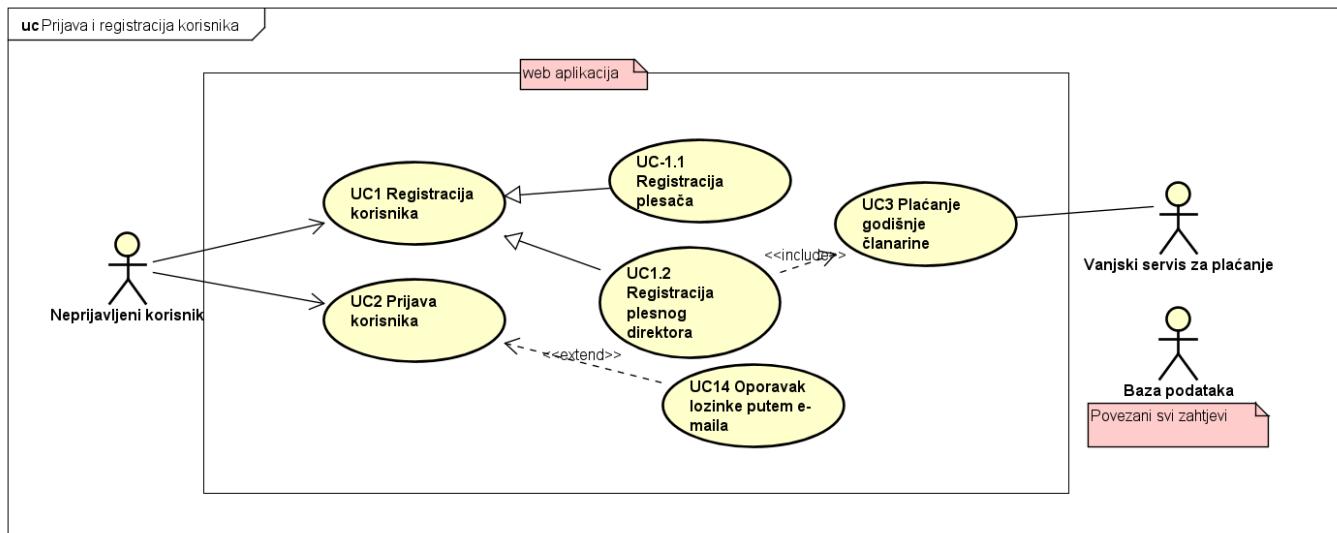
1. Visokorazinski dijagram obrazaca uporabe cijelog sustava

Visokorazinski dijagram cijelokupnog sustava



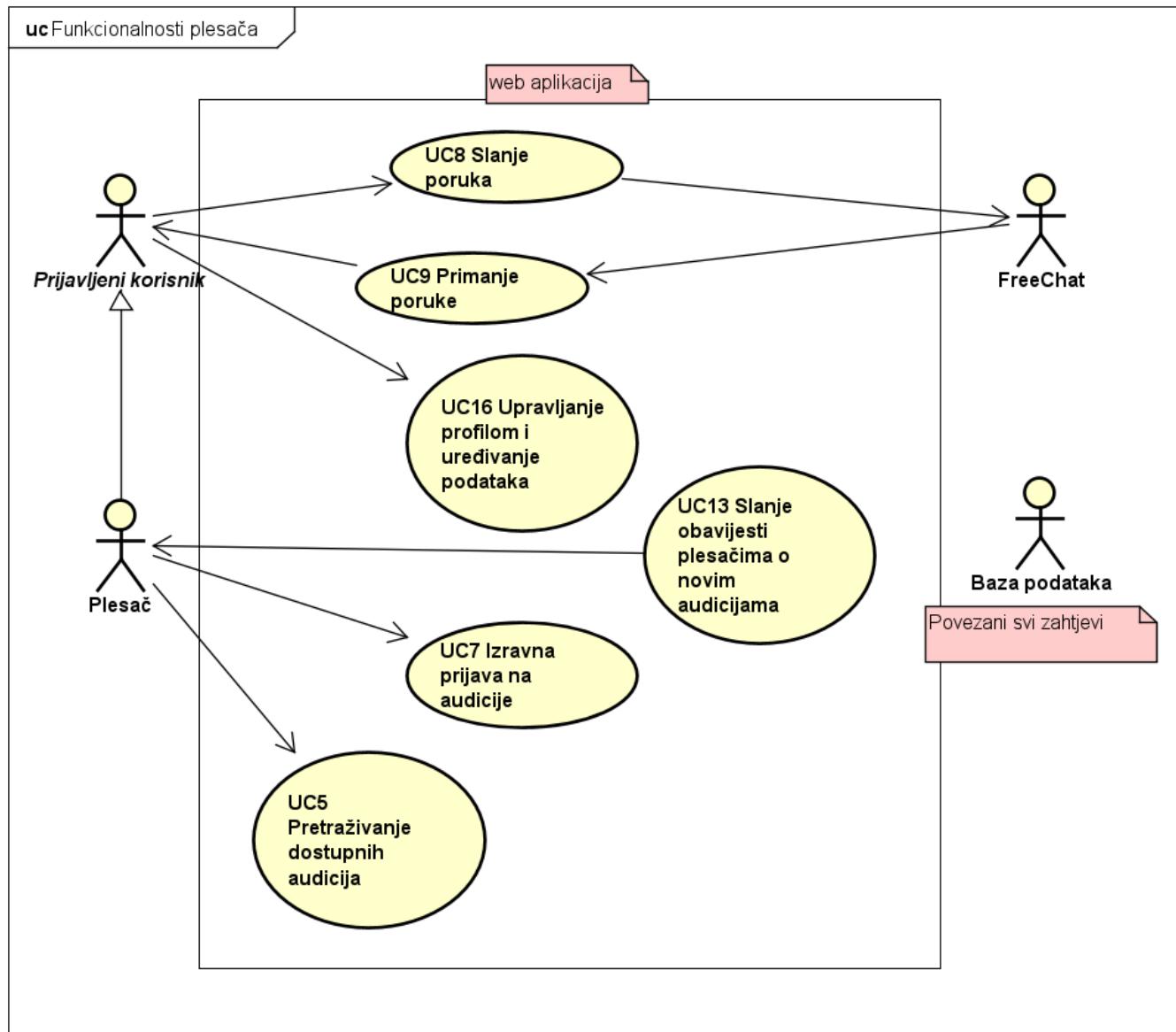
2. dijagram obrazaca uporabe za ključne funkcionalnosti

Dijagram obrazaca uporabe za prijavu i registraciju korisnika u sustav

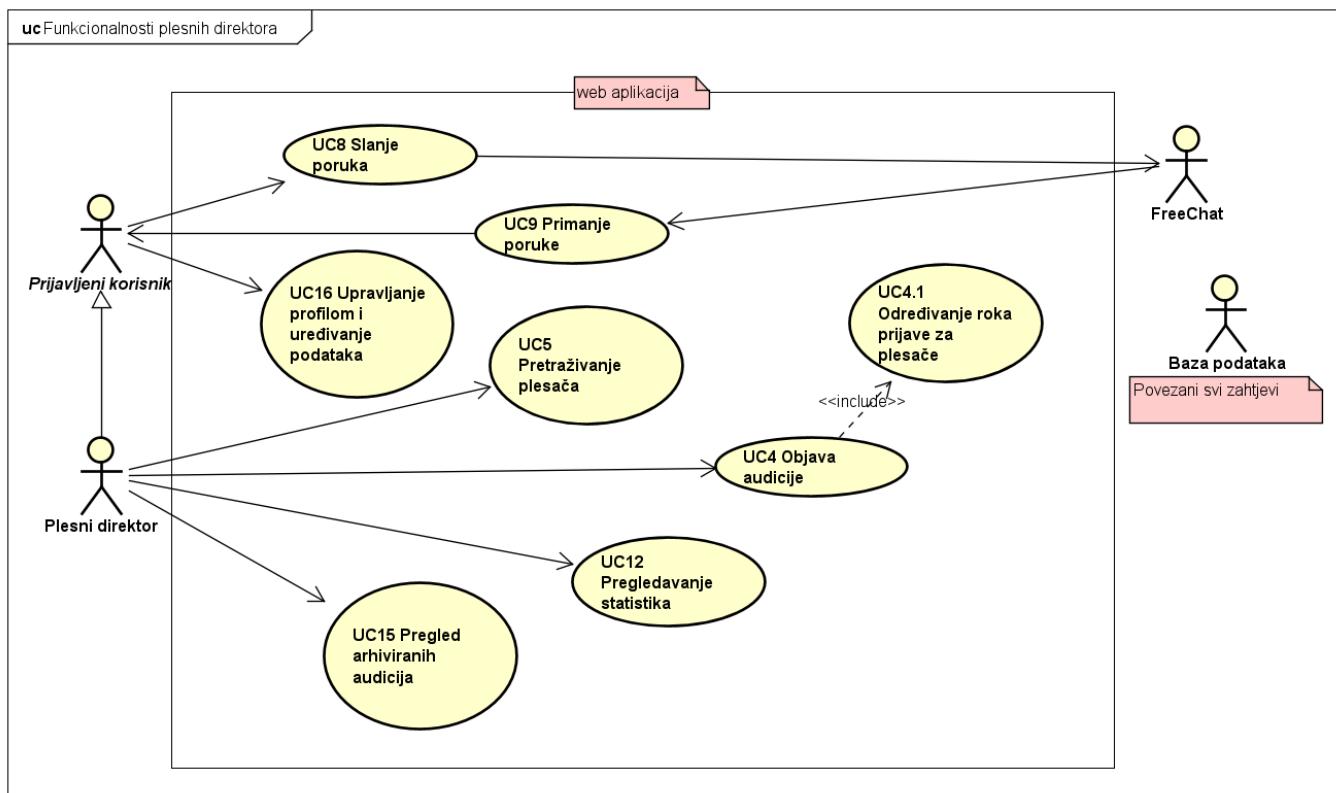


3. dijagram obrazaca uporabe za korisničke role

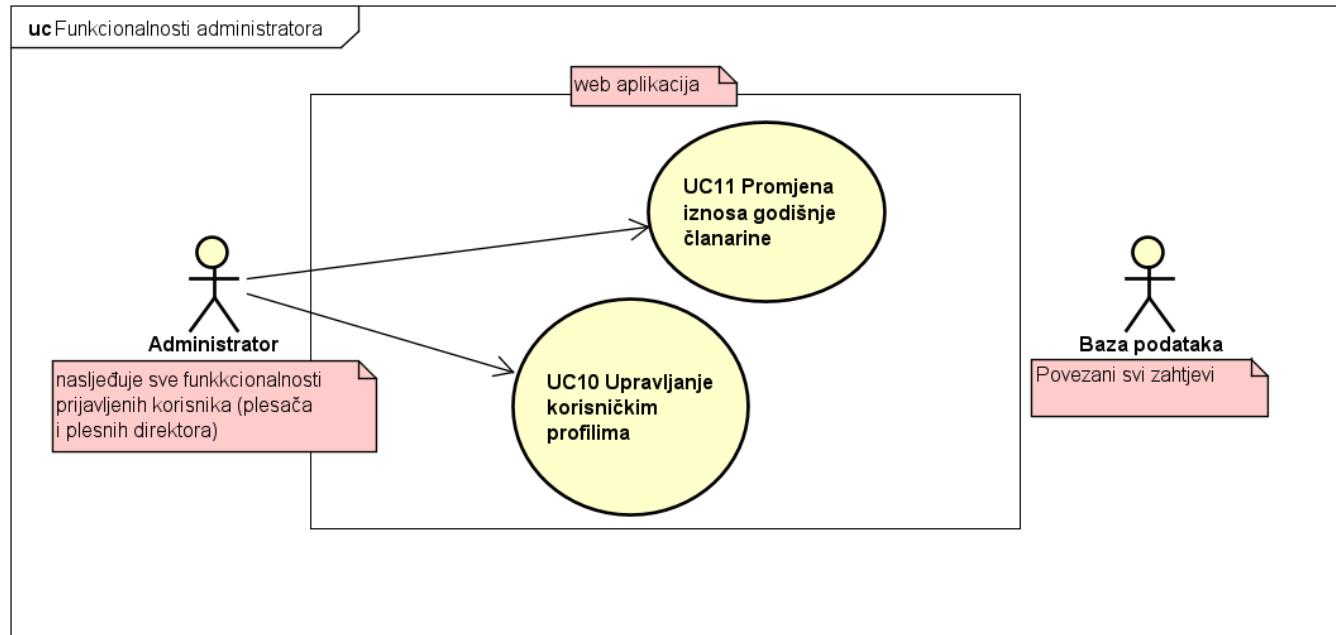
Dijagram obrazaca uporabe za plesača



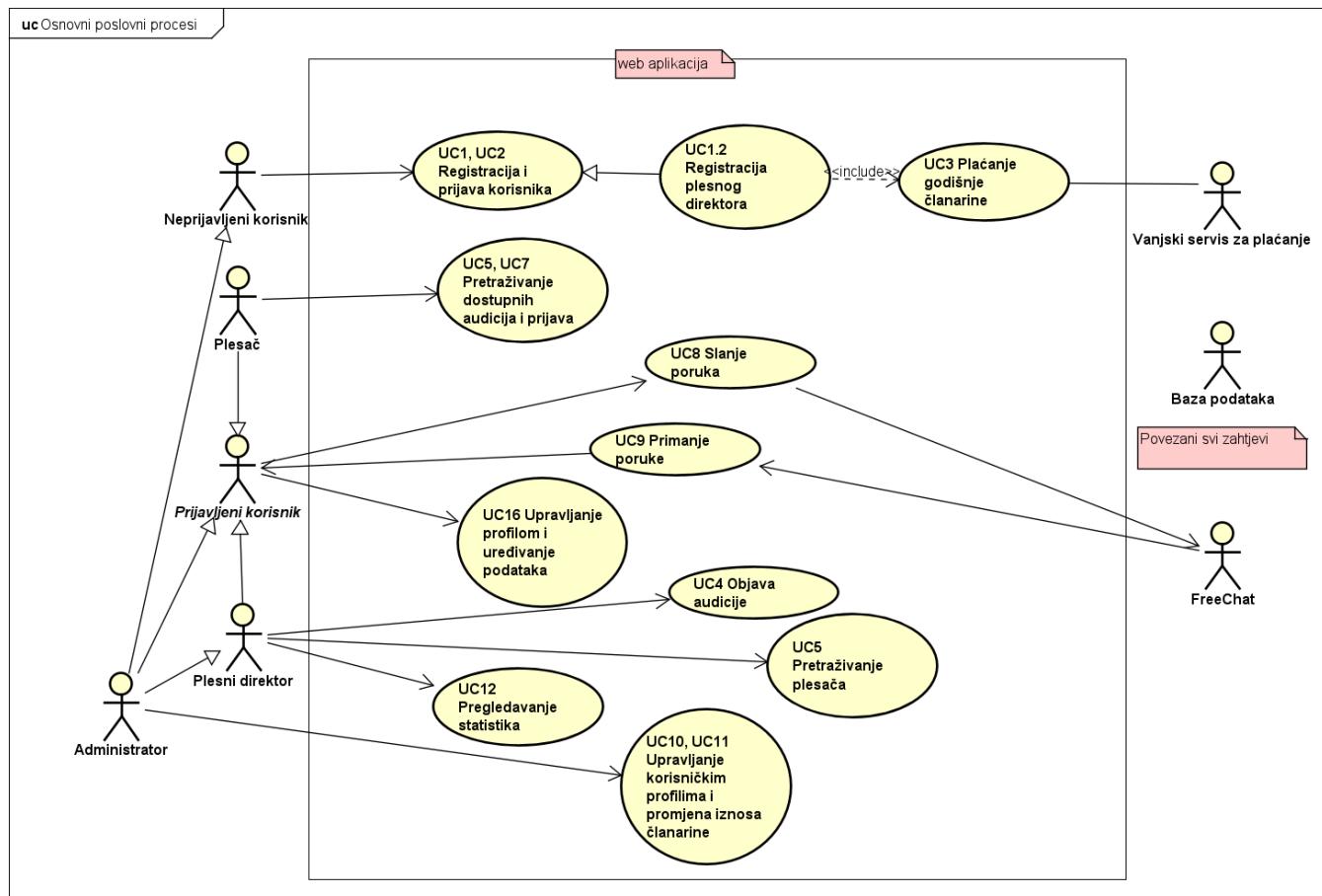
Dijagram obrazaca uporabe za plesnog direktora



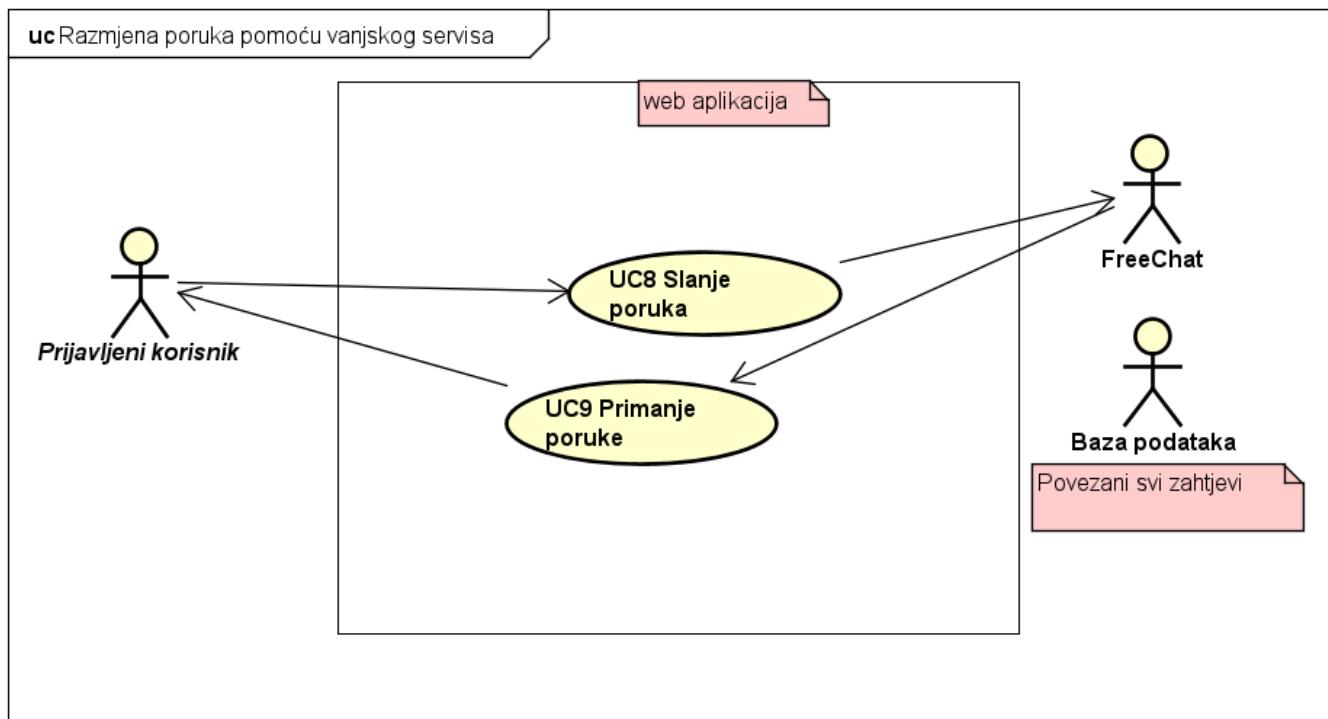
Dijagram obrazaca uporabe za administratora



4. dijagram obrazaca uporabe za osnovne poslovne procese

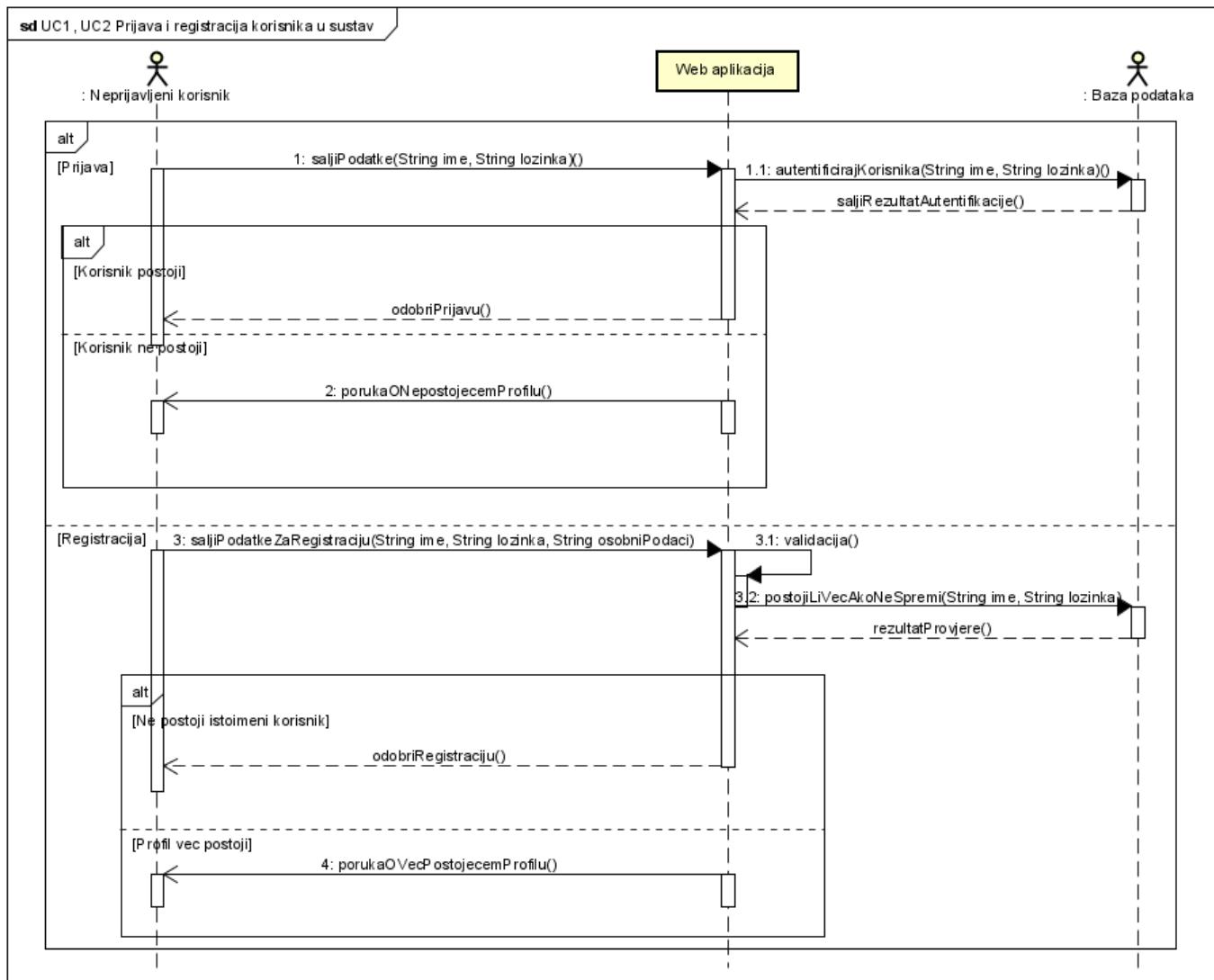


5. dijagram obrazaca uporabe za kritične sustave i integracije



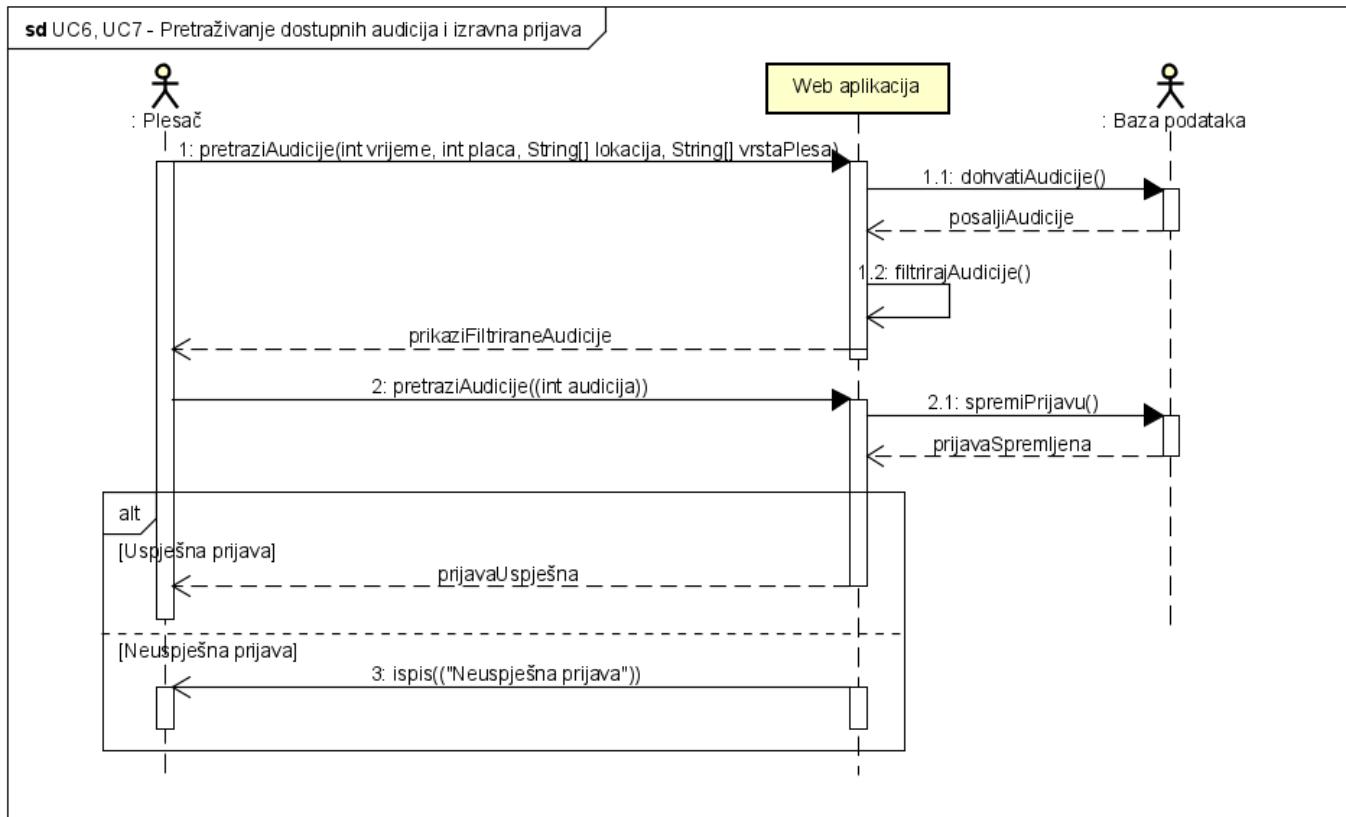
Sekvencijski dijagrami

UC1, UC2 Prijava i registracija korisnika u sustav



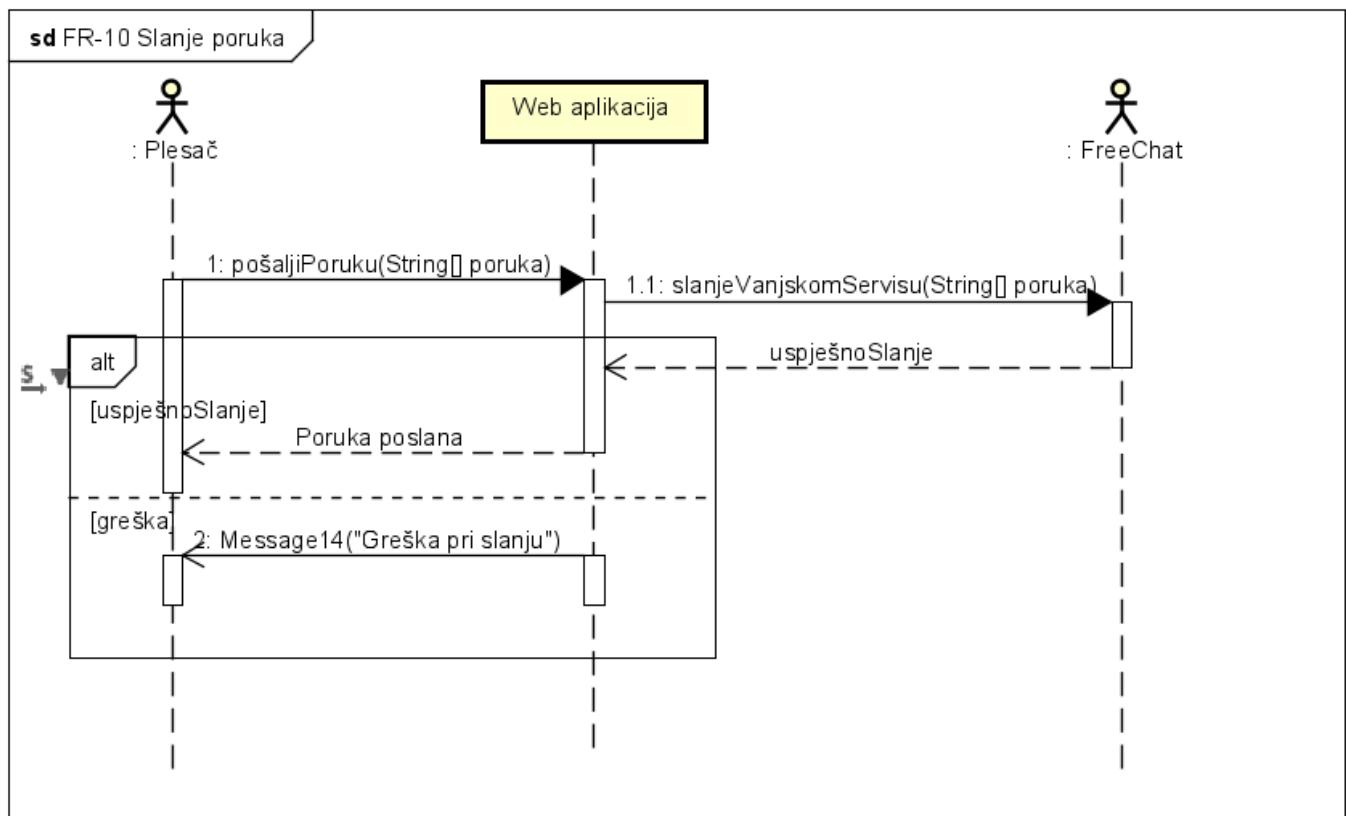
Ako je odabrana prijava, korisnik unosi svoje podatke (korisničko ime i lozinku) te pritišće gumb za prijavu. Taj zahtjev obrađuje server web aplikacije, tako da šalje zahtjev prema bazi podataka. Ako je baza podataka našla odgovarajućeg korisnika odobrava mu prijavu, inače ispisuje poruku o nepostojećem korisniku. Ako je odabrana registracija, korisnik unosi korisničko ime, lozinku i osobne podatke. Radi se validaciju te provjera postoji li već taj profil, ako je sve u redu korisnik se registrira te podaci spremaju u bazu.

UC6, UC7 Pretraživanje audicija pomoću filtera i izravna prijava na audiciju



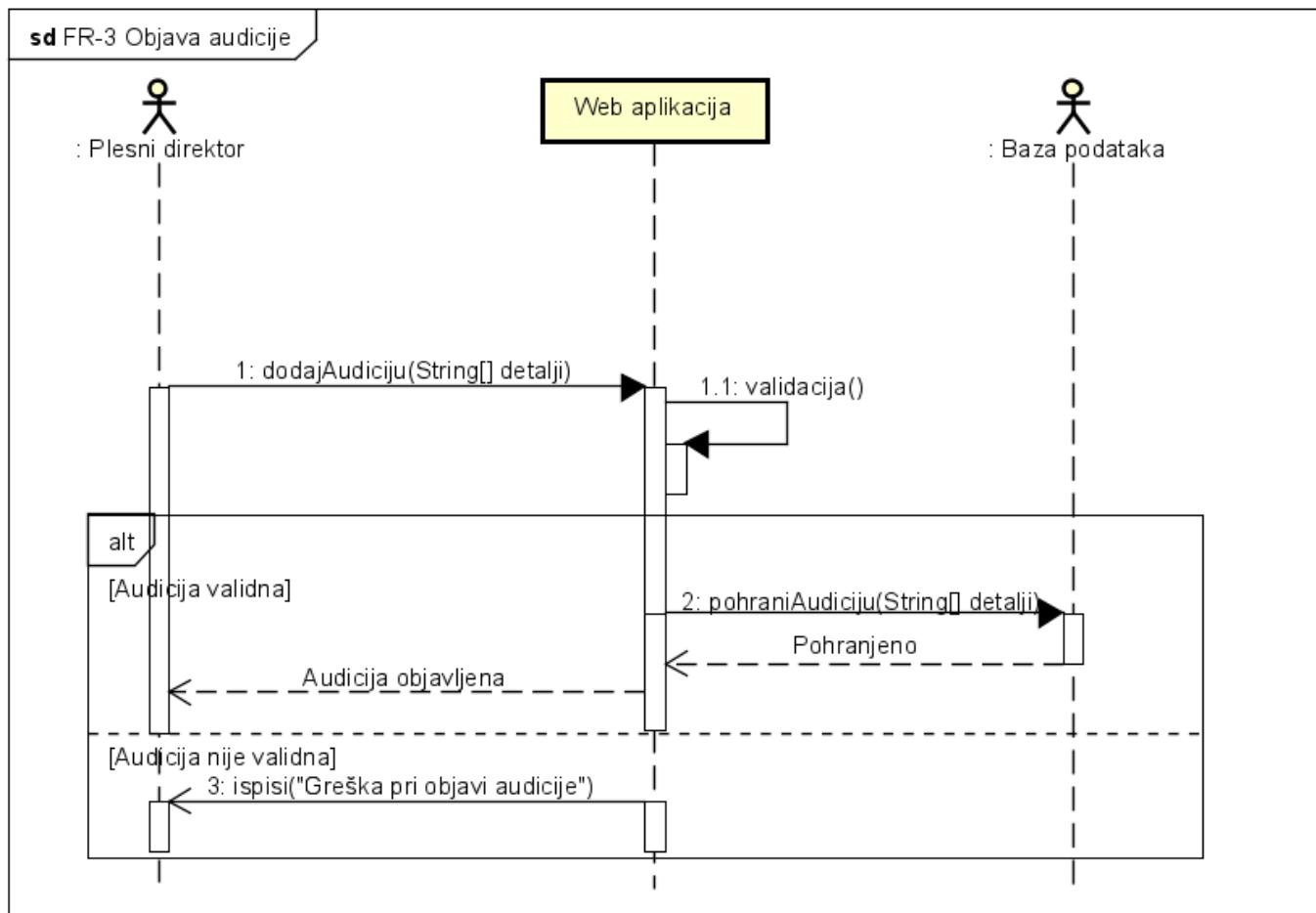
Korisnik(plesač) unosi filtere prema kojim želi pretražiti audicije. Web aplikacija od baze traži sve audicije pa ih filtrira prema zahtjevu korisnika te prikazuje. Korisnik se može izravno prijaviti, prijava se sprema u bazu.

UC8 Slanje poruka



Korisnik šalje zahtjev za slanjem poruke i njen sadržaj. Web aplikacija taj zahtjev proslijeđuje vanjskom servisu (FreeChat) koji ga obrađuje i vraća aplikaciji potvrdu da je poruka poslana. U slučaju greške pri obradi, korisniku se ispisuje poruka o grešci.

UC4 Objava audicije



Korisnik(plesni direktor) ispunjava obrazac sa svim detaljima audicije, oni se validiraju te ako su ispravni pohranjuju u bazu podataka i objavljuju. U slučaju neispravnosti podataka ispisuje se poruka o grešci.

Provjera uključenosti ključnih funkcionalnosti u obrasce uporabe

ID obrasca	Naziv obrasca	Funkcionalni zahtjevi koje obuhvaća
UC1	Registracija korisnika	F-1, F-2, F-7, F-8
UC2	Prijava korisnika	F-1
UC3	Plaćanje godišnje članarine za plesne direktore	F-2.1
UC4	Objava audicije	F-3, F-14
UC5	Pretraživanje plesača	F-4
UC6	Pretraživanje dostupnih audicija	F-5
UC7	Izravna prijava na audicije	F-6
UC8	Slanje poruka	F-10, F-9
UC9	Primanje poruka	F-10

ID obrasca	Naziv obrasca	Funkcionalni zahtjevi koje obuhvaća
UC10	Upravljanje korisničkim profilima	F-12
UC11	Promjena iznosa godišnje članarine	F-13
UC12	Pregledavanje statistika	F-15
UC13	Slanje obavijesti plesačima o novim audicijama	F-16
UC14	Oporavak lozinke putem e-mail adrese	F-17
UC15	Pregled arhiviranih audicija	F-18
UC16	Upravljanje profilom i uređivanje podataka	F-11, F-19

4. Arhitektura i dizajn sustava

Arhitektura sustava

Opis arhitekture

Stil arhitekture

Stil arhitekture kojeg će aplikacija slijediti je klijent-poslužitelj. Zbog jednostavnosti aplikacije je ovaj stil najprikladniji. Tim odabirom smo također osigurali da će nam testiranje aplikacije biti jednostavnije nego da smo se odlučili za neku komplikiraniju stil arhitekture. Testiranje aplikacije je vrlo bitna komponenta koja može znatno ubrzati razvoj. Također aplikacija ne treba biti jako skalabilna zbog toga što nije za širu javnost, nego za plesače i plesne direktore.

Podsistavi

- Frontend
 - **Web preglednik** je softverski alat koji omogućuje korisnicima pristup i pregledavanje sadržaja na internetu. On prevodi kodove (HTML, CSS, JavaScript) i prikazuje stranicu u razumljivom i preglednom obliku. Kada korisnik zatraži web stranicu, preglednik prikupi datoteke sa određenog servera kojemu je poslao zahtjev. Web preglednici postoje na raznim uređajima (računala, tableti, mobiteli, pametni televizori, ...).
 - Za klijentski dio aplikacije (frontend) koristimo React.js. React.js je JavaScript biblioteka za izradu web aplikacija. Jedan je od najpopularnijih alata za izgradnju modernih web aplikacija te idealan za projekte koji zahtijevaju dinamičnost, responzivnost i visoku interaktivnost. Pomoću React.js-a ćemo dobiti dinamično i responzivno korisničko sučelje koje će olakšati i ubrzati razvoj u odnosu na pisanje direktno u JavaScriptu. Razvojno okruženje je Visual Studio Code.
- Backend
 - **Web poslužitelj** omogućava komunikaciju klijenta s aplikacijom. Za komunikaciju se najčešće koristi protokol HTTP i HTTPS. On dobiva zahtjeve od web preglednika, parsira ih i šalje podatke

nazad u odgovarajućem formatu. Zbog velikog internetskog prometa koji u današnje vrijeme nije neobičan, poslužitelji moraju biti skalabilni i imati veliku propusnost da bi bili pouzdani i da bi podržavali korisničke zahtjeve.

- **Web aplikacija** koristi se putem internetskog preglednika i radi na web poslužitelju, a ne na korisničkom računalu. Njima se pristupa online. Web aplikacija obrađuje zahtjeve i komunicira s bazom podataka te korisniku vraća odgovor u obliku HTML dokumenta i ostalih formata koje web preglednik prevodi.
- Za serverski dio aplikacije (backend) koristimo programski jezik Java i Spring Boot programski okvir (framework). Spring Boot omogućava izradu lako održivih web aplikacija, bez kompleksnosti ručne konfiguracije koja se obično povezuje s Java projektima. Radno okruženje za backend je Eclipse IDE.

Spremište podataka

Baza podataka je spremište za podatke od web aplikacije i jako važna komponenta sustava jer može čuvati vrlo osjetljive i bitne podatke i informacije. Koristi se za spremanje i dohvaćanje informacija iz web aplikacije. Baza mora biti otporna na greške, prekide i razne fizičke nepogode koje su neizbjegne. Ovisno o važnosti informacija, za bazu se mora dobro odabrati razina tolerancije grešaka koje utječu na sigurnost i integritet podataka (RAID razine).

Za bazu podataka koristimo PostgreSQL s kojom ćemo komunicirati putem Klijent-Server modela. Baza će biti relacijskog tipa. PostgreSQL je globalno popularna tehnologija koju koriste velike i profitabilne kompanije (Apple, Reddit, Spotify, ...). Razvojno okruženje je PgAdmin.

Mrežni protokoli

Za komunikaciju između podsustava koristimo HTTPS protokol kojim šaljemo HTML i JSON podatke.

Sklopovskoprogramski zahtjevi

Aplikacija će biti napravljena za web preglednike, pa će je trebati uskladiti s većinom preglednika koje korisnici mogu koristiti. To znači da će morati raditi i na starijim verzijama preglednika bez problema prikazivanja elemenata i kompatibilnosti.

Obrazloženje odabira arhitekture

- Kao što je već rečeno, odabrana je arhitektura klijent-poslužitelj zbog jednostavnosti. Klijent-poslužitelj arhitektura nam također pomaže u visokoj koheziji jer su glavne komponente jasno povezane. Tako smo osigurali da je struktura projekta pregledna i aplikacija jednostavna za testirati jer je to bitan aspekt razvoja. Drugi razlog je to što se na aplikaciju mogu nadodati nove mogućnosti i kasnije po potrebi se može promijeniti u neku kompleksniju i skalabilniju arhitekturu.

Organizacija sustava na visokoj razini

- Klijent poslužitelj - Sloj koji vidi korisnik (frontend) komunicira sa slojem koji upravlja zahtjevima i podacima (backend i baza podataka).
- Baza podataka - Koristimo relacijsku bazu podataka i ona će imati informacije o korisnicima, audicijama i projektima.
- Grafičko sučelje - Platforma će biti napravljena za web preglednike, a s poslužiteljem će komunicirati pomoću HTTP protokola.

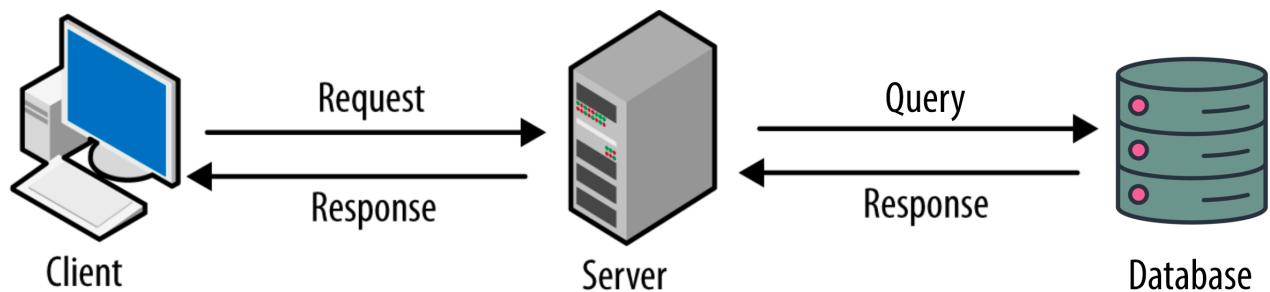
Organizacija aplikacije

Frontend

- Frontend sloj prenosi informacije između korisnika i backenda pomoću klijent-server arhitekture. On je također bitan jer je odgovoran za izgled i responzivnost aplikacije. Dizajn frontenda (UI) i korisničkog iskustva (UX) može značajno utjecati na uspjeh aplikacije. Zato je bitno da on bude lijep, jednostavan i intuitivan za korištenje.

Backend

- Backend sloj aplikacije se brine o slanju i primanju podataka i zahtjeva od frontend sloja, te slanja i primanja podataka i upita od baze podataka. Bitno je da je backend sloj bude siguran i otporan na različite greške.



Baza podataka

Za našu web stranicu koristit ćemo relacijsku bazu podataka PostgreSQL. Baza podataka sastoji se od relacija koje su definirane svojim imenom i skupom atributa. Zadaća baze podataka je brza i jednostavna pohrana, izmjena i dohvata podataka za daljnju obradu. Baza podataka ove aplikacije sastoji se od sljedećih tablica:

- Users

- Usertype
- Dancer
- Director
- Admin
- Dance
- Dancer_dance
- Audition
- Audition_application
- Audition_dance
- Direct_offer
- Portfolio
- Portfolio_photos
- Portfolio_videos
- Payment
- Forgotpassword

Opis tablica

Users

- **Users** sadrži informacije o korisniku aplikacije. Sadrži atribute: age, confirmed, finishedoauth, id, usertype_id, contact, email, gender, location, name, oauth, password, surname, username. Svaki korisnik ima točno jedan id, i id pripada samo jednom korisniku aplikacije pa je ovaj entitet u vezi *One-to-One* s entitetima dancer, director i admin. Veza između korisnika i portfolija je *One-to_many* ostvaruje se preko user id-a, isto vrijedi i za vezu između korisnika i poruka jer svaki korisnik može primati i slati više poruka.

Atribut	Tip podatka	Opis varijable
id	INT	Jedinstveni identifikator korisnika (primarni ključ)
usertype_id	INT	Tip korisnika (strani ključ)
username	VARCHAR(255)	Username korisnika, ne može biti NULL
name	VARCHAR(255)	Ime korisnika
surname	VARCHAR(255)	Prezime korisnika
email	VARCHAR(255)	Email adresa korisnika (jedinstvena)
password	VARCHAR(255)	Lozinka korisnika
oauth	VARCHAR(255)	Prijava preko oautha
finishedoauth	BOOLEAN	Je li autentifikacija završena
location	VARCHAR(255)	Lokacija korisnika
gender	VARCHAR(255)	Spol korisnika

Atribut	Tip podatka	Opis varijable
age	INT	Godine korisnika
contact	VARCHAR(255)	Kontakt korisnika
confirmed	BOOLEAN	Potvrda preko emaila

Usertype

- **Usertype** sadrži atribute: user_id, type. Ovaj entitet je u vezi *One-to-Many* s entitetom user koja se ostvaruje preko usertype_id što je strani ključ u tablici user i primarni ključ u tablici usertype.

Atribut	Tip podatka	Opis varijable
user_id	INT	Jedinstveni identifikator tipa korisnika (primarni ključ)
type	VARCHAR(255)	Naziv tipa korisnika ('plesač', 'plesni direktor', 'admin')

Dancer

- **Dancer** sadrži atribute: id, inactive, inactiveuntil. Ovaj entitet je u vezi *One-to-One* s entitetom user i ona se ostvaruje preko stranog ključa id koji upućuje na primarni ključ id tablice user. U vezi *Many-to-Many* s entitetom dance jer svaki plesač može biti stručan u više plesova, a jedna vrsta plesa može pripadati više plesača. Ta veza se implementira kroz dancer_dance. Entitet je u vezi *One-to-Many* s entitetom audition_application.

Atribut	Tip podatka	Opis varijable
id	INT	Jedinstveni identifikator plesača (primarni ključ i strani ključ)
inactive	BOOLEAN	Status profila (aktivan/neaktivran)
inactiveuntil	DATE	Ako je profil neaktivran, do kada

Director

- **Director** sadrži atribute: id, post_br, paid, subscription. Entitet je u vezi *One-to-One* s entitetom user koja se implementira preko stranog ključa id i upućuje na primarni ključ id u tablici user. U vezi *One-to-Many* s entitetom audition i entitetom payment.

Atribut	Tip podatka	Opis varijable
id	INT	Jedinstveni identifikator plesnog direktora (primarni ključ i strani ključ)
paid	BOOLEAN	Je li članstvo plaćeno
subscription	DATE	Do kada je članstvo aktivno

Admin

- **Admin** sadrži atribut: id, subscriptionprice. Ovaj entitet je u vezi *One-to-One* s entitetom user koja se implementira preko id-a tablice user.

Atribut	Tip podatka	Opis varijable
id	INT	Id admina, primarni ključ i strani ključ
subscriptionprice	BIGINT	Cijena članstva

Dance

- **Dance** sadrži atribut: id, name. Entitet je u vezi *Many-to-Many* s entitetom dancer.

Atribut	Tip podatka	Opis varijable
id	INT	Jedinstveni identifikator vrste plesa (primarni ključ)
name	VARCHAR(255)	Naziv vrste plesa

Dancer_dance

- **Dancer_dance** sadrži atribut: dance_id, dancer_id. Entitet se koristi za vezu između dancer i dancer_dance, ima strani ključ dancer_id prema tablici dancer i strani ključ dance_id prema tablici dance.

Atribut	Tip podatka	Opis varijable
dance_id	INT	Jedinstveni identifikator plesa (strani ključ)
dancer_id	INT	Identifikator plesača (strani ključ)

Audition

- **Audition** sadrži atribut: id, user_id, datetime, location, description, positions, wage, deadline, archived, creation, subscribed. Entitet je u vezi *Many-to-One* s entitetom director i ostvaruje se preko id te u vezi *One-to-Many* s entitetom audition_application.

Atribut	Tip podatka	Opis varijable
id	INT	Jedinstveni identifikator audicije (primarni ključ)
user_id	INT	Identifikator korisnika (strani ključ)
datetime	TIMESTAMP	Datum i vrijeme audicije
location	VARCHAR(255)	Mjesto održavanja audicije
description	VARCHAR(255)	Opis audicije
positions	INT	Broj otvorenih pozicija
wage	INT	Plaća za poziciju
deadline	TIMESTAMP	Rok prijave na audiciju

Atribut	Tip podatka	Opis varijable
archived	BOOLEAN	Je li audicija arhivirana
creation	TIMESTAMP	Vrijeme kreiranja audicije
subscribed	INT	Broj zauzetih pozicija

Audition_application

- **Audition_application** sadrži atribute: audition_id, dancer_id, id, datetime, status. Entitet je u vezi *Many-to-One* s entitetom dancer i entitetom audition. Veza s entitetom dancer se ostvaruje preko dancer_id, a s entitetom audition preko audition_id.

Atribut	Tip podatka	Opis varijable
audition_id	INT	Identifikator audicije (strani ključ)
dancer_id	INT	Identifikator plesača (strani ključ)
id	INT	Jedinstveni identifikator prijave (primarni ključ)
datetime	TIMESTAMP	Datum i vrijeme prijave na audiciju
status	VARCHAR(255)	Status prijave

Audition_dance

- **Audition_dance** sadrži atribute: audition_id, dance_id. Ovaj entitet je u vezi *Many-to-One* s entitetom audition jer svaka audicija može imati više vrta plesa.

Atribut	Tip podatka	Opis varijable
audition_id	INT	Jedinstveni identifikator audicije (strani ključ)
dance_id	INT	Identifikator plesa (strani ključ)

Direct_offer

- **Direct_offer** sadrži atribute: dancer_id, director_id, id, message, created_at, state. Entitet je u vezi *Many-to-Many* s entitetima dancer i director.

Atribut	Tip podatka	Opis varijable
dancer_id	INT	Jedinstveni identifikator plesača (strani ključ)
director_id	INT	Jedinstveni identifikator direktora (strani ključ)
id	INT	Jedinstveni identifikator poruke (primarni ključ)
message	VARCHAR(255)	Sadržaj poruke, ne može biti NULL
created_at	TIMESTAMP	Datum i vrijeme slanja poruke
state	VARCHAR(255)	Status poruke

Portfolio

- **Portfolio** sadrži atribute: id, user_id, description. Entitet je u vezi *Many-to-One* s entitetom user. Veza se implementira preko stranog ključa user_id koji upućuje na primarni ključ id iz tablice user.

Atribut	Tip podatka	Opis varijable
id	INT	Jedinstveni identifikator portfolia (primarni ključ)
user_id	INT	Korisnik kojem pripada portfolio (strani ključ)
description	TEXT	Opis stavke u portfoliju

Portfolio_photos

- **Portfolio_photos** sadrži atribute: portfolio_id, photos. Entitet je u vezi *One-to-One* s entitetom portfolio i ostvaruje se preko stranog ključa portfolio_id koji odgovara primarnom ključu id iz tablice portfolio.

Atribut	Tip podatka	Opis varijable
portfolio_id	INT	Jedinstveni identifikator portfolia (strani ključ)
photos	VARCHAR(255)	URL slike

Portfolio_videos

- **Portfolio_videos** sadrži atribute: portfolio_id, videos. Entitet je u vezi *One-to-One* s entitetom portfolio i ostvaruje se preko stranog ključa portfolio_id.

Atribut	Tip podatka	Opis varijable
portfolio_id	INT	Jedinstveni identifikator portfolia (strani ključ)
videos	VARCHAR(255)	URL videa

Payment

- **Payment** sadrži atribute: id, user_id, date, amount, expiration. Entitet je u vezi *Many-to-One* s entitetom user koja se ostvaruje preko user_id.

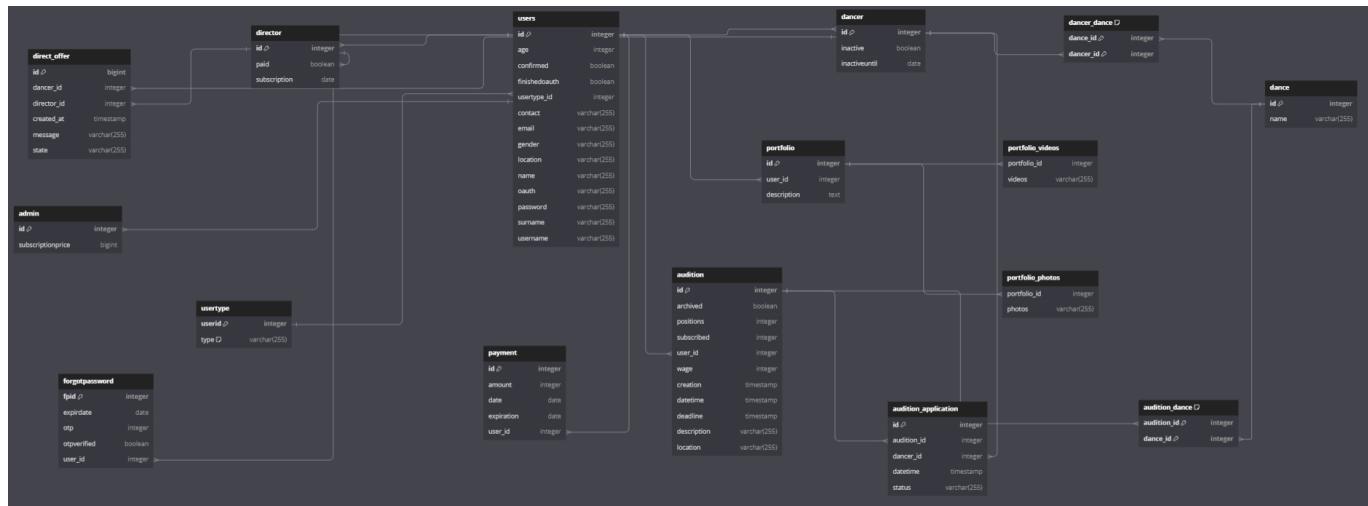
Atribut	Tip podatka	Opis varijable
id	INT	Jedinstveni identifikator uplate (primarni ključ)
user_id	INT	Identifikator korisnika (strani ključ)
date	DATE	Datum uplate
amount	INT	Iznos uplate
expiration	DATE	Datum isteka članstva

Forgotpassword

- **Forgotpassword** sadrži atribute: user_id, expirdate, fpid, otp, otpverified. Ovaj entitet je u vezi *Many-to-One* s entitetom user i ostvaruje se preko stranog ključa user_id.

Atribut	Tip podatka	Opis varijable
user_id	INT	Jedinstveni identifikator korisnika (strani ključ)
expirdate	DATE	Datum isteka koda
fpid	INT	Jedinstveni identifikator (primarni ključ)
otp	INT	Kod za promjenu lozinke
otpverified	BOOLEAN	Kod ukucan (da/ne)

Dijagram baze podataka



Relacijska shema baze podataka

Dijagram razreda

Na prikazanim dijagramima razreda prikazana je struktura backend dijela aplikacije koja odgovara **MVC arhitekturi**. Dijagram obuhvaća ključne komponente aplikacije, uključujući kontrolere, modele i njihove međusobne odnose. Ovdje je kratak opis svakog segmenta:

Controller razredi

- Razredi koji nasljeđuju osnovni **Controller** razred služe za obradu zahtjeva korisnika.
- **Metode** implementirane unutar tih razreda manipuliraju modelima kako bi dohvaćale, obrađivale ili vraćale tražene podatke.
- Rezultati metoda često se vraćaju u obliku lista koje sadrže objekte modela, čime se omogućava jednostavan prikaz podataka na klijentskom dijelu aplikacije.

Model razredi

- **Modeli** predstavljaju preslikavanje (eng. *mapping*) struktura baze podataka u aplikaciju.
- Svaka klasa modela odgovara entitetu iz baze podataka, dok članovi klase (atributi) odgovaraju poljima unutar tablica baze.
- Ovi razredi omogućuju:
 - Jednostavnu interakciju s bazom podataka putem ORM alata.
 - Manipulaciju podacima kroz objekte umjesto direktnim SQL upitima.

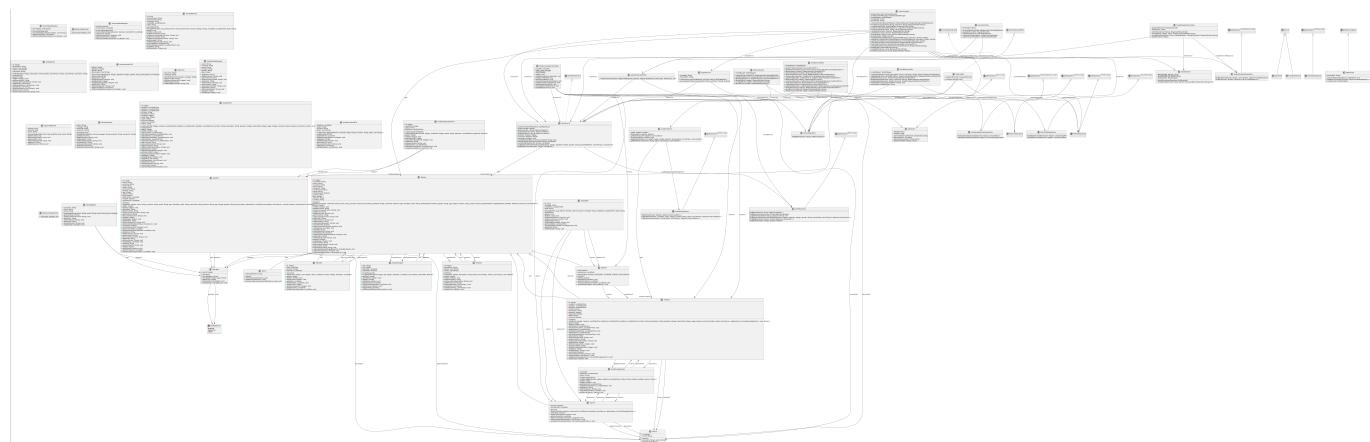
Povezanost

- Dijagrami razreda jasno prikazuju odnose između modela, kontrolera i baze podataka, uključujući:
 - **Nasljeđivanje** – odnos između razreda unutar hijerarhije.
 - **Asocijacije** – veze između različitih razreda.
 - **Kompoziciju i agregaciju** – gdje je to potrebno za definiranje složenijih veza.

Ključne prednosti arhitekture

- **Modularnost** – Omogućuje jednostavno dodavanje novih funkcionalnosti bez utjecaja na postojeći kod.
- **Razdvajanje odgovornosti** – Svaki sloj ima jasno definiranu svrhu, što olakšava razumijevanje i održavanje koda.
- **Ponovna upotreba koda** – Razredi i metode mogu se koristiti u različitim dijelovima aplikacije.

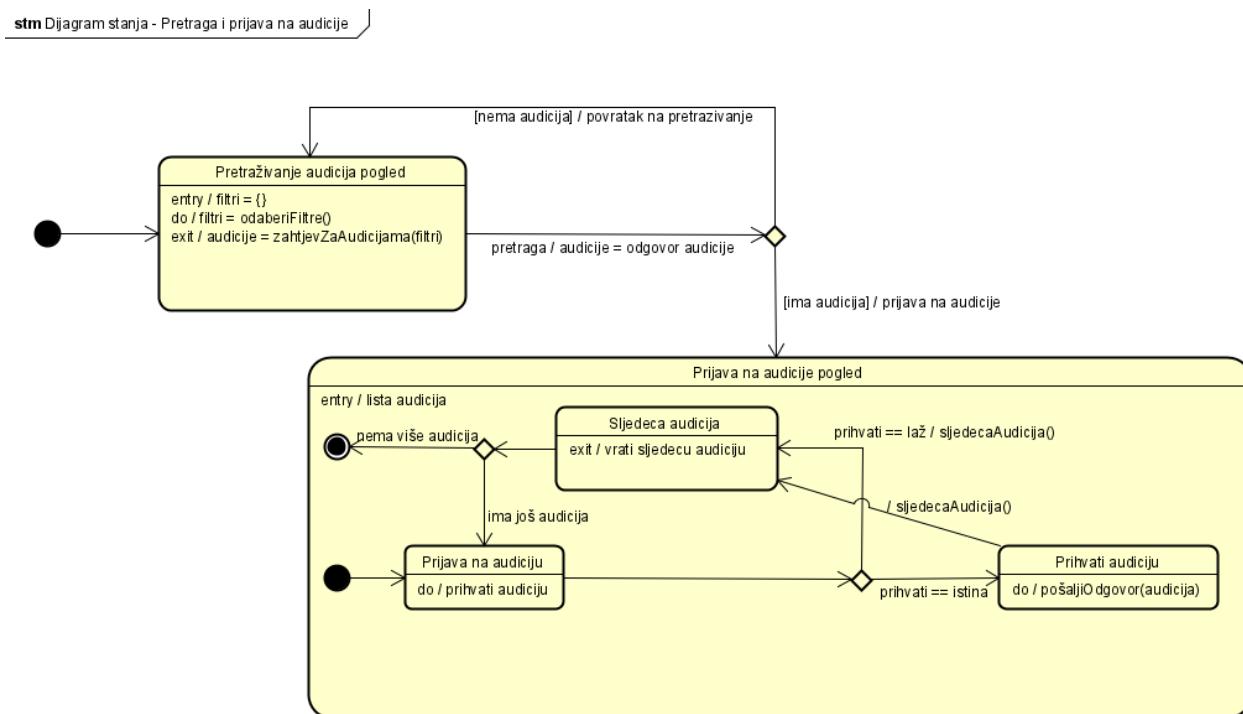
Ovakva arhitektura čini aplikaciju skalabilnom, razumljivom i jednostavnom za održavanje. Za više detalja o implementaciji pojedinih razreda, pogledajte priloženi dijagram razreda.



Dinamičko ponašanje aplikacije

UML dijagrami stanja

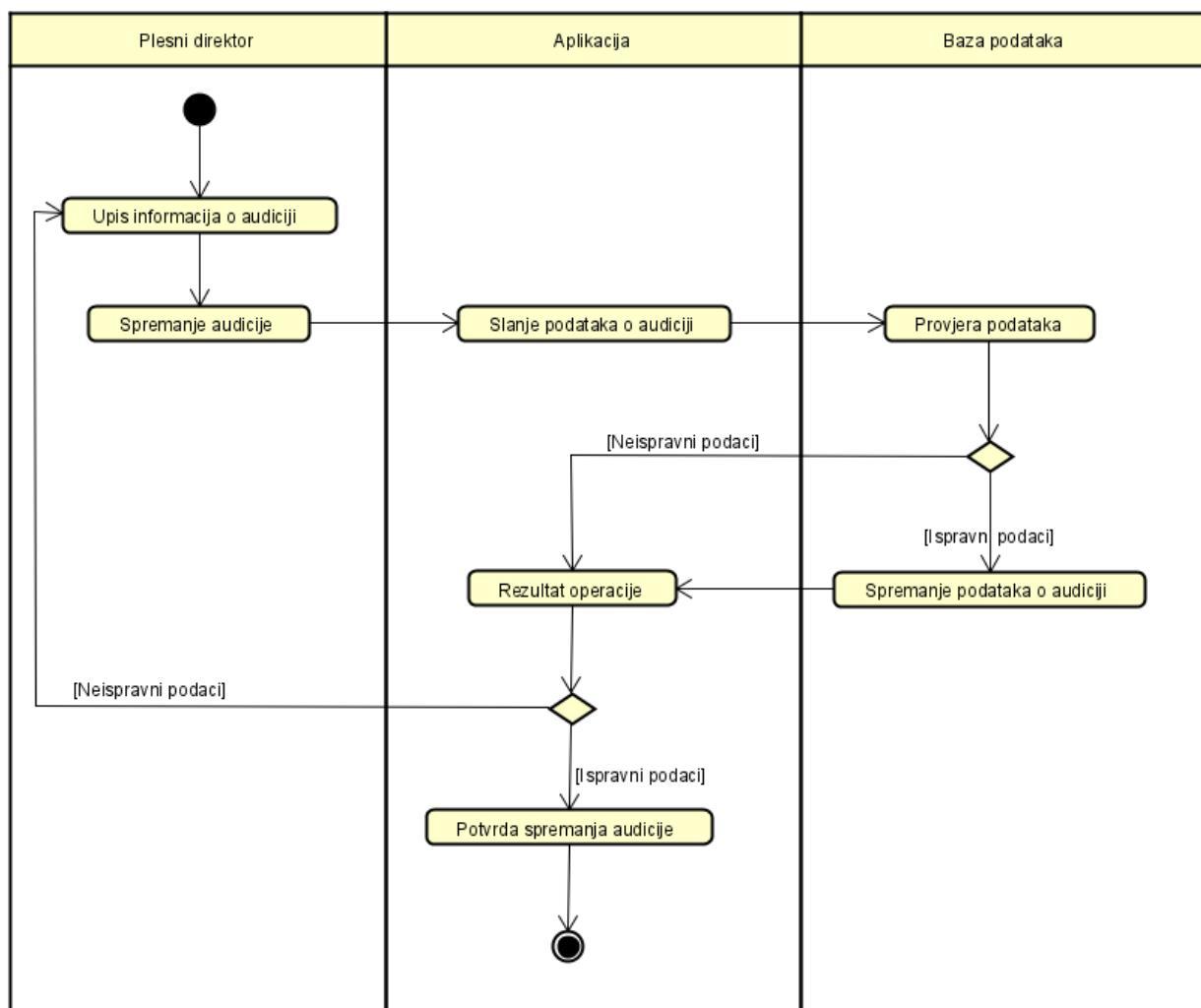
- **Dijagram stanja za pretragu i prijavu na audicije**
 - Ovaj dijagram prikazuje pretragu audicija pomoću filtera i prijavu na odabrane audicije



UML dijagrami aktivnosti

- Ovaj dijagram prikazuje objavu audicije koju obavlja plesni direktor

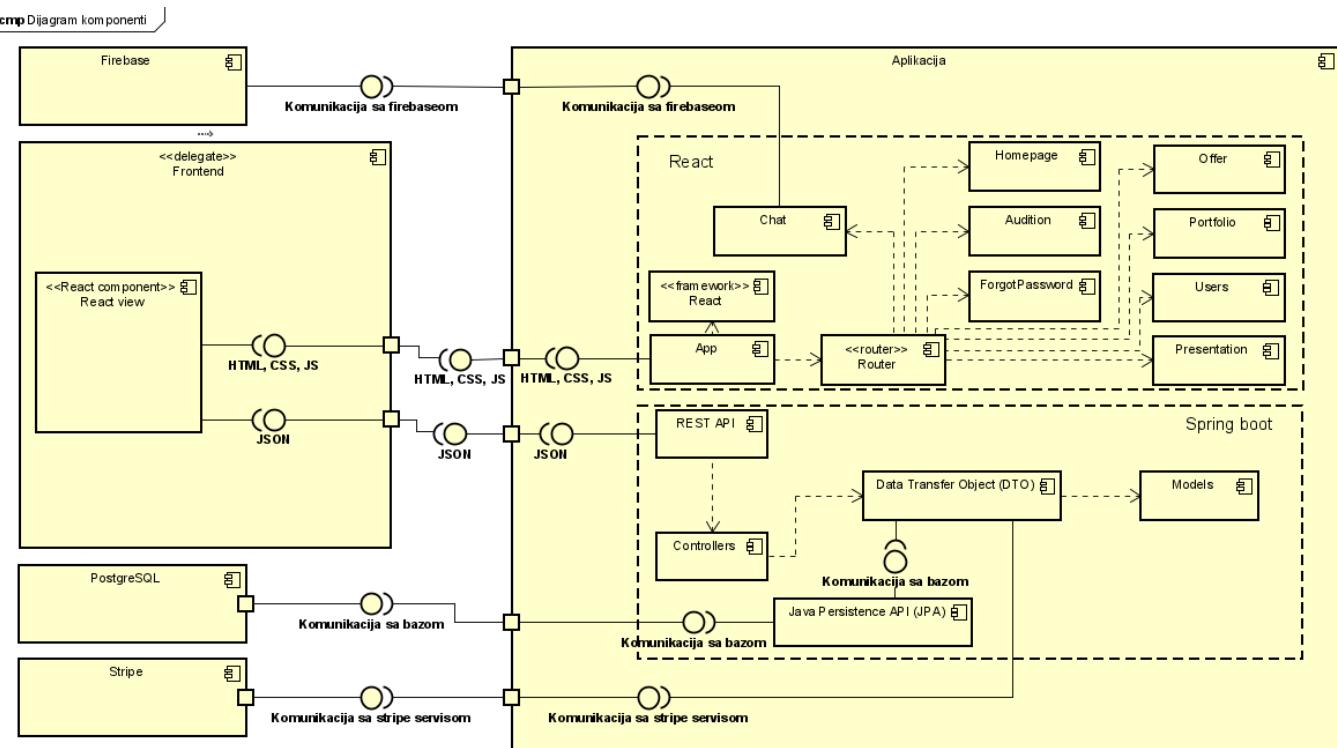
act Dijagram aktivnosti: Plesni direktor objava audicija



5. Arhitektura komponenata i razmještaja

Dijagram komponenata

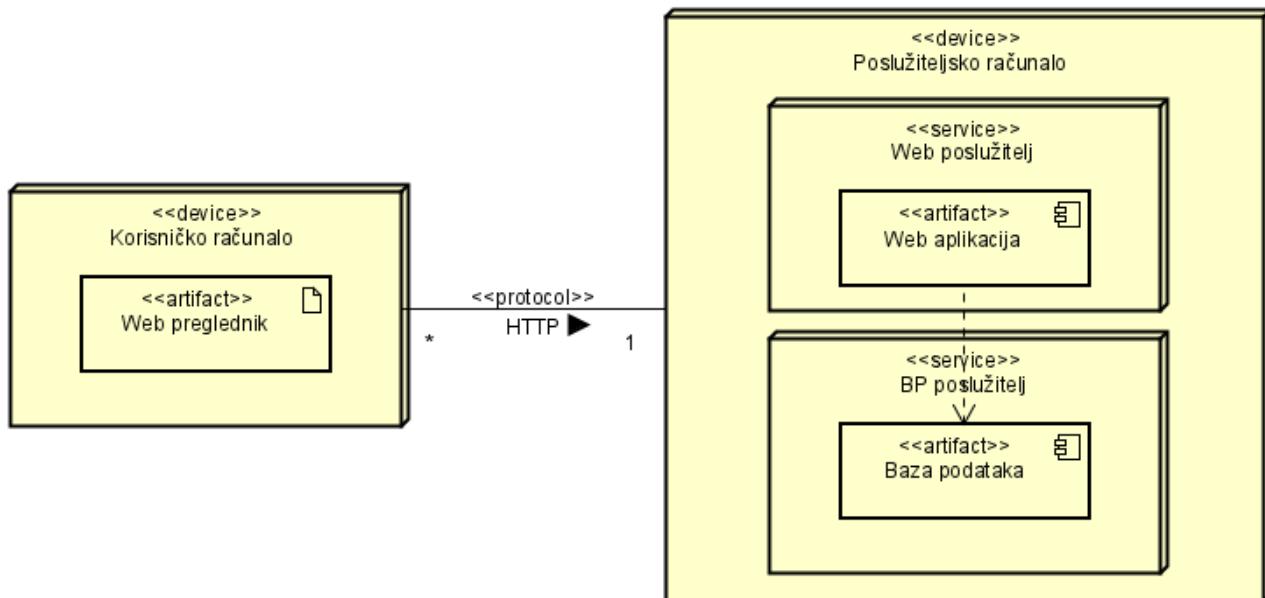
- Dijagram prikazuje kako komponente komuniciraju i dijele podatke jedna sa drugom



Dijagram razmještaja

- Dijagram prikazuje kako sustav radi sa visoke razine apstrakcije

pkg Dijagram razmještaja



6. Ispitivanje programskog rješenja

Ispitivanje komponenti

Cilj ispitivanja komponenti je provjera osnovnih funkcionalnosti implementiranih u razredima sustava. Izolirali smo neke glavne komponente i proveli nekoliko testova koji obuhvaćaju ključne funkcionalnosti.

Ispitni slučaj: testLoginNonExistentUser

- Ulazni podaci: Korisničko ime: "nonExistentUser" Lozinka: "wrongPassword123"
- Očekivani rezultati: "Invalid credentials", HTTP status 200 (OK)
- Dobiven rezultat: Prolaz ispitivanja

```

① 183     @Test
② 184     public void testLoginNonExistentUser() throws Exception {
③ 185         String invalidLoginJson = "{"
④ 186             + "\"username\": \"nonExistentUser\","
⑤ 187             + "\"password\": \"wrongPassword123\""
⑥ 188             + "}";
⑦ 189
⑧ 190         mockMvc.perform(post(urlTemplate:"/users/authenticate")
⑨ 191             .contentType(MediaType.APPLICATION_JSON)
⑩ 192             .content(invalidLoginJson))
⑪ 193             .andExpect(status().isOk())
⑫ 194             .andExpect(content().string(expectedContent:"Invalid credentials")));
⑬ 195     }
⑭ 196

```

Ispitni slučaj: testRegisterSuccessful

Provodimo testiranje registracije korisnika kroz REST API poziv prema metodi za registraciju plesača.

- Ulazni podaci: Korisničko ime: "validUsername" Ime: "Test" Prezime: "User" Email: "test.user@example.com" Lozinka: "securePassword123" Tip korisnika: "DANCER" Status OAuth-a: "true"
- Očekivani rezultati: "Registration successful!", HTTP status 200 (OK)
- Dobiven rezultat: Prolaz ispitivanja

```

① 107     @Test
② 108     public void testRegisterSuccessful() throws Exception {
③ 109         String validRegistrationJson = "{"
④ 110             + "\"username\": \"validUsername\","
⑤ 111             + "\"name\": \"Test\","
⑥ 112             + "\"surname\": \"User\","
⑦ 113             + "\"email\": \"test.user@example.com\","
⑧ 114             + "\"password\": \"securePassword123\","
⑨ 115             + "\"type\": \"DANCER\","
⑩ 116             + "\"finishedoauth\": \"true\""
⑪ 117             + "}";
⑫ 118
⑬ 119         mockMvc.perform(post(urlTemplate:"/users/registerdancer")
⑭ 120             .contentType(MediaType.APPLICATION_JSON)
⑮ 121             .content(validRegistrationJson))
⑯ 122             .andExpect(status().isOk()) // Expecting HTTP 200
⑰ 123             .andExpect(content().string(expectedContent:"Registration successful!"));
⑱ 124     }
⑲ 125

```

Ispitni slučaj: testGetByUsername

Provodimo testiranje pronalaženja korisnika na temelju korisničkog imena koristeći metodu get iz UserService. Ispitni slučaj:

- Ulazni podaci: Korisničko ime: "user123"
- Očekivani rezultati: Metoda get("user123") treba vratiti korisnika sa korisničkim imenom "user123".

- Dobiven rezultat: Prolaz ispitivanja

```

@Mock
private UserRepository userRepository;

@InjectMocks
private UserService userService;

public static final UserType USER_TYPE = new UserType(id:1, UserTypeEnum.DANCER.name());
public static final MyUser USER = new MyUser();
static {
    USER.setId(id:1);
    USER.setUsername(username:"user123");
}

```

```

> 170  @Test
@ 171  public void testGetByUsername() {
172      when(userRepository.findByUsername(username:"user123")).thenReturn(Optional.of(USER));
173      Optional<MyUser> result = userService.get(username:"user123");
174      assertTrue(result.isPresent());
175      assertEquals(USER, result.get());
176      MyUser fetchedUser = result.get();
177      assertEquals(expected:"user123", fetchedUser.getUsername());
178 }

```

Ispitni slučaj: testNonExistentFunctionality

- Ulazni podaci: HTTP GET metoda na nepostojeći URL (/non-existent-url)
- Očekivani rezultati: HTTP status: 403 Forbidden
- Dobiven rezultat: Prolaz ispitivanja

```

@Test
public void testNonExistentFunctionality() throws Exception {
    mockMvc.perform(get(urlTemplate:"/non-existent-url"))
        .andExpect(status().isForbidden());
}

```

Ispitni slučaji: testDancerModel, testDirectorModel

- Ulazni podaci: Ulazni podaci stvoreni setUp metodi, s njima želimo testirati ispravnost metoda za rad s DancerModel i DirectorModel klasama. Svi unaprijed pripremljeni podaci su ispravni.
- Očekivani rezultati: Svi podaci se poklapaju. Prolaz svih assert testova.

- Dobiven rezultat: Prolaz ispitivanja

```

@BeforeEach
public void setUp() {
    danceStyle1 = new Dance();
    danceStyle2 = new Dance();
    List<Dance> danceStyles = Arrays.asList(danceStyle1, danceStyle2);
    List<AuditionApplication> applications = Arrays.asList(new AuditionApplication());
    dancer = new Dancer(inactive:true, LocalDate.of(year:2025, month:1, dayOfMonth:31), danceStyles, applications);

    audition1 = new Audition();
    audition2 = new Audition();
    List<Audition> auditions = Arrays.asList(audition1, audition2);
    director = new Director(paid:true, LocalDate.of(year:2025, month:1, dayOfMonth:1), auditions);

}

@Test
public void testDancerModel() {
    assertTrue(dancer.isInactive(), message:"Dancer should be inactive");
    assertEquals(LocalDate.of(year:2025, month:1, dayOfMonth:31), dancer.getInactiveuntil(), message:"Inactive date s
    assertEquals(expected:2, dancer.getDances().size(), message:"Dancer should have 2 dance styles");
    assertFalse(dancer.getApplications().isEmpty(), message:"Dancer should have audition applications");
}

@Test
public void testDirectorModel() {
    assertTrue(director.isPaid(), message:"Director should be marked as paid");
    assertEquals(LocalDate.of(year:2025, month:1, dayOfMonth:1), director.getSubscription(), message:"Subscription da
    assertEquals(expected:2, director.getAudition().size(), message:"Director should have 2 auditions");
    assertFalse(director.getAudition().isEmpty(), message:"Director should have audition applications");
}

```

Ispitni slučaj: testAuditionApplicationModel

- Ulagni podaci: Ulagni podaci stvoreni na početku metode. Stvorena je audicija sa svim ispravnim parametrima.
- Očekivani rezultati: Svi podaci se poklapaju. Prolaz svih assert testova.
- Dobiven rezultat: Prolaz ispitivanja

```

@Test
public void testAuditionApplicationModel() {

    Audition audition = new Audition();
    Dancer dancer = new Dancer();
    LocalDateTime datetime = LocalDateTime.of(year:2025, month:1, dayOfMonth:31, hour:12, minute:30, second:0, nano:0);
    String status = "Pending";
    AuditionApplication application = new AuditionApplication(id:1, datetime, status, audition, dancer);

    assertEquals(expected:1, application.getId(), message:"Application ID should be 1");
    assertEquals(datetime, application.getDatetime(), message:"Datetime should be 2025-01-31 12:30:00");
    assertEquals(status, application.getStatus(), message:"Status should be 'Pending'");
    assertEquals(audition, application.getAudition(), message:"Audition should be the same as the one provided");
    assertEquals(dancer, application.getDancer(), message:"Dancer should be the same as the one provided");
}

```

Ispitni slučaj: testAdminSubscriptionPrice

- Ulagni podaci: Stvoren novi admin i postavljena cijena na 100.
- Očekivani rezultati: Cijena je ispravno postavljena. Prolaz assert testa.
- Dobiven rezultat: Prolaz ispitivanja

```

@Test
public void testAdminSubscriptionPrice() {
    Long expectedPrice = 100L;
    Admin admin = new Admin(expectedPrice);

    Long actualPrice = admin.getSubscriptionprice();
    assertEquals(expectedPrice, actualPrice, message:"Subscription price should match");
}

```

Testiranje s nvm test:

```
2025-01-21T15:36:29.186+01:00 INFO 22436 --- [skydancers] [ma  
in] h.f.s.SkydancersApplicationTests : Started SkydancersApplicat  
ionTests in 9.837 seconds (process running for 11.329)  
[INFO] Tests run: 7, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 11  
.87 s -- in hr.fer.skydancers.SkydancersApplicationTests  
[INFO]  
[INFO] Results:  
[INFO]  
[INFO] Tests run: 8, Failures: 0, Errors: 0, Skipped: 0  
[INFO]  
[INFO] -----  
[INFO] BUILD SUCCESS  
[INFO] -----
```

Ispitivanje sustava

Ispitivanje sustava provedeno je pomoću Selenium IDE alata koji omogućava automatsko snimanje korisničkih akcija u pregledniku te njihovo ponavljanje za potrebe testiranja funkcionalnosti aplikacije. Ispitni slučajevi su dizajnirani kako bi obuhvatili tipične scenarije, kao i rubne uvjete (npr. unos neispravnih podataka), kako bi se testirala reakcija sustava na neočekivane ulaze i greške.

Registracija plesnog direktora

1. Ulazi:

- korisničko ime ("direktor"), ime i prezime ("Ivan Horvat"), email adresa ("email@gmail.com") i lozinka ("TestTest1"), odabrana opcija "plesni direktor"

2. Koraci ispitivanja:

- Otvoriti aplikaciju.
- Unijeti podatke za registraciju.
- Odabrati opciju plesnog direktora.
- Kliknuti na "Registracija".

3. Očekivani izlaz:

- Uspješna registracija.

4. Dobiveni izlaz:

Project: SkyDancers

Command	Target	Value
✓ open	http://localhost:3000/	
✓ set window size	784x816	
✓ click	css=button	
✓ click	css=submit:nth-child(2)	
✓ click	css=input:nth-child(1) > input	
✓ type	css=input:nth-child(1) > input	direktor
✓ click	css=input:nth-child(2) > input	
✓ type	css=input:nth-child(2) > input	Ivan
✓ click	css=input:nth-child(3) > input	
✓ type	css=input:nth-child(3) > input	Horvat
✓ click	css=input:nth-child(4) > input	
✓ click	css=input:nth-child(4) > input	
✓ type	css=input:nth-child(4) > input	email@gmail.com
✓ click	css=input:nth-child(5) > input	
✓ type	css=input:nth-child(5) > input	TestTest1

Command: open //

Target: http://localhost:3000/

Value:

Description:

Odjava i ponovna prijava korisnika

1. Ulazi:

- korisničko ime ("test") i lozinka ("TestTest1")

2. Koraci ispitivanja:

- Otvoriti aplikaciju.
- Kliknuti na gumb za odjavu.
- Odabratи opciju plesnog direktora.
- Kliknuti na gumb za prijavu.

3. Očekivani izlaz:

- Odjava iz korisničkog računa te uspješna ponovna prijava.

4. Dobiveni izlaz:

Project: SkyDancers

Tests	Command	Target	Value
✓ odjava-prijava	1 ✓ open	http://localhost:3000/	
✓ registracija-plesniDirektor	2 ✓ set window size	784x816	
✓ uređivanje profila	3 ✓ click	css=logout > button	
	4 ✓ click	css=button	
	5 ✓ click	css= input:nth-child(1) > input	
	6 ✓ type	css= input:nth-child(1) > input	test
	7 ✓ click	css= input:nth-child(2) > input	
	8 ✓ type	css= input:nth-child(2) > input	TestTest1
	9 ✓ click	css= submit-container > submit:nth-child(1)	

Command: //

Target:

Value:

Description:

Log Reference

Uređivanje profila

1. Ulaz:

- broj godina korisnika("25")

2. Koraci ispitivanja:

- Otvoriti aplikaciju.
- Kliknuti na gumb "Moj Profil".
- Kliknuti na gumb "Uredi Profil".
- Ažurirati dob (broj godina).
- Kliknuti na gumb "Spremi".

3. Očekivani izlaz:

- Uspješno ažuriranje profila.

4. Dobiveni izlaz:

The screenshot shows a software interface for managing tests. At the top, there's a toolbar with icons for file operations like Open, Save, and Print. Below the toolbar is a search bar labeled 'Search tests...' and a button 'Run current test Ctrl+R'. The main area is titled 'Project: SkyDancers' and contains a table of test cases:

	Command	Target	Value
1	✓ open	http://localhost:3000/	
2	✓ set window size	784x816	
3	✓ click	css= login > button	
4	✓ click	css=button:nth-child(10)	
5	✓ click	name=age	
6	✓ type	name=age	26
7	✓ click	css= buttons:nth-child(6)	
8	✓ assert alert	Profil uspješno ažuriran!	

Below the table, there are input fields for 'Command', 'Target', 'Value', and 'Description', each with a dropdown arrow and a magnifying glass icon. At the bottom of the interface, there are tabs for 'Log' and 'Reference'.

Slanje poruke drugom korisniku

1. Ulaz:

- pretraživanje traženog korisnika (upisano samo di te odabran korisnik direktor), poslana poruka (pozdrav!)

2. Koraci ispitivanja:

- Otvoriti aplikaciju.
- Kliknuti na gumb "Chat s drugim korisnicima".
- U tražilicu upisano korisničko ime osobe kojoj želimo poslati poruku.
- Odabran korisnik.
- Napisana poruka.
- Kliknut gumb "Pošalji".

3. Očekivani izlaz:

- Uspješno poslana poruka

4. Dobiveni izlaz:

The screenshot shows a test recording interface for a project named "SkyDancers". The left sidebar lists tests: "odjava-prijava", "registracija-plesniDirektor", "slanje poruke*", and "uredivanje profila". The main area displays a sequence of recorded commands:

	Command	Target	Value
1	✓ open	http://localhost:3000/	
2	✓ set window size	784x816	
3	✓ click	css= navigation-button:nth-child(2)	
4	✓ click	css= chat-input	
5	✓ type	css= chat-input	di
6	✓ click	css= user-item:nth-child(2)	
7	✓ click	css= chat-form > .chat-input	
8	✓ type	css= chat-form > .chat-input	pozdravl
9	✓ click	css= message-button	
10	✓ mouse over	css= message-button	
11	✓ mouse out	css= message-button	

Below the table are input fields for "Command", "Target", "Value", and "Description", each with a clear button.

Unos neispravne lozinke

1. Ulaz:

- korisničko ime ("direktor") i neispravna lozinka ("kriva lozinka")

2. Koraci ispitivanja:

- Otvoriti aplikaciju.
- Upisati korisničko ime i neispravnu lozinku.
- Kliknuti na gumb "Prijava".

3. Očekivani izlaz:

- Obavijest o neispravnim podacima

4. Dobiveni izlaz:

The screenshot shows the Selenium IDE interface with the project set to 'SkyDancers'. A test case titled 'unos neispravne lozinke*' is selected. The test steps are listed in the table below:

	Command	Target	Value
1	✓ open	http://localhost:3000/	
2	✓ set window size	720x816	
3	✓ click	css=button	
4	✓ click	css=input:nth-child(1) > input	
5	✓ type	css=input:nth-child(1) > input	direktor
6	✓ click	css=input:nth-child(2) > input	
7	✓ type	css=input:nth-child(2) > input	krivalozinka
8	✓ click	css=submit-container > submit:nth-child(1)	
9	✓ assert alert	Pogrešni podaci!	

Below the table, there are four input fields: 'Command' (dropdown), 'Target' (with search icons), 'Value' (with search icons), and 'Description'.

Kreiranje audicije i pokušaj stavljanja datuma roka prijave u prošlost

1. Ulaz:

- datum i vrijeme audicije ("2025-01-30 T 14.33"), pokušaj stavljanja roka prijave na ("2025-01-10 T 14.33"), nakon neuspjeha stavljanje na ispravan datum ("2025-01-23 T 00.33"), lokacija ("zagreb"), opis ("ples"), plaća ("100E") i odabir stila ("Balet")

2. Koraci ispitivanja:

- Otvoriti aplikaciju.
- Označiti datum i vrijeme audicije.
- Označiti datum i vrijeme prijave na audiciju, provo odabrati datum u prošlosti, nakon dojave greške ispraviti.
- Ispuniti ostatak formulara.
- Kliknuti na gumb za objavu audicije.

3. Očekivani izlaz:

- Greška prilikom odabira datuma u prošlosti, nakon ispravka očekujemo uspješnu objavu audicije.

4. Dobiveni izlaz:

Project: SkyDancers*

Executing: ✓ kreiranje audicije*

http://localhost:3000/

Command	Target	Value
1 ✓ open	http://localhost:3000/	
2 ✓ set window size	720x816	
3 ✓ click	css=navigation-button:nth-child(1)	
4 ✓ click	name=datetime	
5 ✓ type	name=datetime	2025-01-30T14:33
6 ✓ click	css=create-audition-container	
7 ✓ click	name=deadline	
8 ✓ assert alert	Rok prijave ne može biti u prošlosti.	
9 ✓ click	name=deadline	
10 ✓ type	name=deadline	2025-01-23T00:33
11 ✓ click	css=div:nth-child(1)	
12 ✓ click	name=location	
13 ✓ type	name=location	zagreb
14 ✓ click	css=textarea	
15 ✓ type	css=textarea	ples

Command

Target

Value

Description

Runs: 1 Failures: 0

7. Tehnologije za implementaciju aplikacije

Korištene tehnologije i alati

Programski jezici

- Java (verzija 19.0.2)
 - Java je objektno orijentirani programski jezik visoke razine dizajniran da ima što manje implementacijskih ovisnosti. Zamišljen je kao jezik opće namjene koji omogućava da se programi pokreću na bilo kojoj platformi.
 - U aplikaciji Javu koristimo kao jezik u kojem razvijamo backend pomoću Spring Boota.
- JavaScript (verzija 1.5)
 - JavaScript je programski jezik namijenjen za razvoj Web tehnologija. Web preglednici imaju ugrađene JavaScript pogone (engl. JavaScript engine) koji pokreću klijentski kod.
 - U aplikaciji JavaScript koristimo za razvoj korisničke (frontend) strane aplikacije pomoću React JS biblioteke.

Radni okviri i biblioteke

- Spring Boot (verzija 3.1.5)
 - Spring Boot je Javin razvojni okvir (engl. Framework) za razvoj skalabilnih aplikacija. Cilj ovog razvojnog okvira je da pojednostavi konfiguraciju i postavljanje projekta, te omogući veći fokus na pisanje koda programerima.
 - U aplikaciji Spring Boot koristimo kao server za web stranicu.
- React JS (verzija 19)
 - React je JavaScript biblioteka za izradu korisničkih sučelja. On nam dozvoljava da stvaramo komponente za višekratnu uporabu.
 - U aplikaciji React koristimo kao frontend, što omogućava da imamo responzivni dizajn.
- Node JS (verzija 22.12.0)

- Node JS je višeplatformsko JavaScript radno okruženje namijenjeno za razvoj koda koji koristi serverske komponente.
- U aplikaciji Node JS koristimo za alate komandne linije i za 'server-side' skriptiranje.

Baza podataka

- PostgreSQL (verzija 17)
 - PostgreSQL je sustav za upravljanje bazama podataka koji podržava relacijske i nerelacijske modele baza.
 - U aplikaciji PostgreSQL koristimo kao bazu podataka.

Razvojni alati

- Git (verzija 2.44.0) i GitHub
 - Git i GitHub su programi koji služe za upravljanje, spremanje i dijeljenje verzija projekta. Git je glavni alat za praćenje verzija datoteka i dokumenata, a GitHub je platforma koja služi za spremanje i dijeljenje koda.
 - Naša aplikacija se također nalazi na GitHubu i koristi Git za praćenje verzija projekta.
- Visual Studio Code (verzija 1.96.2) i IntelliJ (verzija 2024.3.1.1)
 - Visual Studio Code i IntelliJ su integrirana razvojna okruženja u kojima se razvijaju aplikacije.
 - U aplikaciji VS Code i IntelliJ koristimo za razvoj frontenda, backenda i pisanje dokumentacije.

Alati za ispitivanje

- Selenium (verzija 4.28.0)
 - Selenium je alat za testiranje web-baziranih aplikacija kako bi se provjerilo rade li one prema očekivanjima. Smatran je jednim od najboljih alata za automatsku evaluaciju i nije ovisan o platformi na kojoj se razvija.
 - U aplikaciji Selenium koristimo kao alat za ispitivanje većih dijelova aplikacije (ispitivanje sustava).

Cloud platforma

- Render
 - Render je ujedinjeni oblak (engl. unified cloud) za pokretanje aplikacija i web stranica.
 - Render koristimo za hosting naše aplikacije.

Ostalo

- Stripe
 - Stripe je platforma i alat koji pruža usluge plaćanja.
 - U aplikaciji Stripe koristimo kao platformu za plaćanje godišnje članarine plesnim direktorima.
- Firebase (verzija 11.1.0)
 - Firebase je skup backend cloud servisa i alata za mobilne i web aplikacije. Pruža usluge baze podataka u stvarnom vremenu, autentifikaciju, hosting, poruke, ...
 - U aplikaciji Firebase koristimo kao servis za spremanje i slanje poruka.

8. Upute za puštanje u pogon

1. Instalacija

Ovdje treba navesti korake potrebne za instalaciju svih potrebnih komponenti:

- **Preduvjeti:**
 - Visual Studio Code, Eclipse, Intelij ili slični IDE. (Upute su za Eclipse i VS code)
 - Java 19, React 19, Node 22
 - Render
- **Preuzimanje:** Upute za preuzimanje izvornog koda

```
git clone https://github.com/mvukoj/SkyDancers.git
```

Instalacija ovisnosti: Upute za instaliranje ovisnosti.

Upute za backend:

Prilikom učitavanja projekta u Eclipsu, sve ovisnosti će se automatski preuzeti. Ako se to ne dogodi potrebno je uraditi desni klik na projekt, Maven > Update project.

Upute za frontend:

U VSC je potrebno u terminalu upisati sljedeću naredbu da bi se instalirale sve ovisnosti:

```
npm install
```

2. Postavke

Detaljne upute za konfiguraciju aplikacije:

- **Konfiguracijska datoteka** `application.properties` se nalazi u `src/main/resources`. U njoj se nalazi konfiguracija aplikacije odnosno potrebni API ključevi. Potrebno je `.env` datoteku prilagoditi podatke vlastitim potrebama da bi aplikacija uspješno radila. Potrebno je unijeti sljedeće varijable:
 - SPRING_DATASOURCE_PASSWORD, SPRING_DATASOURCE_URL, SPRING_DATASOURCE_USERNAME (vezane uz bazu podataka)
 - SPRING_JWT_SECRET (nasumični JWT secret)
 - SPRING_OAUTH_GITHUB_ID, SPRING_OAUTH_GITHUB_SECRET (GitHub OAuth postavke)
 - SPRING_STRIPE_SECRETKEY (tajni ključ sa Stripe servisa)
 - SPRING_MAIL_API, SPRING_MAIL_USERNAME (podaci o mail API-ju)

Primjer: `SPRING_FRONTEND_URL="https://skydancers.onrender.com"`

- **Konfiguracijska datoteka** `.env` vezana uz frontend. U njoj se nalazi konfiguracija aplikacije odnosno potrebni API ključevi. Potrebno je `.env` datoteku prilagoditi podatke vlastitim potrebama da bi aplikacija uspješno radila. Potrebno je unijeti sljedeće varijable:
 - REACT_APP_API_KEY, REACT_APP_AUTH_DOMAIN, REACT_APP_PROJECT_ID, REACT_APP_STORAGE_BUCKET, REACT_APP_MESSAGING_SENDER_ID, REACT_APP_APP_ID,

- REACT_APP_MEASUREMENT_ID, (sve prije navedene su vezane uz Firebase API) te
- REACT_APP_BACKEND_URL

Primjer: REACT_APP_BACKEND_URL="<https://skydancers-back.onrender.com>"

3. Pokretanje aplikacije

- **Pokretanje aplikacije lokalno:**

- Unutar direktorija src/main/java/hr/fer/skydancers treba pronaći klasu SkydancersApplication te modificirati njezinu Run konfiguraciju. Potrebno je unijeti sve "Environment variables" (vidljive na slici u dijelu "Postavke") kako bi ispravno funkcionirala aplikacija. Nakon toga je backend moguće pokrenuti preko tipke "Run" na istoj klasi. Backend je po standardnim postavkama pokrenut na portu 8080. Ako pristupimo npr. <http://localhost:8080> trebali bismo dobiti 403 Forbidden error koji označava da aplikacija radi.
- U terminalu VSC je potrebno pokrenuti naredbu `npm start` koja vraća poruku sa adresom frontenda. Po standardnim postavkama je to <http://localhost:3000>. Pristupivši toj stranici bi trebali vidjeti početnu stranicu naše aplikacije. Ako želimo aplikaciju pokrenuti za produkcijsko okruženje treba izvršiti naredbu `npm run build` koja potom kreira frontend aplikaciju spremnu za produkciju. Potrebno ju je poslužiti bilo kakvim serverom (npr. Express, Python...)

4. Upute za administratore

Smjernice za administratore aplikacije nakon puštanja u pogon:

- **Pristup administratorskom sučelju:**

- Admin ima posebne podatke (username i lozinku) i prijavljuje se kao svi ostali korisnici. Admin ima posebne mogućnosti koje su vidljive samo njemu.

- **Redovito održavanje:**

- Arhiviranje baze podataka.
 - Moguće na Renderu gdje se nalazi baza podataka. <https://dashboard.render.com/>
- Pregled logova.
 - Moguće na Renderu gdje se nalaze sve komponente aplikacije (Frontend, backend i baza podataka) i svaka ima zasebne logove. <https://dashboard.render.com/>
- Ažuriranje aplikacije.
 - Aplikacija se automatski ažurira na svaki novi commit koji stvara izmjene.

5. Puštanje aplikacije u pogon (Render deploy)

Render je popularna cloud platforma za jednostavno smještanje aplikacija.

- **Puštanje u pogon backenda:**

- **Priprema repozitorija:**

- Osigurajte da vaš projekt ima datoteku Dockerfile za konfiguraciju deploya.
- Primjer Dockerfile:

```
FROM maven:3.8.1-openjdk-17 AS build  
WORKDIR /app  
COPY . /app  
RUN mvn clean package -DskipTests  
FROM openjdk:17-jdk-slim  
WORKDIR /app  
COPY --from=build /app/target/*.jar app.jar  
EXPOSE 8080  
ENTRYPOINT ["java", "-jar", "app.jar"]`
```

- **Postavljanje na Render:**

- Prijavite se na [Render](#).
- Kreirajte novi **Web Service** i povežite ga s vašim GitHub repozitorijem.
- Dodajte environment varijable koje su potrebne (gore navedene).
- Pustite u pogon

- **Pokretanje aplikacije:**

Render će automatski preuzeti repozitorij, instalirati ovisnosti i pokrenuti aplikaciju. Nakon deploya, aplikaciji možete pristupiti putem generiranog URL-a (npr. <https://skydancers-back.onrender.com>).

- **Puštanje u pogon frontenda:**

- **Postavljanje na Render:**

- Prijavite se na [Render](#).
- Kreirajte novi **Static Page** i povežite ga s vašim GitHub repozitorijem.
- Dodajte environment varijable koje su potrebne (gore navedene).
- Dodajte preusmjerivanje sa svih ruta na index.html file kako bi React Router pravilno radio
- Pustite u pogon

- **Pokretanje aplikacije:**

Render će automatski preuzeti repozitorij, instalirati ovisnosti i pokrenuti aplikaciju. Nakon deploya, aplikaciji možete pristupiti putem generiranog URL-a (npr. <https://skydancers.onrender.com>).

Opis pristupa aplikaciji na javnom poslužitelju

Pristup aplikaciji

Naša aplikacija je deployana na sustavu Render. Upute za korištenje:

- potrebno je učitati stranicu na kojoj je postavljen backend koji zatim prikazuje 403 Forbidden error od Springa, što označava da je funkcionalan.
- zatim učitati stranicu na kojoj je frontend i koristiti aplikaciju

Linkovi:

-backend: <https://skydancers-back.onrender.com/>

-frontend: <https://skydancers.onrender.com/>

9. Zaključak i budući rad

Cilj projekta je bio razvoj online platforme za povezivanje plesača i plesnih direktora, pomoću koje bi se plesnim direktorima olakšala potraga za plesačima, a plesačima pregled i prijava audicija. Trajanje projekta je bilo 15 tjedana, točnije od 9. 10. 2024. do 16. 1. 2025. U tom razdoblju izrađene su sve tražene funkcionalnosti. Ostvarivanje projekta podijeljeno je u dvije faze.

U prvih 6 tjedana se okupio tim za izradu aplikacije, napravljen je plan rada, podijeljene su uloge, određeni su funkcionalni i nefunkcionalni zahtjevi, odabrane su tehnologije s kojima ćemo raditi. Nakon upoznavanja s odabranim tehnologijama, osmišljavanja izgleda i rada aplikacije te razrade dokumentacije, započete su implementacije funkcionalnosti te je na kraju prve faze (kraju prvog ciklusa) prva verzija aplikacije s osnovnim funkcionalnostima deployana na sustavu Render.

U drugoj fazi radilo se na implementaciji svih preostalih funkcionalnosti i usavršavanje implementacije popraćeno s radom na dokumentaciji. Između članova tima već je bila uspostavljena dobra komunikacija, koja se pretežito odvijala putem discorda. Napravljeni su UML dijagrami stanja, aktivnosti, komponenti i razmještaja te provedeno ispitivanje rada aplikacije. Krajem drugog ciklusa je aplikacija završena i puštena u pogon.

Moguće proširenje aplikacije bio bi razvoj mobilne aplikacije te dodavanje drugih vanjskih servisa za plaćanje.

Najveći izazov u izradi projekta bilo je upoznavanje s novim tehnologijama poput Reacta za izradu frontenda, Java Springa za izradu backenda, Astaha za izradu UML dijagrama i Githuba za upravljanje projektom. Rad s ovim alatima zahtijevao je dodatno vrijeme za učenje i prilagodbu. Uspješno smo savladali njihove koncepte i primijenili ih u razvoju projekta. Osim tehničkog znanja, stekli smo i iskustvo rada u timu. Učinkovita komunikacija, koordinacija i podjela uloga pokazale su se važnim za realizaciju zadatka. Sudjelovanje na ovom projektu dao nam je uvid u rad na kompleksnijim projektima, te će nam ovo iskustvo sigurno koristit u budućim projektima, bilo tijekom školovanja ili na radnim mjestima.

A. Popis literature

1. Programsко inženjerstvo, FER ZEMRIS, <http://www.fer.hr/predmet/proinz>
2. The Unified Modeling Language, <https://www.uml-diagrams.org>

3. Astah Community, <http://astah.net/editions/uml-new>
4. Spring Boot Documentation Overview, <https://docs.spring.io/spring-boot/documentation.html>
5. React Reference Overview, <https://react.dev/reference/react>
6. Selenium getting started, https://www.selenium.dev/documentation/webdriver/getting_started
7. PostgreSQL documentation, <https://www.postgresql.org/docs>
8. Firebase Learn the fundamentals, <https://firebase.google.com/docs>
9. OAuth 2.0, <https://oauth.net/2>
10. Stripe docs, <https://docs.stripe.com>
11. Node.js API, <https://nodejs.org/api/all.html>

A. Dnevnik promjena dokumentacije

Rev.	Opis promjene/dodataka	Autori	Datum
0.1	Napravljen predložak	Mario Vukoja	25.10.2024.
0.2	Dodana analiza zahtjeva	Leonardo Klišanić	31.10.2024.
0.3	Dodani obrasci uporabe	Fani Jurak	2.11.2024.
0.4	Dodani sekvencijski dijagrami	Fani Jurak	6.11.2024.
0.5	Dodani opis projekta	Leonardo Klišanić	9.11.2024.
0.6	Dodan opis baze podataka	Katarina Bubalo	10.11.2024.
0.6.1	Dodan opis arhitekture sustava	Leonardo Klišanić	11.11.2024.
0.7	Dodan prikaz aktivnosti grupe	Leonardo Klišanić	13.11.2024.
0.8	Odrađeni ostali dijagrami	Leonardo Klišanić	10.1.2025.
0.9	Napravljena sekcija zaključak i budući rad	Fani Jurak	10.1.2025.
1.0	Napravljeno ispitivanje programskog rješenja	Fani Jurak	19.1.2025.
1.1	Izmijenjena baza podataka zbog usklađivanja	Katarina Bubalo	20.1.2025.
1.2	Dodana literatura i glavni izazovi + zaključak	Leonardo Klišanić	22.1.2025.

B. Prikaz aktivnosti grupe

Dnevnik sastajanja

1. sastanak

- Datum: 9. listopada 2024.
 - Prisustvovali: B.Glavina, K.Bubalo, F.Jurak, L.Malešević, A.Šimić, L.Klišanić, M.Vukoja
 - Teme sastanka:
- Upoznavanje tima

- Diskusija o temi i njezino proučavanje
 - Postavljanje GitHub repozitorija
-

2. sastanak

- Datum: 17. listopada 2024.
- Prisustvovali: B.Glavina, K.Bubalo, F.Jurak, L.Malešević, A.Šimić, L.Klišanić, M.Vukoja
- Teme sastanka:

- Određivanje funkcijskih i nefunkcijskih zahtjeva
 - Podjela tima po zadacima
-

3. sastanak

- Datum: 24. listopada 2024.
- Prisustvovali: B.Glavina, K.Bubalo, F.Jurak, L.Malešević, A.Šimić, L.Klišanić, M.Vukoja
- Teme sastanka:

- Prezentacija funkcijskih i nefunkcijskih zahtjeva
 - Rasprava o koordinaciji i početku rada
-

4. sastanak

- Datum: 31. listopada 2024.
- Prisustvovali: B.Glavina, K.Bubalo, F.Jurak, L.Malešević, A.Šimić, L.Klišanić, M.Vukoja
- Teme sastanka:

- Pregled napravljenog
 - Rasprava o arhitekturi sustava, bazi podataka i dokumentaciji
 - Određena platforma za hosting aplikacije
-

5. sastanak

- Datum: 7. studenog 2024.
- Prisustvovali: B.Glavina, K.Bubalo, F.Jurak, L.Malešević, A.Šimić, L.Klišanić, M.Vukoja
- Teme sastanka:

- Pregled napravljenog
 - Dodatna rasprava o arhitekturi sustava
-

6. sastanak

- Datum: 29. studenog 2024.
- Prisustvovali: B.Glavina, K.Bubalo, F.Jurak, L.Malešević, A.Šimić, L.Klišanić, M.Vukoja
- Teme sastanka:

- Pregled napravljenog
- Određen cjelokupni dizajn aplikacije

7. sastanak

- Datum: 9. prosinca 2024.
- Prisustvovali: B.Glavina, K.Bubalo, F.Jurak, L.Malešević, A.Šimić, L.Klišanić, M.Vukoja
- Teme sastanka:

- Pregled napravljenog
-

8. sastanak

- Datum: 16. prosinca 2024.
- Prisustvovali: B.Glavina, K.Bubalo, F.Jurak, L.Malešević, A.Šimić, L.Klišanić, M.Vukoja
- Teme sastanka:

- Pregled napravljenog
-

9. sastanak

- Datum: 4. siječnja 2024.
- Prisustvovali: B.Glavina, K.Bubalo, F.Jurak, L.Malešević, A.Šimić, L.Klišanić, M.Vukoja
- Teme sastanka:

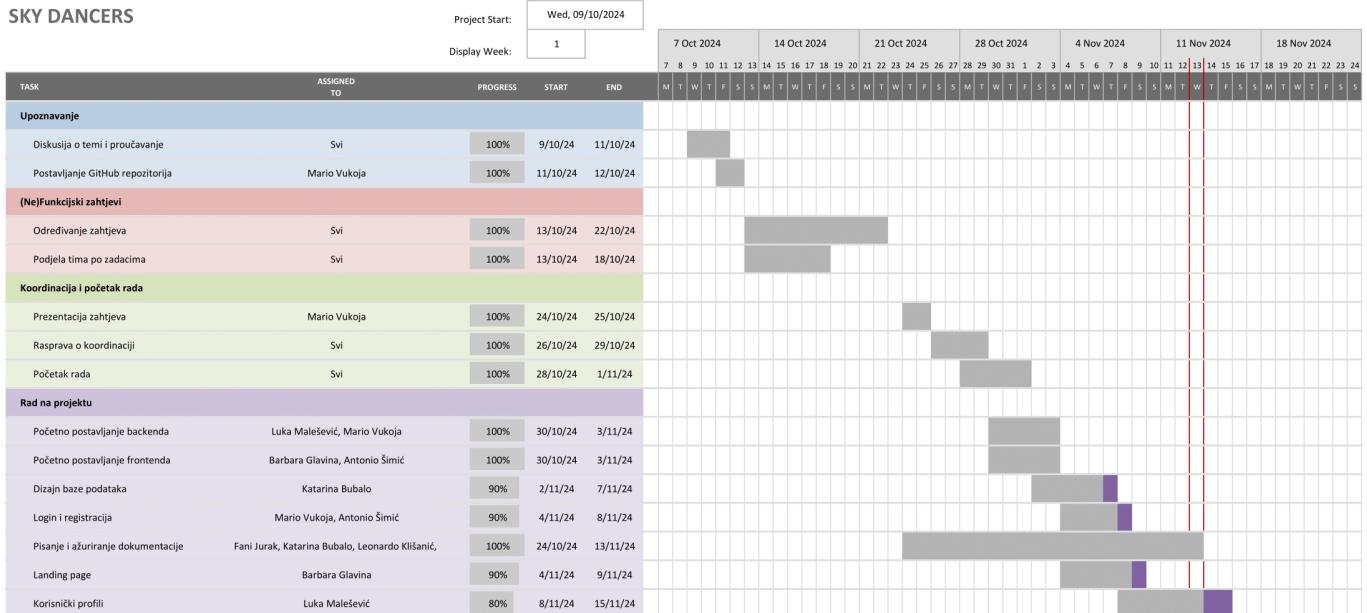
- Pregled napravljenog
 - Dogovor oko dijagrama u dokumentaciji
-

10. sastanak

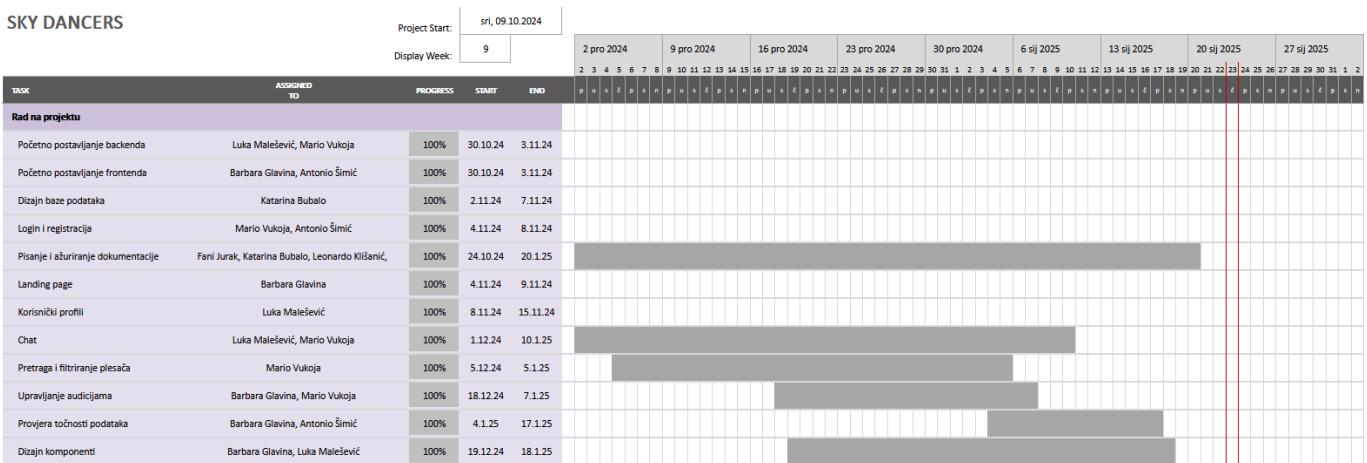
- Datum: 14. siječnja 2024.
- Prisustvovali: B.Glavina, K.Bubalo, F.Jurak, L.Malešević, A.Šimić, L.Klišanić, M.Vukoja
- Teme sastanka:

- Pregled napravljenog
 - Provjera funkcionalnosti i testiranje
-

Plan rada



Prvi dio



Drugi dio

Tablica aktivnosti

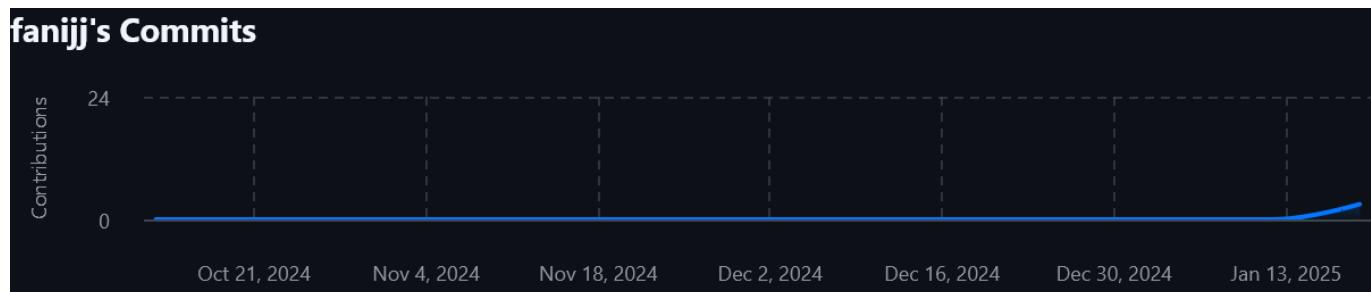
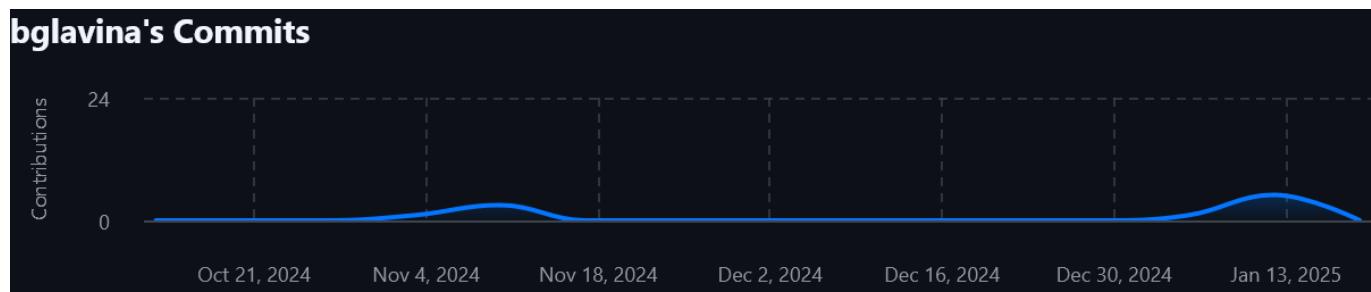
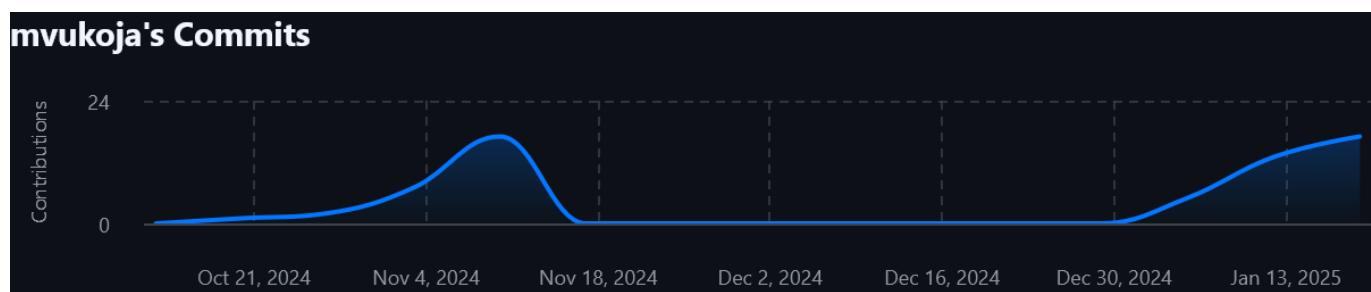
	Barbara Glavina	Katarina Bubalo	Fani Jurak	Luka Malešević	Antonio Šimić	Leonardo Klišanić	Mario Vukoja
Upravljanje projektom	15 sati	15 sati	13 sati	15 sati	15 sati	13 sati	16 sati
Opis projektnog zadatka	2 sata	2 sata	2 sata	2 sata	2 sata	9 sati	9 sati

	Barbara Glavina	Katarina Bubalo	Fani Jurak	Luka Malešević	Antonio Šimić	Leonardo Klišanić	Mario Vukoja
Funkcionalni zahtjevi	1 sat	1 sat	2 sata	1 sat	1 sat	4 sata	15 sati
Opis pojedinih obrazaca	0 sati	0 sati	5 sati	0 sati	0 sati	4 sata	1 sat
Dijagram obrazaca	0 sati	0 sati	4 sata	0 sati	0 sati	1 sat	1 sat
Sekvencijski dijagrami	0 sati	0 sati	5 sati	0 sati	0 sati	0 sati	1 sat
Opis ostalih zahtjeva	1 sat	1 sat	2 sata	1 sat	1 sat	2 sata	2 sata
Arhitektura i dizajn sustava	0 sati	5 sati	1 sat	3 sata	0 sati	1 sat	10 sati
Baza podataka	0 sati	10 sati	0 sati	0 sati	0 sati	0 sati	5 sati
Dijagram razreda	0 sati	0 sati	0 sati	6 sati	0 sati	0 sati	2 sata
Dijagram stanja	0 sati	0 sati	1 sat	0 sati	0 sati	5 sati	0 sati
Dijagram aktivnosti	0 sati	0 sati	1 sat	0 sati	0 sati	3 sata	0 sati
Dijagram komponenti	0 sati	0 sati	1 sat	0 sati	0 sati	4 sata	1 sat
Dijagram razmještaja	0 sati	0 sati	1 sat	0 sati	0 sati	3 sata	2 sata
Tehnologije za implementaciju aplikacije	0 sati	1 sat	2 sata	0 sati	0 sati	3 sata	12 sati
Ispitivanje programskog rješenja	0 sati	0 sati	6 sati	0 sati	0 sati	0 sati	5 sati
Upute za puštanje u pogon	0 sati	0 sati	0 sati	0 sati	0 sati	0 sati	2 sata
Dnevnik sastajanja	0 sati	0 sati	1 sat	0 sati	0 sati	1 sat	1 sat
Zaključak i budući rad	0 sati	0 sati	3 sata	0 sati	0 sati	0 sati	0 sati
Popis literature	0 sati	0 sati	0 sati	0 sati	0 sati	0 sati	0 sati
Izradu aplikacije	30 sati	0 sati	0 sati	18 sati	35 sati	1 sat	50 sati
Izrada početne stranice	10 sati	0 sati	0 sati	0 sati	5 sati	0 sati	10 sati
Izrada baze podataka	1 sat	8 sati	0 sati	0 sati	1 sat	0 sati	5 sati

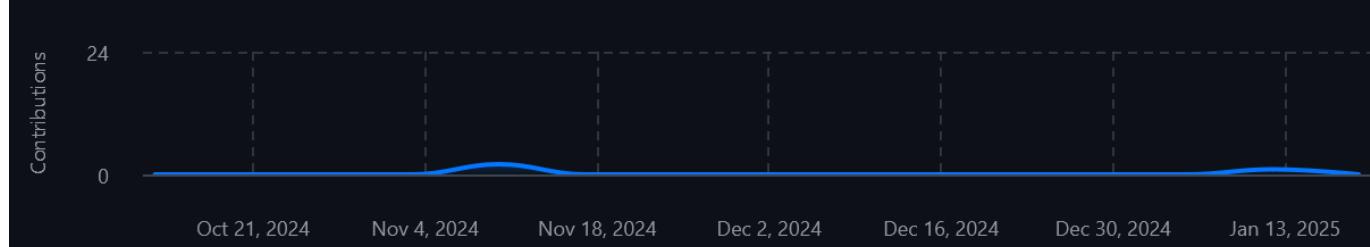
	Barbara Glavina	Katarina Bubalo	Fani Jurak	Luka Malešević	Antonio Šimić	Leonardo Klišanić	Mario Vukoja
Spajanje s bazom podataka	2 sata	2 sata	2 sata	2 sata	4 sata	0 sati	5 sati
Izrada prezentacije	0 sati	6 sati	1 sat	0 sati	0 sati	0 sati	1 sat

Dijagram pregleda promjena

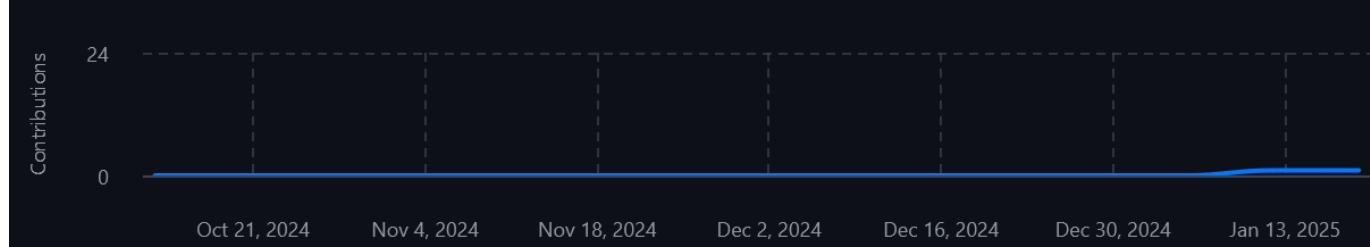
 Commits over time



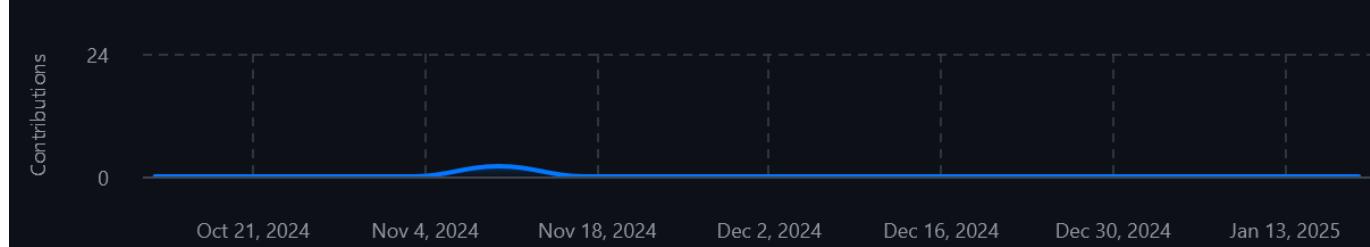
malesevicluka's Commits



katarinabubalo's Commits



LeonardoKlisanic's Commits



Ključni izazovi i rješenja

- Jedan od glavnih izazova je bio razvoj s nepoznatim razvojnim sučeljima i servisima koji usporavaju napredak, ali je sveukupno pozitivan izazov jer se puno može naučiti radeći u nepoznatom. Rješenje na ovaj izazov nije komplikirano, ali je zahtjevno; trebalo je samo kontinuirano raditi.
- Drugi izazov je bio raspodjela poslova koju smo dobro i brzo odradili, pa nam nije stvorila veliki problem.
- Treći izazov je bilo spajanje različitih komponenti u jednu povezanu cjelinu. Ovaj izazov se pojavljuje u svakom razvoju, ali odradili smo ga bez velikih poteškoća.

Ovaj projekt je bio dobar pogled u stvarna radna okruženja i emulirao je određene probleme koji se mogu naći u njima, od timskog rada do projekta. Pomoću ovog projekta smo si uvećali iskustva koja će nam biti potrebna na budućim poslovima, ali i drugim obavezama koje ćemo susretati.