Razvoj sustava e-izlaznice

Finderle, Petar

Sveučilište Jurja Dobrile u Puli
Odjel za informacijsko-komunikacijske tehnologije

PETAR FINDERLE

RAZVOJ SUSTAVA E-IZLAZNICE

Završni rad

Pula, 2016.

Sveučilište Jurja Dobrile u Puli
Odjel za informacijsko-komunikacijske tehnologije

PETAR FINDERLE

RAZVOJ SUSTAVA E-IZLAZNICE

Završni rad

JMBAG: 0036377606, izvanredni student
Studijski smjer: Informatika

Predmet: Projektiranje IS
Mentor: Prof. dr. sc. Vanja Bevanda

Pula, svibanj, 2016.

IZJAVA O AKADEMSKOJ ČESTITOSTI

Ja, dolje potpisani Petar Finderle, kandidat za prvostupnika informatike ovime izjavljujem da je ovaj Završni rad rezultat isključivo mojega vlastitog rada, da se temelji na mojim istraživanjima te da se oslanja na objavljenu literaturu kao što to pokazuju korištene bilješke i bibliografija. Izjavljujem da niti jedan dio Završnog rada nije napisan na nedozvoljen način, odnosno da je prepisan iz kojega necitiranog rada, te da ikoji dio rada krši bilo čija autorska prava. Izjavljujem, također, da nijedan dio rada nije iskorišten za koji drugi rad pri bilo kojoj drugoj visokoškolskoj, znanstvenoj ili radnoj ustanovi.

Student

_____

U Puli, 25. svibnja, 2016. godine

IZJAVA

o korištenju autorskog djela

Ja, Petar Finderle dajem odobrenje Sveučilištu Jurja Dobrile u Puli, kao nositelju prava iskorištavanja, da moj završni rad pod nazivom "Razvoj sustava e-izlaznice" koristi na način da gore navedeno autorsko djelo, kao cjeloviti tekst trajno objavi u javnoj internetskoj bazi Sveučilišne knjižnice Sveučilišta Jurja Dobrile u Puli te kopira u javnu internetsku bazu završnih radova Nacionalne i sveučilišne knjižnice (stavljanje na raspolaganje javnosti), sve u skladu s Zakonom o autorskom pravu i drugim srodnim pravima i dobrom akademskom praksom, a radi promicanja otvorenoga, slobodnoga pristupa znanstvenim informacijama.

Za korištenje autorskog djela na gore navedeni način ne potražujem naknadu.

U Puli, 25. svibnja, 2016. godine

Potpis

_____

Sadržaj

1 UVOD....1
2 ANALIZA ZAHTJEVA....3
2.1 Korisnički zahtjev....4
2.2 Dijagram slučajeva korištenja....7
2.3 Model baze podataka....9
2.4 Prototip izgleda....13
3 *APPLICATION DEVELOPMENT FRAMEWORK – ADF*....17
3.1 Uvod u ADF....17
3.2 Model....19
3.3 Kontroler sloj....21
3.4 Pogled sloj....23
4 IZRADA APLIKACIJE U ORACLE ADF - U....26
4.1 Izrada ADF modela....29
4.1.1 Izrada entitet objekata....29
4.1.2 Izrada pogled objekata....35
4.1.3 Validacije....42
4.1.4 Liste vrijednosti....49
4.2 Izrada toka zadataka....53
4.2.1 Prijava u sustav e-Izlaznice....54
4.2.2 Pregled, unos, ažuriranje i brisanje izlaznica....55
4.2.3 Unos i ažuriranje osobne izlaznice....57
4.2.4 Unos i slanje elektroničke pošte....58
4.3 Izrada korisničkog sučelja....60
4.3.1 Izrada sučelja za prijavu u sustav....61
4.3.2 Sučelje za pregled osobnih izlaznica....62
4.3.3 Sučelje za unos i ažuriranje osobnih izlaznica....67
4.3.4 Sučelje za unos i slanje elektroničke pošte....69
4.4 Grafički pregled....72
5 ZAKLJUČAK....76
6 LITERATURA....77

1 UVOD

U ovome radu biti će prikazana izrada sustava pod nazivom "e-izlaznice". Sustav e-izlaznice je sustav koji će zamijeniti postojeći sustav papirnatih izlaznica. Izlaznica je formular kojim se evidentira dozvola privatnog ili službenog izlaza zaposlenika ULJANIK Grupe iz kruga poduzeća odnosno van prostora poduzeća.

U sklopu sustava biti će razvijena aplikacija u kojoj će se uz primarnu funkciju odobravanja izlaza zaposlenika, voditi i evidencija izlazaka. Aplikacija e-izlaznice će se integrirati u postojeći poslovno informacijski sustav poduzeća ULJANIK Poslovno informacijski sustavi d.o.o. (skraćeno ULJANIK PIS d.o.o.) uz poštivanje pravila poslovnog procesa poduzeća. Obzirom da je ULJANIK PIS d.o.o. partner tvrtke Oracle Hrvatska d.o.o., posjeduje licencu za Oracle bazu podataka i Oracle aplikacijske servere, postojeći je poslovno informacijski sustav razvijen u Oracle tehnologiji/infrastrukturi, izrada aplikacije e-izlaznice biti će realizirana Oracle razvojnim alatom odnosno u Oracle ADF tehnologiji (eng. *Application development framework*).

U ovome radu biti će prikazan kompletan ciklus izrade aplikacije od prikupljana i analize korisničkog zahtjeva, preko modeliranja i izrade baznih objekata pa do same izrade aplikacije.

U prvom poglavlju biti će prikazano prikupljanje i analiza korisničkog zahtjeva, te određivanje funkcija sustava kroz dijagram slučajeva. Nadalje, biti će prikazana izrada modela baze podataka kroz dijagram entiteta i veza, generiranje baznih objekata kao što su tablice, pogledi, itd. U zadnjem dijelu prvog poglavlja opisati će se izrada prototipa izgleda aplikacije.

Drugo poglavlje ovoga rada govoriti će općenito o Oracle-ovoj ADF tehnologiji, slojevima ADF tehnologije. Biti će pobliže opisani model, pogled i kontroler sloj koji čine osnovu ADF aplikacije. Opisati će se i dijagrami toka zadataka koji u ADF-u čine temelj za određivanje toka aplikacije odnosno navigaciju između pojedinih dijelova i slojeva aplikacije. Pokazati će se i

mogućnost korištenja mnogobrojnih standardnih komponenti prilikom izrade korisničkog sučelja.

U trećem poglavlju biti će prikazana izrada aplikacije u ADF tehnologiji kroz Oracle alat JDeveloper. U JDeveloper-u biti će izrađena kompletna aplikacija e-izlaznice. Izraditi će se model sloj i pripadajući objekti kao što su entitet objekti, pogled objekti te će se prikazati i ugrađivanje zadanih poslovnih pravila, tipove validacija i izrade listi vrijednosti. Opisati će se izrada dijagrama toka zadataka te njihova praktična primjena. U zadnjem dijelu poglavlja biti će prikazana izrada korisničkog sučelja te povezivanje sučelja sa dijagramima toka zadataka i model slojem. Pokazati će se i izrada kalendarskog prikaza podataka, te izrada grafova uz pomoć ADF tehnologije i jednostavnih upita prema bazi podataka.

2 ANALIZA ZAHTJEVA

Analizom zahtjeva utvrđuju se funkcije i karakteristike sustava. To je vrlo bitan dio razvoja sustava zato što kroz korisnički zahtjev doznajemo što se očekuje od sustava odnosno koje funkcije sustav treba mora obavljati. Analizom zahtjeva dolazi se do odgovora na pitanja kao što su :

- Što sustav treba raditi?
- Koja je uloga korisnika?
- Koje karakteristike treba imati?
- Kako sustav treba biti izgrađen? (u koje "tehnološke okvire" treba biti smješten)

Konceptualno analiza zahtjeva može se podijeliti na tri koraka:

1. prikupljanje zahtjeva
2. analiza i obrada zahtjeva
3. dokumentiranje zahtjeva

Kako se u samom procesu razvoja sustava nikad u potpunosti ne prikupe svi zahtjevi (potrebe) korisnika, uvijek se vraćamo i ponavljamo korake prikupljanja, analize i dokumentiranja zahtjeva.

U koraku prikupljana zahtjeva, komunikacijom sa kupcima i korisnicima sustava utvrđuju se njihovi zahtjevi. U ovom koraku prikupljamo informacije o tome što korisnici očekuju od sustava te stječemo "sliku" funkcioniranja cjelokupnog budućeg sustava. Najčešće korištene tehnike prikupljanja zahtjeva su:

- intervju
- anketa
- da/ne pitanja
- promatranjem "na licu mjesta"

U koraku analize i obrade zahtjeva obrađuju se podaci prikupljeni u prethodnom koraku. Utvrđuje se da li su navedeni zahtjevi nejasni, nepotpuni, dvosmisleni ili proturječni. Ako se isto utvrdi, kreće se u rješavanje tih pitanja.

U koraku dokumentiranja zahtjeva, zahtjevi se zapisuju u različite oblike i forme - od pisanih dokumenata kao što je specifikacija sustava do raznih dijagrama, kao što su dijagrami slučajeva i dijagrami toka podataka. Ako je potrebno može se izraditi i prototip izgleda aplikacije. Sve to nam služi za daljnju komunikaciju sa korisnicima, ali i za daljnji razvoj sustava.

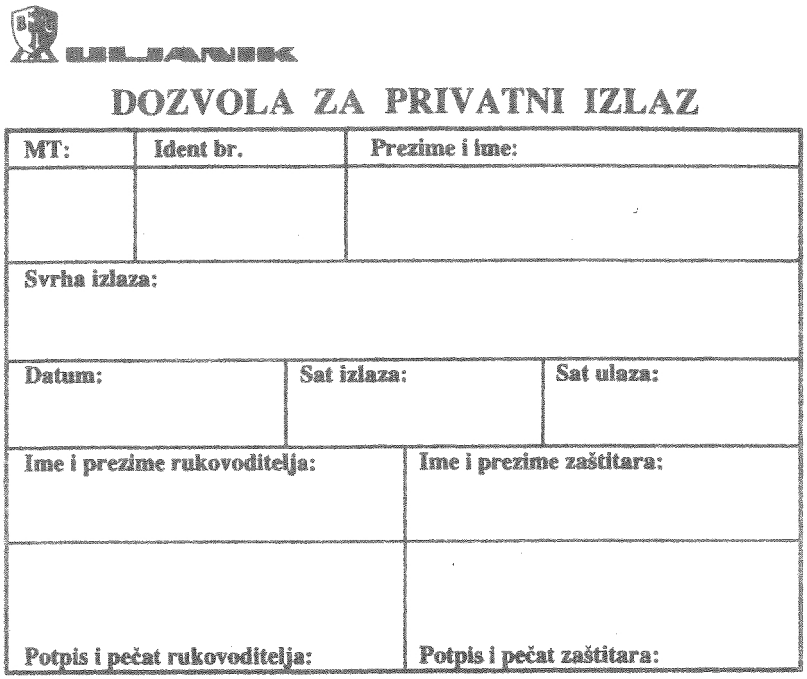
Zahtjev na kojem se temelji ovaj rad dan je u slijedećem potpoglavlju.

2.1 Korisnički zahtjev

Za prikupljanjem odnosno utvrđivanjem zahtjeva u ovom radu korištene su metode intervjuiranja i promatranja na licu mjesta. Nakon obrade rezultata tih metoda zahtjev bi se mogao zapisati odnosno dokumentirati na slijedeći način:

“Sustav e-izlaznice zamijeniti će postojeći sustav papirnatih izlaznica. Sustav omogućuje generiranje i održavanje izlaznica na način da svaki radnik samostalno može generirati zahtjev za izlaz: privatni ili službeni. Osnovne podatke koje treba pratiti su ime i prezime zaposlenika, svrha izlaza, mjesto izlaza, datum i vrijeme izlaza, ime i prezime rukovoditelja i ime i prezime zaštitara. Rukovoditelj zahtjev odobrava elektronski: elektroničkom poštom na osobnom računalu ili mobilnom telefonu. Također je potrebno pratiti i poslane poruke elektroničke pošte, sa slijedećim podacima, ime i prezime primatelja, vrijeme slanja, te tekstualnom napomenom ako je potrebno”.

Da bi se razumio zahtjev opisati ćemo kako taj proces izgleda sada odnosno u “papirnatom obliku”. Papirnati oblik izlaznice vidljiv je na slici 1. , a sam proces odobravanja izlaza iz kruga poduzeća ULJANIK vidljiv je na slici 2.

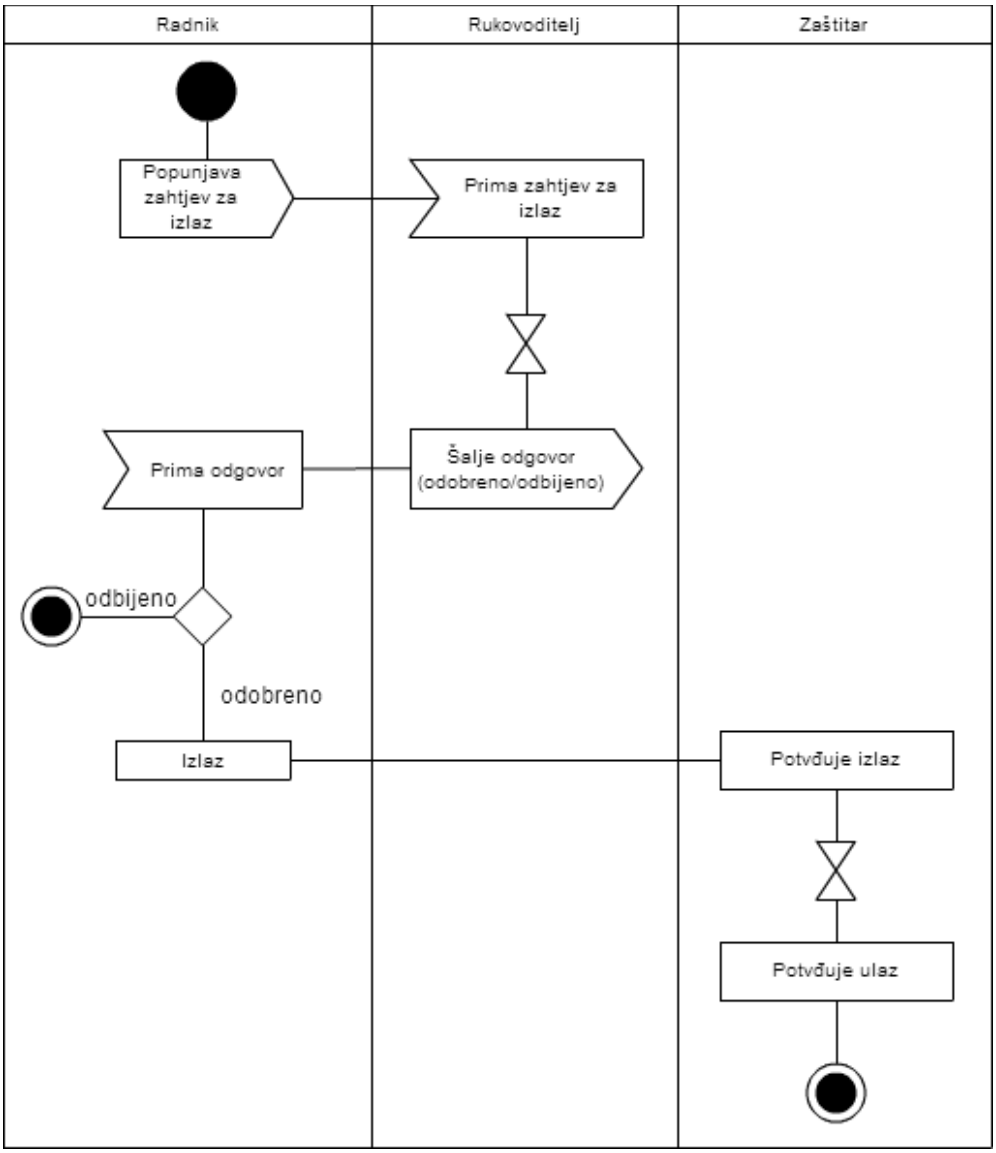


DOZVOLA ZA PRIVATNI IZLAZ

MT:	Ident br.	Prezime i ime:	
Svrha izlaza:			
Datum:	Sat izlaza:	Sat ulaza:	
Ime i prezime rukovoditelja:		Ime i prezime zaštitara:	
Potpis i pečat rukovoditelja:		Potpis i pečat zaštitara:	

Slika 1. Izgled izlaznice

Radnik popunjava zahtjev za izlaz - izlaznicu prikazanu na slici 1. Navode se podaci o mjestu troška (MT), broj radnika (ident.br), ime i prezime radnika, svrha izlaza, datum te sat izlaza i sat ulaza. Nakon što se formular popuni, radnik se upućuje nadređenom rukovoditelju. Rukovoditelj potvrđuje ili odbija zahtjev za izlaz. U slučaju odbijanja zahtjeva za izlaz, proces se zaustavlja. Ako rukovoditelj odobri zahtjev za izlaz, potpisan zahtjev predaje se radniku. Radnik u navedeno vrijeme izlazi iz poduzeća uz prethodno javljanje zaštitaru koji zapisuje vrijeme izlaska i time potvrđuje radnikov izlaz van "granica" poduzeća. Prilikom povratka radnika u poduzeće, zaštitar na izlaznici zapisuje vrijeme povratka.



Slika 2. Proces izlaza zaposlenika iz kruga poduzeća

Razlike u samom procesu između "starog" i "novog" sustava gotovo da i nema. Jedina razlika je u načinu komunikacije koja će se nakon razvoja sustava odvijati elektronskim putem tj. korisnik popunjava zahtjev na računalu, rukovoditelj zahtjev odobrava elektronskim putem na osobnom računalu ili pametnom telefonu i isto tako zaštitar potvrđuje izlaz radnika na računalu.

2.2 Dijagram slučajeva korištenja

Dijagrami slučajeva korištenja (eng. *use case diagram*) prikazuju ponašanje sustava, dijelova sustava ili konkretne klase na način vidljiv korisniku sustava. Ponašanje sustava opisano je pomoću ciljeva odnosno funkcija sustava i učesnika koji predstavljaju apstrakciju korisnika sustava. Ovi dijagrami opisuje samo poglede na ponašanje sustava sa strane korisnikove percepcije, a ne opisuju kako je funkcionalnost izvedena unutar sustava.

Dijagram slučajeva korištenja je prikaz veze između dva slučaja korištenja ili između slučaja korištenja i učesnika. Slučajevi korištenja usko su vezani uz scenariji. Scenarij je slijed koraka koji opisuje interakciju između korisnika i sustava.

Osnovni elementi dijagrama slučajeva su:

- učesnici (eng. *actors*)
- slučajevi korištenja (eng. *use case*)
- veze među njima

Učesnik je vanjski entitet tj. netko izvan sustava, ali u međusobnoj interakciji sa sustavom. Može biti živo biće ili neki drugi sustav. Učesnik je inicijator svih akcija unutar sustava. Učesnici prikazuju uloge koje poprimaju korisnici sustava.

Slučajevi korištenja su slijed operacija koje izvodi sustav, a daju rezultat za izvođača. Predstavljaju apstraktni zadatak kojeg izvode učesnici. Slučajevi korištenja oblikuju usluge (funkcije, mogućnosti) koje sustav nudi.

Veza predstavlja vrstu relacije između učesnika i slučaja korištenja ili između dva ili više slučajeva korištenja ili između dva ili više učesnika.

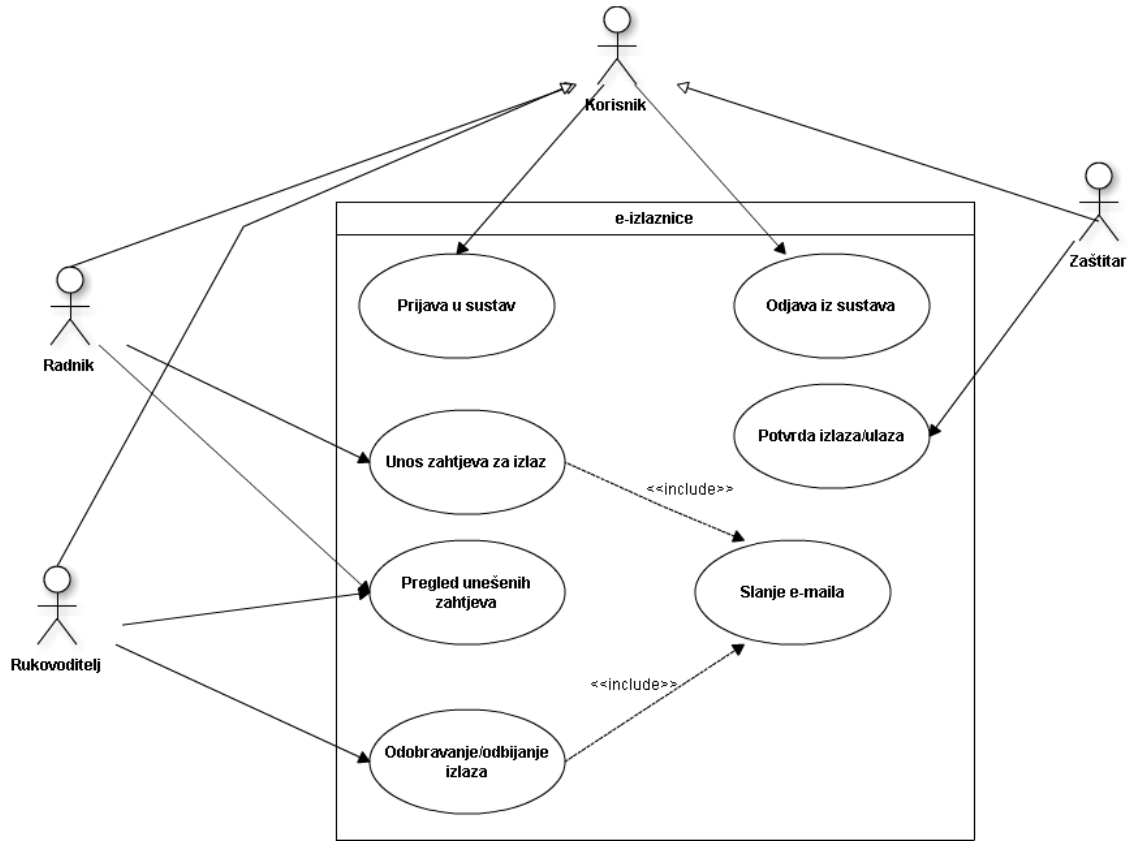
Dijagrami slučajeva koriste se u koraku dokumentacije korisničkog zahtjeva. Zbog razumljivosti samog dijagrama često se koriste i u komunikaciji sa korisnikom.

Konkretno u našem slučaju, analizom korisničkog zahtjeva iz prethodnog potpoglavlja dolazimo do sljedećih učesnika:

- radnik
- rukovoditelj
- zaštitar

Svi ti učesnici su korisnici kojima je zajednički slučaj korištenja prijava odnosno odjava u/iz sustava. Uz taj slučaj korištenja postoje i slijedeći slučajevi korištenja:

- unos zahtjeva za izlaz
- pregled unesenih zahtjeva
- slanje elektroničke pošte
- odobravanje/odbijanje zahtjeva
- potvrda izlaza



Slika 3. Dijagram slučajeva korištenja

Postoje još neki učesnici i slučajevi korištenja koji radi zadržavanja jednostavnosti, prvenstveno iz razloga komunikacije sa korisnikom neće biti ucrtani u dijagram. Sa stanovišta razvoja sustava podrazumijevaju se da postoje. To su na primjer: kod učesnika mogli bi još navesti bazu, aplikacijski server, server elektroničke pošte, itd.

Dijagram slučajeva prikazan je na slici 3. U dijagramu su vidljivi svi prethodno navedeni učesnici i slučajevi korištenja. Vidljive su i veze između elemenata sustava.

2.3 Model baze podataka

Model baze podataka predstavlja reprezentaciju podataka koji će se kreirati, pamtiti i koristiti u poslovnom sustavu. Za grafičko prikazivanje modela baze podataka koristimo model entiteta i veza tj. ER dijagram (eng. *entity relationship diagram*). Osnovni elementi ER dijagrama su entiteti i veze među njima.

Entiteti su objekti odnosno događaji koji nas interesiraju tj. ono o čemu želimo pamtiti podatke (osoba, stvar, mjesto, događaj, itd.). Svaki entitet sastoji se od atributa. Atributi su svojstva entiteta koji ga opisuju. Postoje atributi koji jednoznačno definiraju entitet. Takve atribute nazivamo ključ. Ključ može biti jednostavan ili složen ovisno o broju atributa koji jednoznačno definiraju entitet.

Između entiteta uspostavljaju se veze. Entitet može imati i vezu na samog sebe, npr. u slučaju neke hijerarhije. Funkcionalnost veze može biti:

- 1:1 - instanca jednog entiteta može biti u vezi sa jednom instancom drugog entiteta
- 1:N - instanca jednog entiteta može biti u vezi sa više instanci drugog entiteta
- M:N - više instanci jednog entiteta može biti u vezi sa više instanci drugog entiteta

Sustav koji je tema ovoga rada implementira se u postojeću bazu podataka poslovno informacijskog sustava, stoga je broj novonastalih entiteta mali. Analizirajući zahtjev opisan u

poglavlju 1.1 vidimo da je potrebno pamtiti podatke o zahtjevima za izlaz. Za tu potrebu definiramo entitet RR_IZLAZNICE[1]. Iz zahtjeva vidimo da su neki od atributa entiteta:

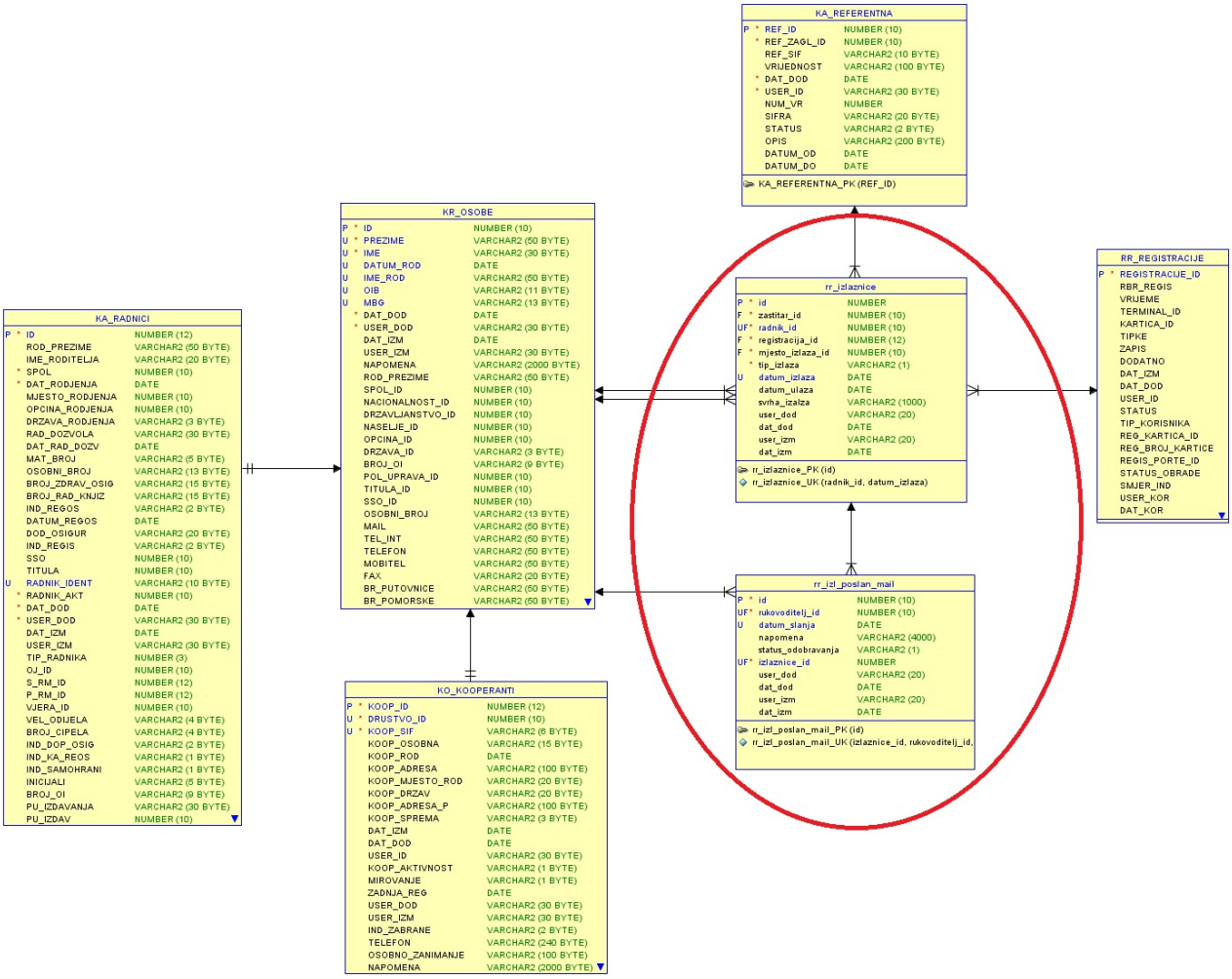
- radnik
- vrijeme izlaza (navedeno prilikom popunjavanja zahtjeva)
- vrijeme povratka (navedeno prilikom popunjavanja zahtjeva)
- status (odobreno, odbijeno)
- vrsta izlaza (privatno, službeno)
- mjesto izlaza
- stvarno vrijeme izlaza (registracija karticom zaposlenika)
- stvarno vrijeme povratka (registracija karticom zaposlenika)
- zaštitar
- itd.

Dalje, vidimo da će se za svaki zahtjev generirati jedna ili više elektronskih poruka (eng. *e-mail*), te definiramo slijedeći entitet RR_IZL_POSLAN_MAIL. U RR_IZL_POSLAN_MAIL pamtit će se (atributi):

- vrijeme slanja
- rukovoditelj kojem se šalje elektronska pošta
- tijelo poruke
- vrijeme odgovora
- itd.

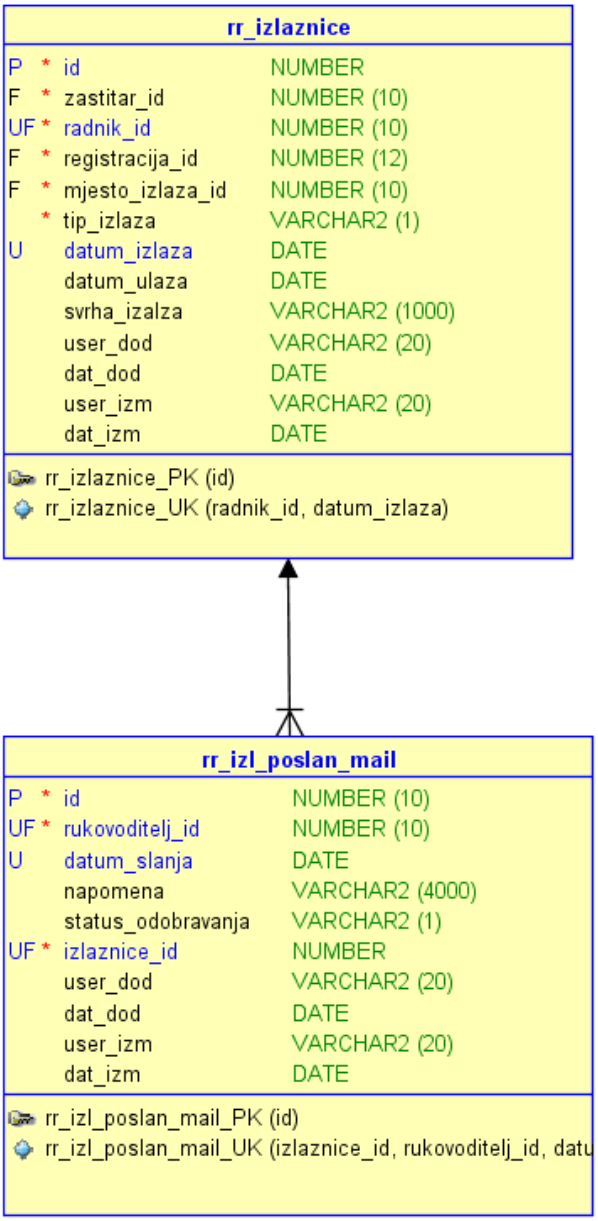
Iz gore opisanih entiteta i atributa dolazimo do slijedećeg modela entiteta i veza tj. do ER dijagrama prikazanog na slici 4. Na ER dijagramu crvenom bojom zaokruženi su novonastali entiteti. Ostali entiteti su dio već postojećeg informacijskog sustava, a koristiti ćemo ih u sustavu e-izlaznice.

[1] Sustav e-izlaznice implementira se u postojeći poslovno informacijski sustav kao dio modula "**R**egistracija **r**adnog vremena", pa otuda prefiks RR.



Slika 4. ER dijagram

Iz ER dijagrama vidljiva je veza između entiteta RR_IZLAZNICE i RR_IZL_POSLAN_MAIL (Slika 5.). Kardinalnost te veze je 1:N, odnosno za jedan zahtjev za izlaz postoje više poslanih elektronskih poruka. Najjednostavniji je slučaj kada elektronska pošta "putuje" radnik - rukovoditelj (jedan slog u RR_IZL_POSLAN_MAIL) i rukovoditelj – radnik (još jedan slog u RR_IZL_POSLAN_MAIL). Međutim, postoji i slučaj kada rukovoditelj može iz nekog razloga proslijediti poruku drugom rukovoditelju ili nadređenoj osobi te ta osoba dalje obrađuje zahtjev za izlaz. Ta prosljeđivanja također se pamte u RR_IZL_POSLAN_MAIL.



Slika 5. Entiteti korišteni u aplikaciji e-izlaznice

Uz entitete RR_IZLAZNICE i RR_IZL_POSLAN_MAIL koristiti ćemo i entitete iz već postojećeg sustava. To su sljedeći entiteti:

- RR_REGISTRACIJE – u toj tablici vodi se evidencija registracija zaposlenika pri ulasku i izlasku iz poduzeća. Tablica se automatski puni prilikom prolaska radnika sa osobnom

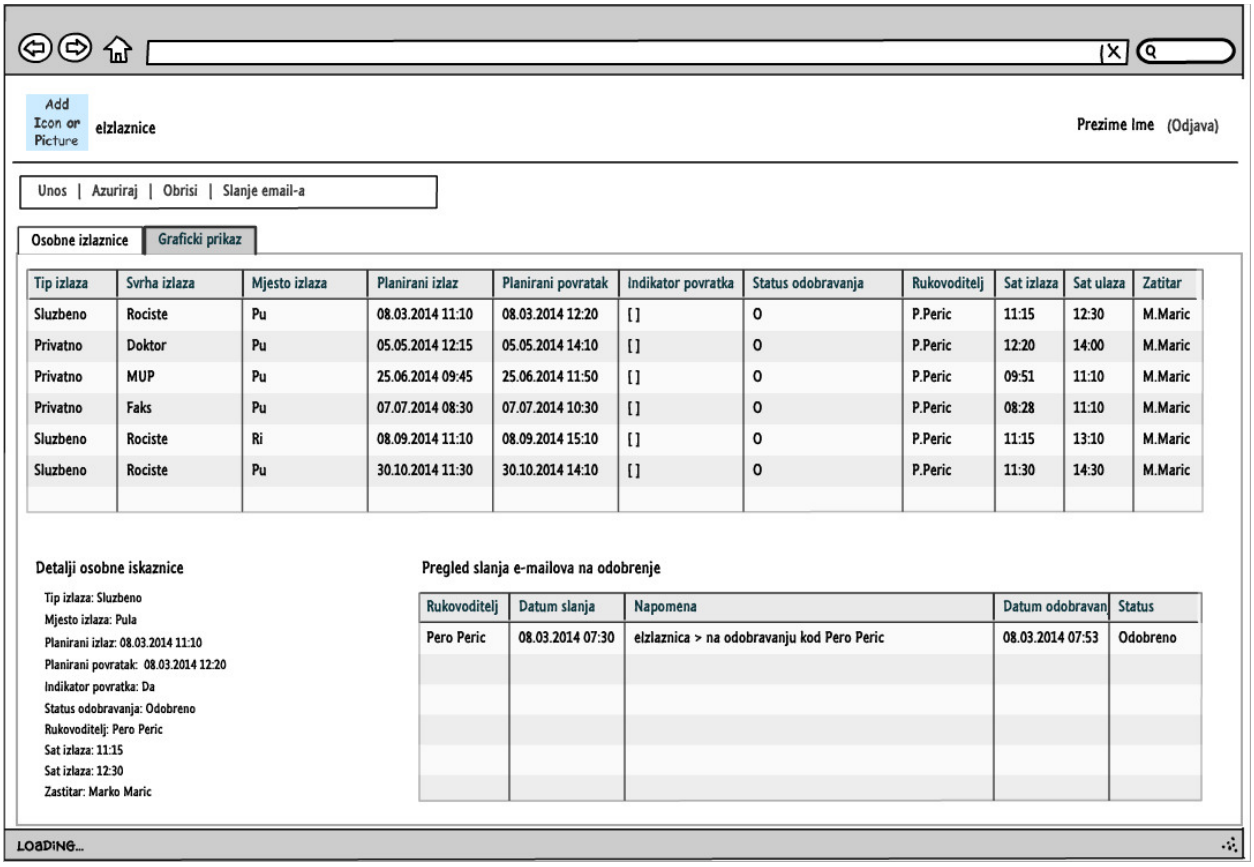
karticom kroz uređaj za registraciju. Taj entitet iskoristiti ćemo u obliku vanjskog ključa za pamćenje vremena izlaska i ulaska

- KR_OSOBE – u toj tablici zapisane su osobe koje su zaposlenici (KA_RADNICI), kooperanti (KO_KOOPERANTI) i vanjske osobe. Vanjski ključ na ovu tablicu služiti će nam kao veza za pamćenje zaposlenika i zaštitara u RR_IZLAZNICE, a kod RR_IZL_POSLAN_MAIL za pamćenje rukovoditelja
- KA_REFERENTNA – je opći šifrarnik. U njemu se između ostalog nalazi i šifrarnik mjesta izlaza koji želimo koristiti u aplikaciji e-izlaznice.

2.4 Prototip izgleda

Izgled aplikacije definira kako će vanjski entiteti i sustav komunicirati. Korisničko sučelje predstavlja vezu između vanjskih entiteta i sustava. Cilj izrade korisničkog sučelja je jednostavnost uporabe, intuitivnost prilikom uporabe, oku ugodan izgled, te da umanji trud koji korisnik treba uložiti prilikom obavljanja posla. Prototipom izgleda definiramo kako će okvirno izgledati korisničko sučelje te nam služi kao vodič prilikom predstavljanja izgleda aplikacije korisniku. Tehnologija u kojoj se sustav realizira definira tip korisničkog sučelja. Ako se radi o web aplikaciji, korisničko sučelje je web stranica odnosno neka web tehnologija koja se izvršava unutar internet preglednika. Ako se radi o aplikaciji koja se pokreće unutar operacijskog sustava klijenta onda govorimo o *desktop* aplikaciji koja je namijenjena tom operacijskom sustavu. U našem slučaju, obzirom da će sustav e-izlaznice biti dio postojećeg poslovno informacijskog sustava, tehnologija u kojoj će sustav biti realiziran već nam je zadan. Radi se o Oracle web tehnologiji odnosno o "*Application Development Framework*" (ADF) o kojem će detaljnije biti riječi u slijedećem poglavlju. Prilikom osmišljavanja izgleda sustava e-izlaznice, moramo paziti da koristimo dostupne elemente koje nam ADF nudi za oblikovanje korisničkog sučelja odnosno komunikaciju sa korisnikom.

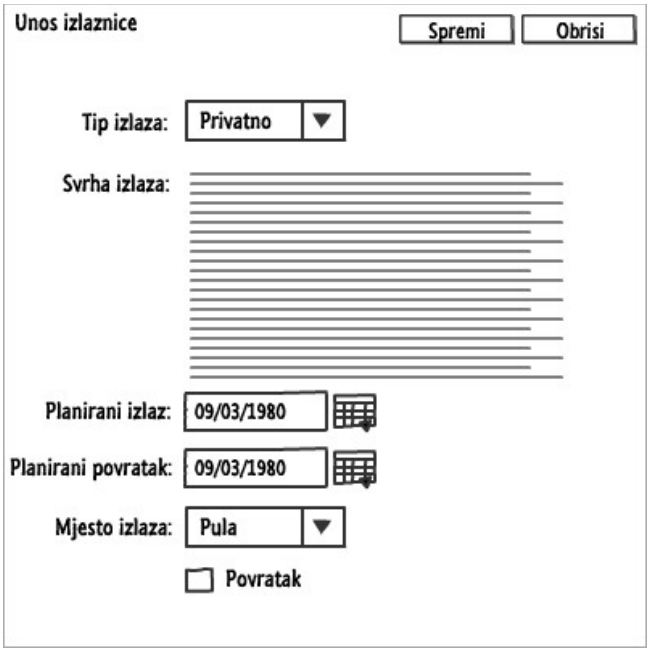
Za sustav e-izlaznice odlučili smo se na izgled kao što je prikazano na slici 6. Bitno je naglasiti da se tu radi o prototipu izgleda, te da će završni izgled aplikacije više - manje odstupati od prototipa.



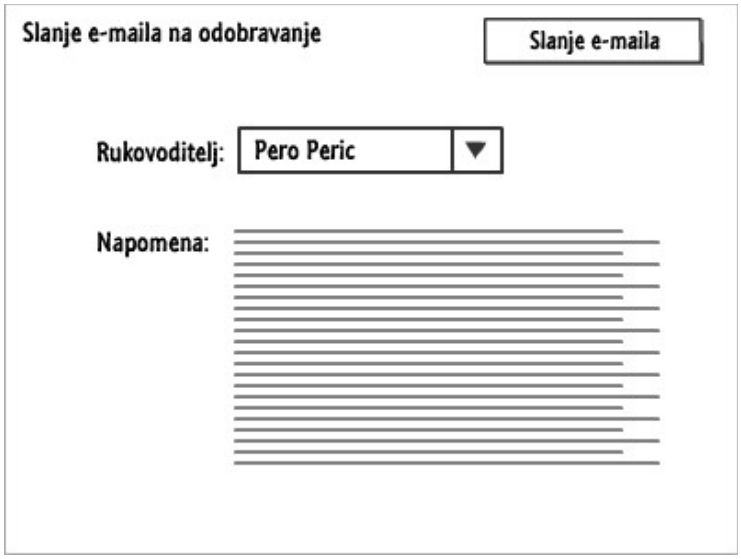
Slika 6. Prototip izgleda korisničkog sučelja

Izgled aplikacije možemo podijeliti u četiri dijela kao što je prikazano na slici 6. U prvom dijelu nalazi se logo i naziv aplikacije te podaci o prijavljenom korisniku i tipka za odjavljivanje iz sustava. Između prvog i drugog dijela imamo poveznice (eng. *link*) za navigaciju između različitih dijelova aplikacije. Drugi dio smješten je unutar kartica (eng. *tab panel*), te omogućuje kretanje između osobnih zahtjeva i grafičkog prikaza. *Tab panel* omogućuje nam i buduće lakše nadograđivanje sustava - lakim dodavanjem novih kartica.

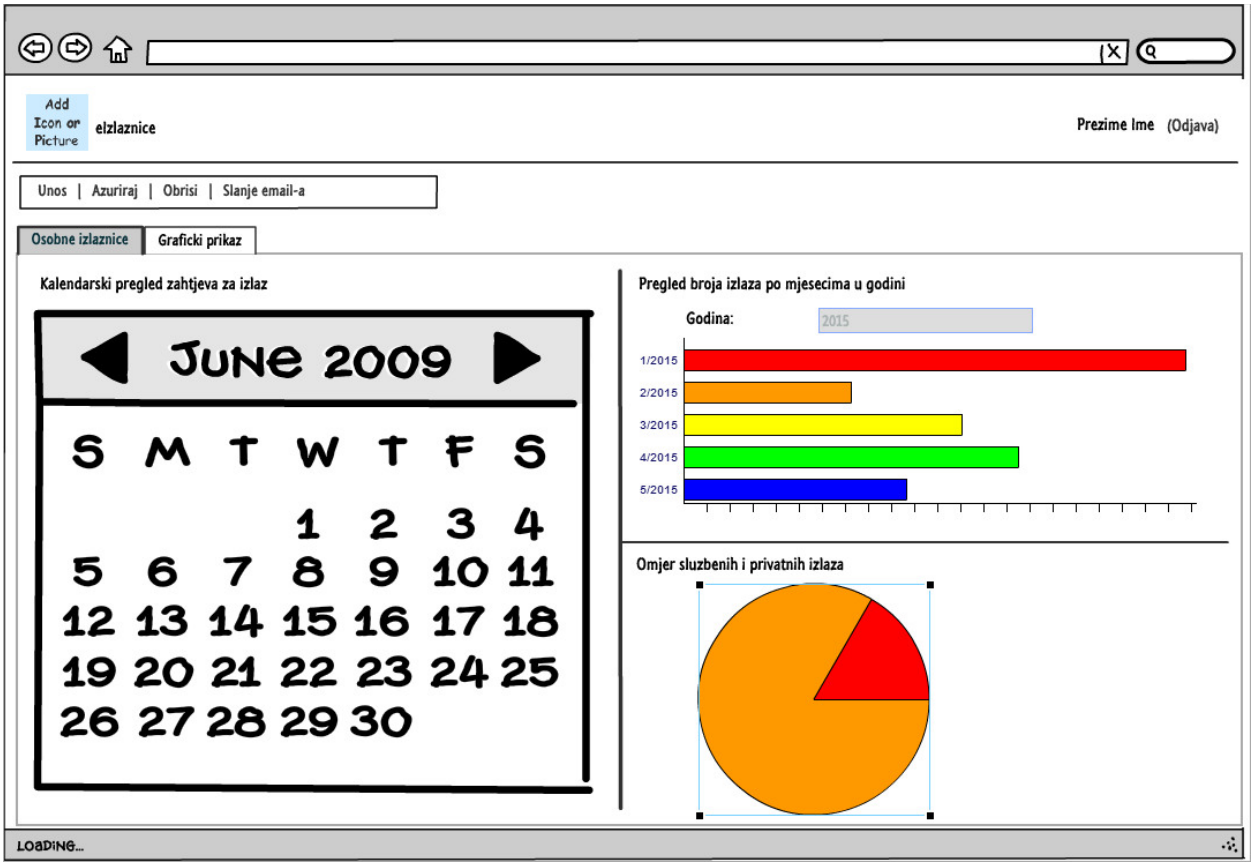
U drugom dijelu prikazan je tabelarni pregled zahtjeva za izlaz, dok je u trećem dijelu prikazan isti taj pregled samo za pojedinačan slučaj zahtjeva za izlaz ovisno o poziciji pokazivača u drugom dijelu tzv. prikaz pojedinačnog sloga (eng. *single record view*). Četvrti dio prikazuje evidenciju poslane elektroničke pošte. Za dodavanje i ažuriranje slogova te unos i slanje elektroničke pošte zamišljeno je korištenje skočnih prozora (eng. *popup wndow*) (slika 7. i 8.).



Slika 7. Prototip izgleda unosa osobne izlaznice



Slika 8. Prototip izgleda za slanje elektroničke pošte



Slika 9. Kalendarski prikaz i grafovi

U drugoj kartici (slika 9.) smješten je kalendarski prikaz zahtjeva za izlaz, dok su na desnoj strani kartice smješteni grafički prikaz broja izlaza po mjesecima za godinu i omjer privatnih i službenih izlazaka za godinu.

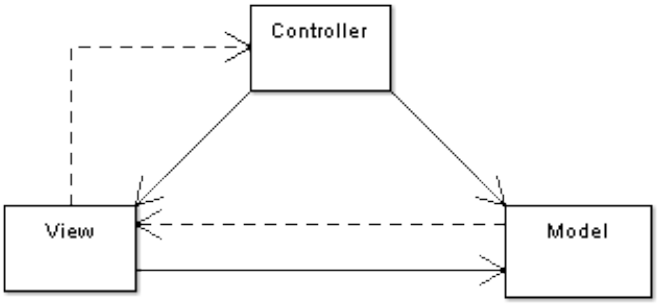
3 *APPLICATION DEVELOPMENT FRAMEWORK – ADF*

3.1 Uvod u ADF

Oracle "Application Development Framework" je Java *framework* namijenjen izradi poslovnih aplikacija. Skraćeno ga nazivamo Oracle ADF. Oracle ADF podržava model brzog razvoja aplikacija (eng. *RAD - Rapid Application Development*) kroz gotove komponente i funkcionalnosti. Oracle kao razvojno okruženje (eng. *IDE - Integrated Development Environment*) za rad u ADF-u preporuča Oracle JDeveloper. JDeveloper pruža vizualni i deklarativni pristup izradi Java poslovnih aplikacija u ADF-u na način da su ADF komponente grafički izložene programeru kroz sučelje JDeveloper-a .

Oracle ADF baziran je na *Model-View-Controller* (MVC) arhitekturi (slika 10.). Kao što iz samog naziva proizlazi, MVC arhitektura podijeljena je u tri sloja:

- model
- pogled (eng. *view*)
- kontroler (eng. *Controller*)



Slika 10. MVC arhitektura.

(Izvor: https://doc.odoo.com/6.0/developer/1_3_oo_architecture/mvc/)

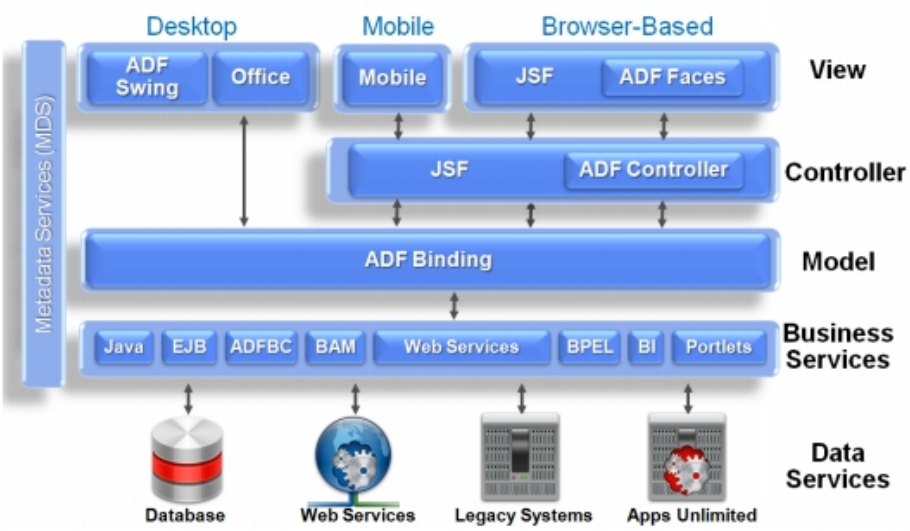
Model se sastoji od podataka, poslovnih pravila/logike i funkcija ugrađenih u programsku logiku. Također model dio zadužen je za interakciju sa bazom podataka. Model sloj spaja poslovne servise i objekte koji koriste te servise u drugim slojevima MVC arhitekture. U Oracle ADF-u model sloj predstavlja jedinstveno sučelje koje povezuje baze podataka, poslovne servise i ostale

izvore podataka (ulaze) sa ostalim dijelovima ADF arhitekture. Takvo sučelje dozvoljava nam mijenjanje izvora tip podataka bez mijenjanja cijele aplikacije.

Pogled dio zadužen je za izgled aplikacije odnosno za interakciju sa korisnikom na način da korisniku prezentira podatke, ali i prima podatke i akcije od strane korisnika. Prezentacija može biti kroz tablice, forme, grafikone, itd. U Oracle ADF-u može se izraditi korisničko sučelje namijenjeno klasičnoj (*desktop*) aplikaciji ili korisničko sučelje namijenjeno izvođenju u Internet pretraživaču (web aplikacija). Pogled sloj može se mijenjati neovisno o model dijelu te je to jedna od prednosti MVC arhitekture.

Kontroler dio zadužen je za vezu između korisnika i sustava. Kontroler vrši interakciju između korisnika, modela sloja i pogled sloja tj. na temelju korisničkog ulaza upravlja modelom i pogledom. Na primjer, kada korisnik pritisne akciju (tipku) pretraživanja, kontroler sloj odlučuje koju akciju treba izvršiti (pretraživanje) i koju stranicu treba prikazati (rezultate pretraživanja). Iz ovog primjera vidimo da je kontroler sloj zadužen za upravljanje korisničkim ulazima i tokom aplikacije.

Ovakav tip arhitektura olakšava razvoj i održavanje aplikacije jer se neovisno mogu razvijati odnosno održavati sva tri sloja MVC arhitekture.



Slika 11. Oracle ADF arhitektura (izvor: http://www.oracle.com/)

Uz ova osnovna tri sloja MVC arhitekture, Oracle ADF model sloj dijeli na još jedan dio, sloj "Poslovnih servisa"(eng. *Business Services layer*). To je sloj koji nam omogućuje vezu prema izvorima podataka (baza podataka, datoteke, web servisi) i upravlja poslovnog logikom. Izgled Oracle ADF arhitekture vidi se na slici 11. Ovakva arhitektura ostavlja projektantu na izbor tehnologiju koju će koristiti u pojedinom sloju.

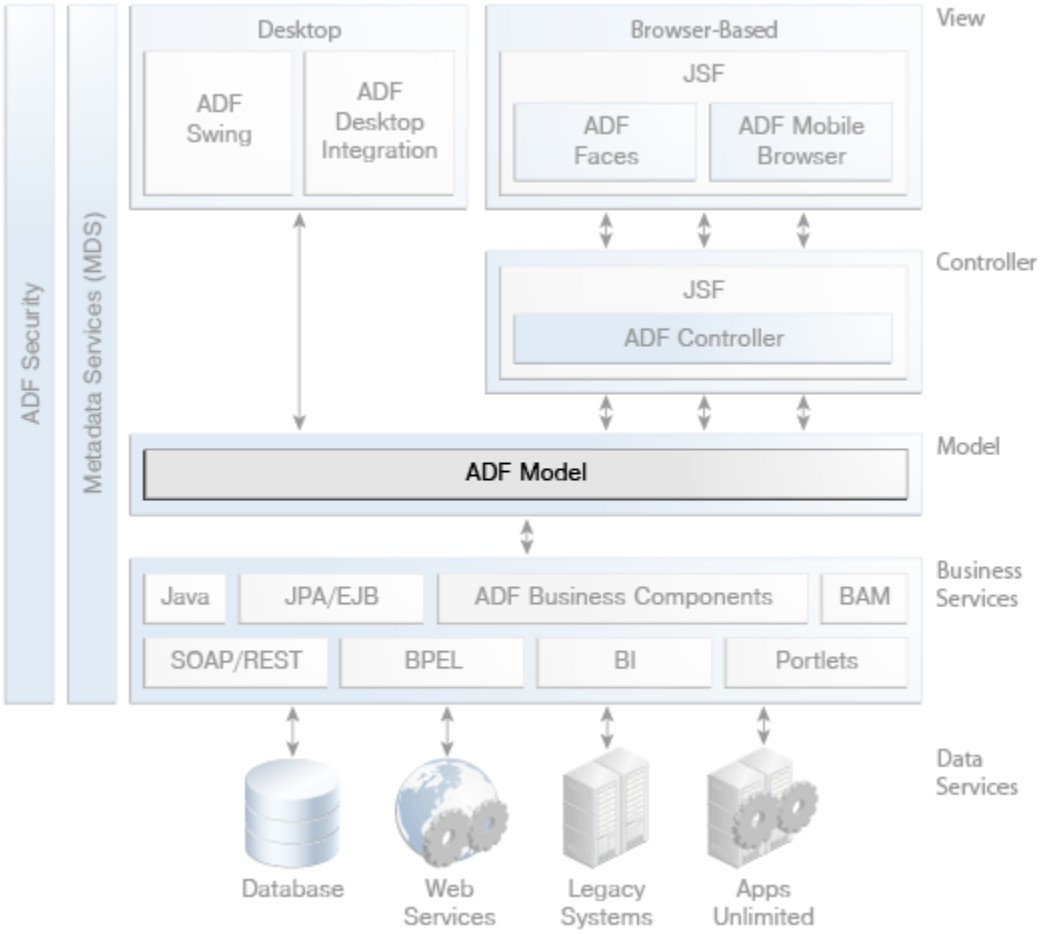
Za sustav e-izlaznice izvor podataka nam je Oracle baza podataka. Poslovni servisi odnosno model sloj realizirati će se korištenjem "ADF-BC" (eng. ADF *business component*) komponente. Za kontroler dio odabrana je komponenta "ADF Controller". Obzirom da je sustav e-izlaznice web aplikacija tj. izvoditi će se unutar Internet pretraživača za izradu pogled sloja koristiti će se "ADF Faces" komponente.

Iz gore navedenoga zaključujemo da Oracle ADF olakšava razvoj poslovnih Java aplikacija nudeći gotove komponente i funkcionalnosti koje se uz primjenu JDeveloper-a mogu na jednostavan način koristiti i ugraditi prilikom razvoja aplikacije. Odabir tehnologije koje će se koristiti ostavljena je samom projektantu na izbor. To nam omogućuje ADF-ova arhitektura i model sloj koji čini sučelje između tehnologije korištene na kontroler i pogled sloju sa izvorima podataka tj. najčešće sa bazom podataka.

3.2 Model

ADF model sloj povezuje poslovne servise sa objektima u drugim slojevima kao što je vidljivo na slici 12. ADF model možemo promatrati kao apstraktni sloj između poslovnih servisa, kontroler i pogled sloja, tj. predstavlja jedinstveno sučelje (eng. *interface*) za pristup bilo kojem tipu poslovnih servisa.

Model sloj sastoji se od dvije komponente, kontrole podataka(eng. *data control*) i podatkovnih veza (eng. *data bindings*). Ti slojevi koriste meta podatke zapisane u datotekama za definiranje sučelja između model sloja i ostalih slojeva ADF *framework-a*.



Slika 12. ADF model sloj

(Izvor:https://docs.oracle.com/middleware/1212/adf/ADFCG/model.htm#ADFCG122*/)

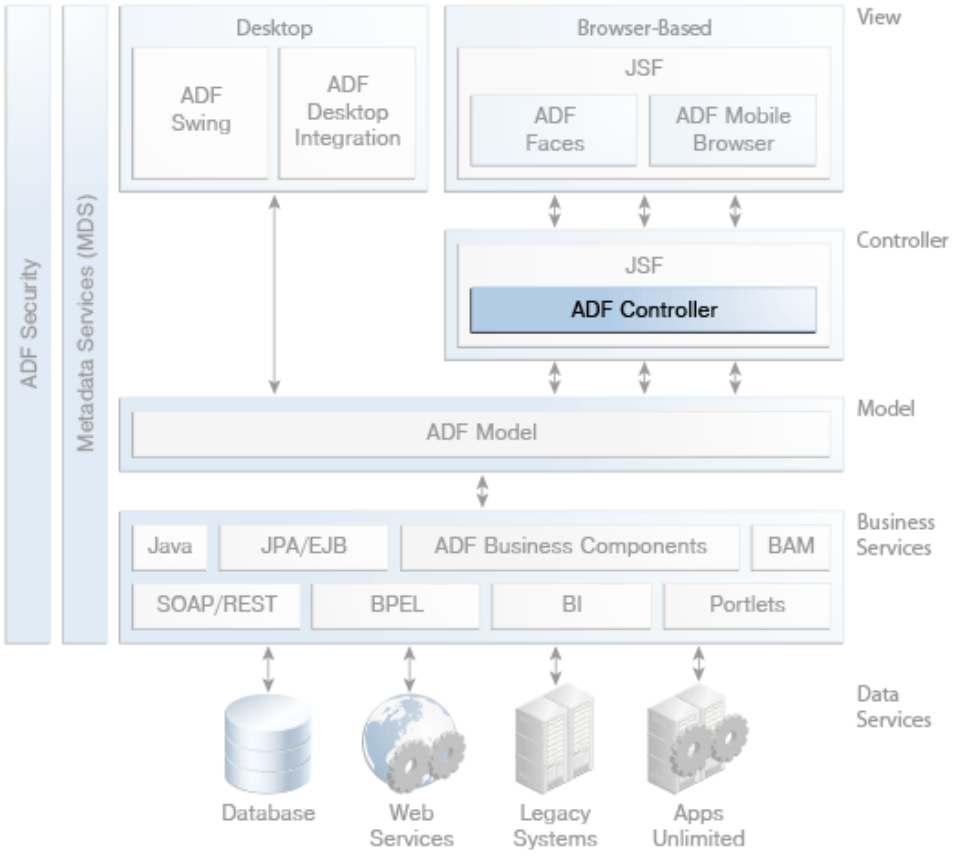
Komponenta kontrole podataka predstavlja detalje implementacije poslovnog servisa, a komponenta podatkovnih veza izlaže metode i atribute podatkovne kontrole komponentama korisničkog sučelja. Na taj se način dobiva odvajanje model i pogled sloja (MVC arhitektura).

Sa projektantsko/programerskog pogleda ovakva arhitektura model sloja omogućava promjenu poslovnog servisa bez promjena ostalih dijelova aplikacije, te stvara jedinstven način rada bez obzira o kojem se poslovnom servisu radi.

3.3 Kontroler sloj

U Oracle ADF-u kontroler sloj zadužen je za upravljanje tokom aplikacije i upravljanje korisničkim akcijama. Pri izradi web aplikacije kao što je aplikacija e-izlaznice na raspolaganju su nam dvije tehnologije za izradu kontroler sloja (slika 13.):

- JSF kontroler (eng. *Java Server Faces*)
- ADF kontroler

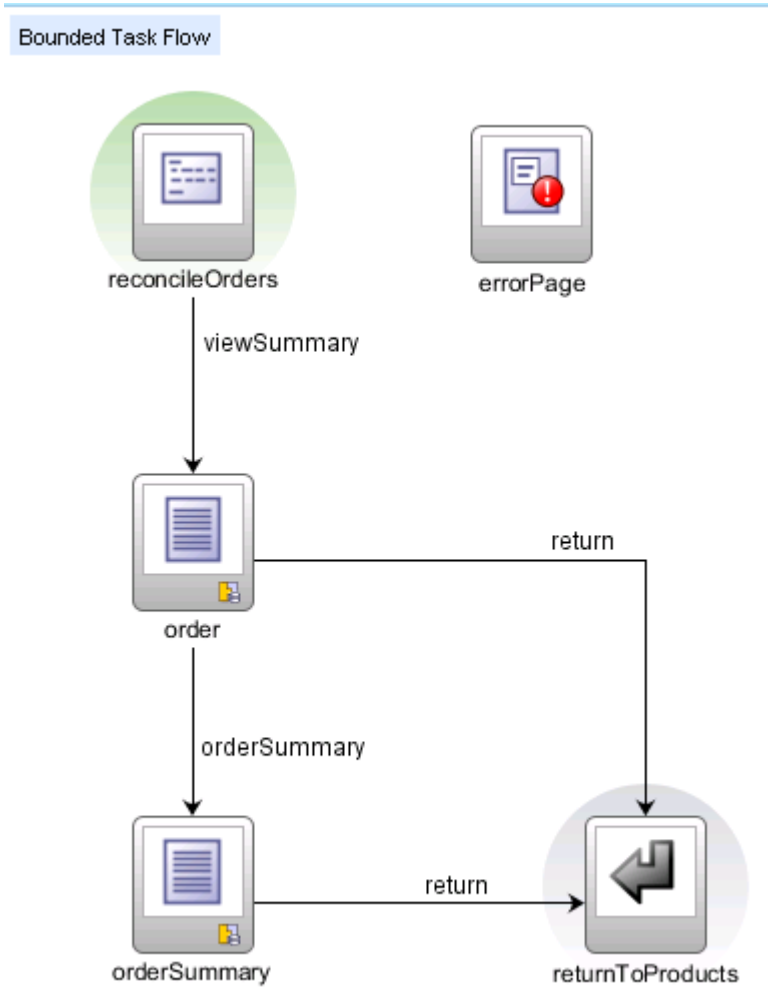


Slika 13. Kontroler sloj

(Izvor:https://docs.oracle.com/middleware/1212/adf/ADFCG/model.htm#ADFCG129)

Za izradu aplikacije e-izlaznice korišten je ADF kontroler koji je zapravo nadogradnja funkcionalnosti JSF kontrolera.

Neovisno o izbranoj tehnologiji kontroler sloja, izrada tog dijela aplikacije u JDeveloper-u svodi se na crtanje dijagrama kroz "*drag and drop*" sučelje. Takvi dijagrami nazivaju se dijagrami toka zadataka (eng. *task flow)*. Primjer jednog takvog dijagrama prikazan je na slici 14.



Slika 14. Primjer dijagrama toka zadataka

(Izvor: https://docs.oracle.com/middleware/1212/adf/ADFCG/controller.htm#ADFCG217)

Dijagram toka zadataka sastoji se od :

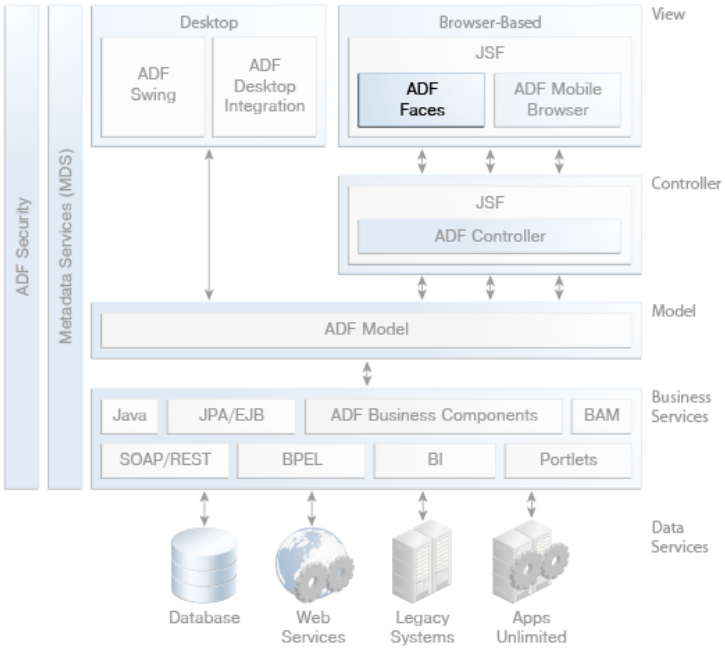
- čvorova aktivnosti - predstavlja radnju kao što je prikaz stranice, izvršavanje aplikacijske logike(poziv procedura, funkcija, metoda), pozivanje drugog *task flow-a* i slično.

- veza između aktivnosti - predstavlja tranziciju između dva čvora aktivnosti.

Dijagrami toka zadataka omogućuju modularan pristup kontroli toka aplikacije, na način da možemo tok aplikacije podijeliti u više manjih logičkih cjelina, a te manje cjeline možemo ponovo upotrijebiti u drugim aplikacijama.

3.4 Pogled sloj

Pogled sloj predstavlja korisničko sučelje aplikacije. Za izradu pogled sloja također su nam na raspolaganju nekoliko tehnologija ovisno o tome da li se radi o desktop ili web aplikaciji (slika 15). Budući da će se korisničko sučelje aplikacije e-izlaznice prikazivati u Internet pretraživaču, tehnologije koje spadaju pod "desktop" ne odgovaraju. Za tehnologiju izrade korisničkog sučelja izabrana je "ADF Faces" tehnologija.



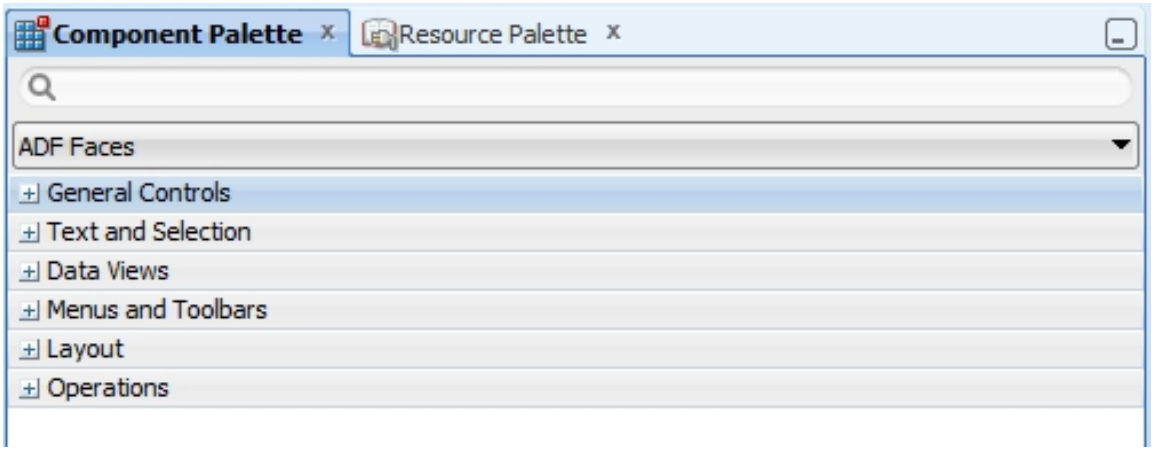
Slika 15. Pogled sloj

(Izvor:https://docs.oracle.com/middleware/1212/adf/ADFCG/faces.htm#ADFCG203)

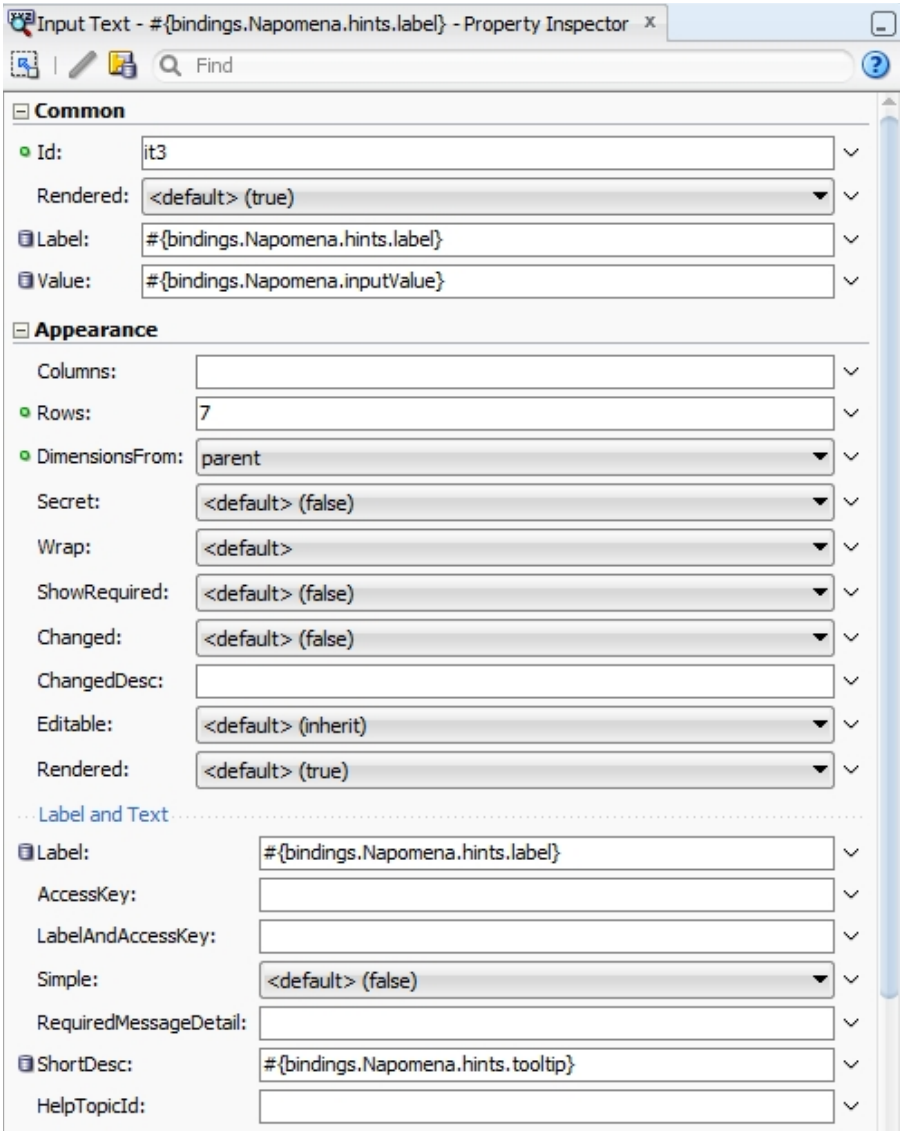
Za izradu korisničkog sučelja u "ADF Faces" tehnologiji na raspolaganju nam je preko 150 komponenti. Komponente su podijeljene u nekoliko skupina (slika 16):

- komponente zadužene za vizualni smještaj i raspored ostalih komponenti

- komponente za unos teksta i komponente izbora
- komponente za prikaz podataka kao što su tablice, forme, hijerarhijska stabla, kalendari, grafovi, itd.
- komponente za izradu meni-a i alatnih traka
- komponente za izradu akcija - *drag and drop*, izvoz podataka u tablični kalkulator, akcije na pritisak tipki, skočni (eng. *pop-up*) meni, itd.



Slika 16. Komponente "ADF Faces" tehnologije.

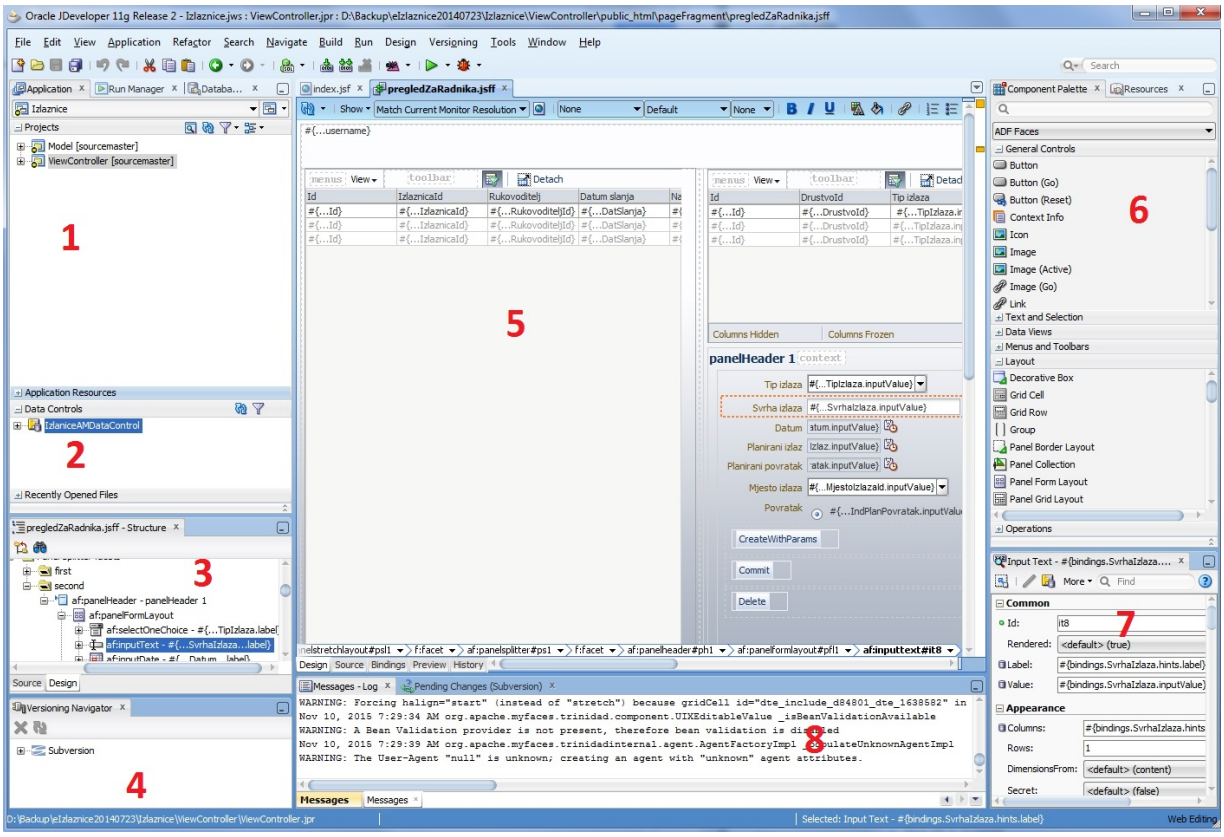


Slika 17. Dio palete svojstva polja za unos podataka

Rad na pogled sloju u Oracle JDeveloper-u svodi se na izbor i "povlačenje" (eng. *drag and drop*) komponente iz palete na radnu površinu. Svaka komponenta ima više desetaka svojstva koje možemo mijenjati kroz paletu svojstva (eng. *property palette*) (slika 17.) bez potrebe za izmjenom programerskog koda tj. nakon izmjene svojstva u paleti, JDeveloper se "brine" o izmjeni programerskog koda.

4 IZRADA APLIKACIJE U ORACLE ADF - U

Za izradu aplikacije e-izlaznice koristiti će se Oracle alat JDeveloper. JDeveloper je razvojna okolina koja podržava razvoj aplikacije u raznim programskim jezicima kao što su Java, Javascript, PHP, PL/SQL, itd. Također podržava i objektni pristup projektiranju informacijskog sustava te izradu dijagrama slučajeva, dijagrama klasa, dijagramom aktivnosti, itd. JDeveloper je u potpunosti integriran sa ADF-om što olakšava razvoj aplikacije u toj tehnologiji. Na slici 18. vidimo izgled JDeveloper razvojne okoline.



Slika 18. JDeveloper razvojno okruženje

Iako izgled radne okoline JDeveloper možemo mijenjati po volji na način da se komponente pozicioniraju na lijevu ili desnu stranu ekrana odnosno gore ili dole, postoje neki "uobičajeni" dijelovi JDeveloper radne okoline.

Te su komponente na slici 18. označene crvenim brojevima:

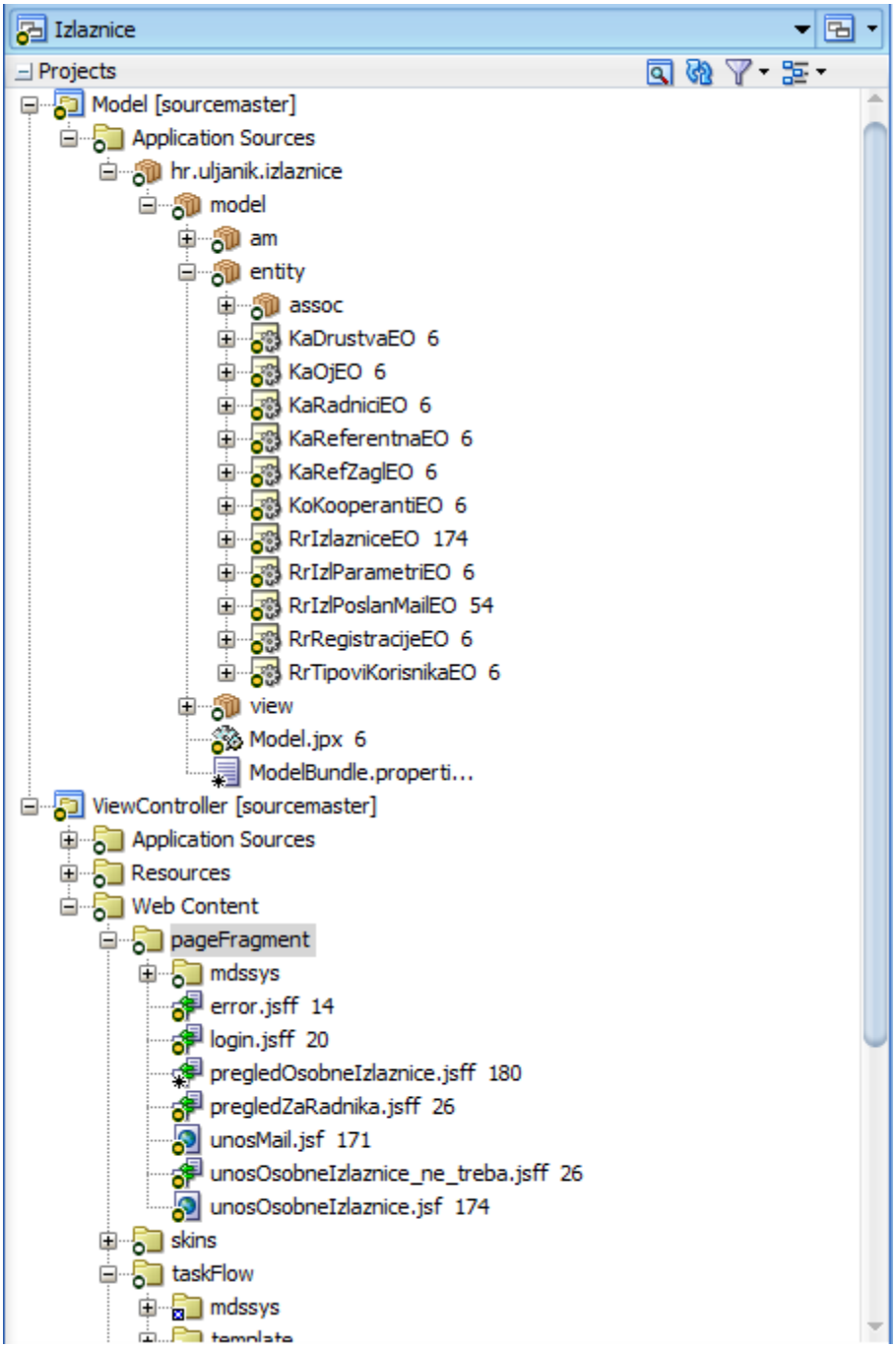
1. Pregled projekata
2. Pregled objekata izloženih pogled sloju
3. Struktura objekta prikazanog u sredini radne površine (5)
4. Komponenta za kontrolu verzija programskog koda (eng. v*ersion control*)
5. Glavni dio radne površine koji služi za uređivanje objekata. Prikaz rasporeda komponenti, programskog koda, datoteka, raznih dijagrama, itd.
6. Pregled dostupnih komponenti za povlačenje na radnu površinu.
7. Svojstva odabrane komponente na glavnom dijelu radne površine (5)

U JDeveloper-u organizacija aplikacije podijeljena je u projekte, a projekti u podpodručja. Tipična ADF aplikacija podijeljena je u dva projekta. Prvi projekt sadrži entitete, pogled objekte, poslovna pravila, Java klase, aplikacijski modul i naziva se "*Model"*. Model projekt podijeljen je u pakete radi lakšeg snalaženja, preglednosti i organizacije projekta. Za imenovanja paketa, klasa, atributa i metoda, varijabli, konstanti koristimo Java konvenciju imenovanja[2].

Drugi projekt sadrži korisničko sučelje, dijagrame toka zadataka i Java klase kojima želimo pristupati iz pogled sloga npr. obrada prijavljivanja (autorizacija i autentikacija korisnika). Taj drugi projekt zove se "*ViewController"* (slika 19). Projekt ViewController podijeljen je u mape koje sadrže korisničko sučelje, u našem slučaju ADF stranice, ikone, tokove zadataka. Također sadrži i jave klase tzv. "*java bean-ove*" koje koristimo za prilagodbu aplikacije korisničkim željama i potrebama ako nam klasična ADF funkcionalnost ne odgovara.

Oracle JDeveloper možemo koristiti i kao alat za pristupanje bazi podataka. U tu svrhu smo ga koristili prilikom izrade upita, pogleda, procedura, funkcija, paketa i ostalih baznih objekata.

[2] Java-ina konvencija imenovanja (eng. *Java naming conventions*)
http://www.oracle.com/technetwork/java/javase/documentation/codeconventions-139411.html



Slika 19. Organizacija ADF projekta

Kroz JDeveloper podržan je cijeli ciklus izrade aplikacije od modeliranja do implementacije aplikacije na aplikacijski server.

4.1 Izrada ADF modela

ADF model najčešće se sastoji od :

- entitet objekata (eng. *entity object*)
- pogled objekata (eng. *view object*)
- veza između entiteta
- veza između pogled objekata
- poslovnih pravila

Prije izrade modela u ADF-u potrebno je kreirati tablice na bazi. ER dijagram koji je prikazan na slici 4 iz prvog poglavlja izrađen je također u Oracle alatu naziva "Data Modeler". Data modeler nakon izrade ER dijagrama ima mogućnost kreiranja skripte na temelju nacrtanog dijagrama. Skripta nam služi za generiranje objekata na bazi. Budući da se radi o nadogradnji postojećeg sustava, skripta sadrži isključivo *DDL* (eng. *Data definition language*) naredbe za novo nastale entitete (zaokruženo na slici). To nam je preduvjet za izradu ADF entiteta.

U prethodnom odlomku spominju se entiteti u kontekstu ADF-a i baze odnosno entiteti na ER dijagramu. Iako predstavljaju isti objekt, tablicu na bazi, ADF entitet je Java reprezentacija tog objekta (u ovom slučaju tablice) na bazi. U slijedećem poglavlju biti će detaljne razrađeni ADF entiteti korišteni u aplikaciji e-izlaznice.

4.1.1 Izrada entitet objekata

Oracle ADF entiteti, objekti su poslovne komponente koje čine odnosno opisuju poslovni model, uključujući podatke, pravila i sposobnost komunikacije sa izvorom podataka. Entitet objekti su Java klase koje predstavljaju neki objekt, u ovom konkretnom slučaju tablicu na bazi. Jedan entitet objekt predstavlja jedinstven objekt iz izvora podataka. To znači da će npr. za tablicu RR_IZLAZNICE postojati njoj pripadajući entitet RrIzlazniceEO. Entitet RrIzlazniceEO

predstavljati će tablicu RR_IZLAZNICE u ADF "svijetu". Za svaku tablicu koje ćemo koristiti unutar aplikacije e-izlaznice postojati će njezin ADF entitet. Kao što je tablica predstavljena entitetom tako je i svaki redak tablice predstavljen jednom instancom pripadajućeg entiteta, isto tako i za svaku kolone tablice postoji atribut unutar entiteta. Ukoliko je model izvora podataka iz koje kreiramo ADF entitete dobro osmišljen tj. modeliran, struktura dijagrama klasa koji čine ADF entiteti i struktura ER dijagrama je gotovo ista.

Entitet objekti su nam osnova za izradu validacija tj. provjeru ispravnosti unesenih podataka, poslovnih pravila, upravljanje logikom dodavanja, ažuriranja i brisanja podataka na bazi te izradu dinamičkih kalkulacija i događaja kada dođe do promjene određenih podataka. Dogovoreno je da će se entitetima dodati sufiks EO (RrIzlazniceEO).

Entiteti će se izraditi za svaku tablicu koju ćemo koristiti u aplikaciji bez obzira da li ćemo direktno mijenjati podatke u njoj ili ćemo tablicu koristiti samo za čitanje podatka (tablice za liste vrijednosti eng. *LOV - List of values*).

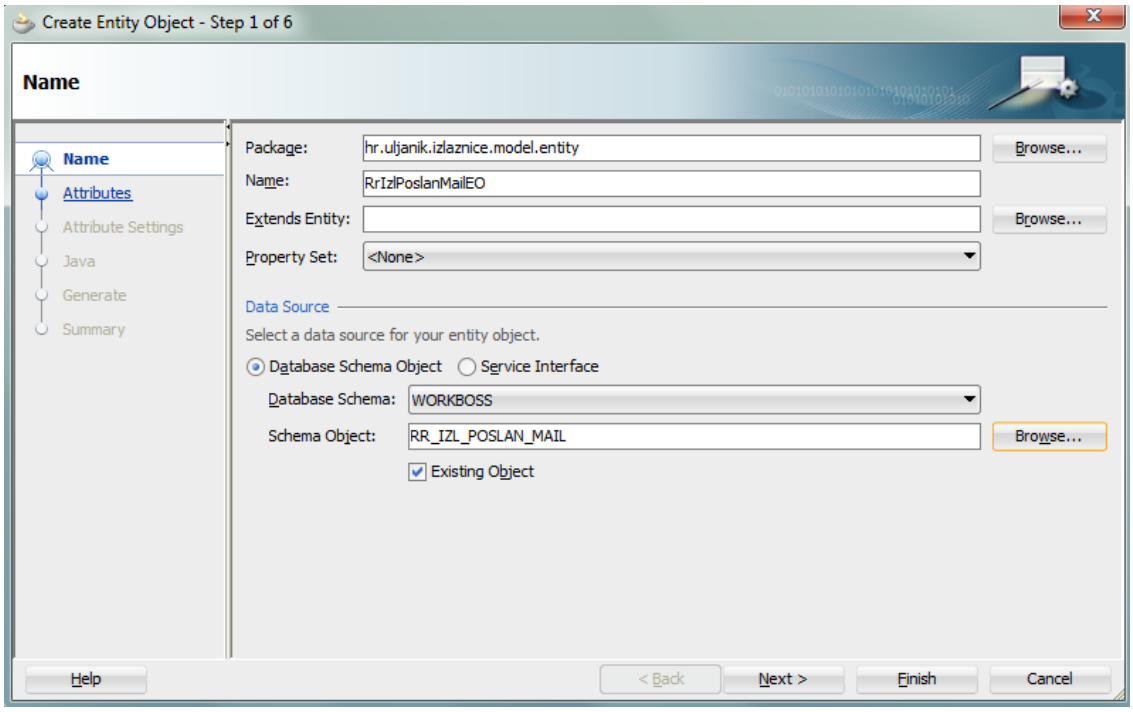
Za primjer pokazati ćemo izradu jednog entitet objekta. Zbog jednostavnosti i manjeg broja atributa uzeti ćemo tablicu RR_IZL_POSLAN_MAIL.

Izrada je podijeljena kroz nekoliko koraka:

- **Korak 1:** Definiranje imena objekta i tablice na kojoj je objekt baziran. (slika 20)

Radi bolje preglednosti i organizacije izrade aplikacije objekte smještamo u pakete. Pakete definiramo prema smislenim cjelinama i u njih smještamo objekte koji pripadaju toj cjelini. Tako za smještanje svih entitet objekata aplikacije e-izlaznice kreirali smo paket hr.uljanik.izlaznice.model.entity. Paket je imenovan prema Java konvenciji imenovanja tj. domena poduzeća (pisana odostraga - hr.uljanik), ime aplikacije (izlaznice), cjeline unutar aplikacije (model.entity).

Entitet objekt nazvali smo RrIzlPoslanMailEO. Kako je opisano ranije u poglavlju dogovoreno je da entitetima dodamo nastavak EO.

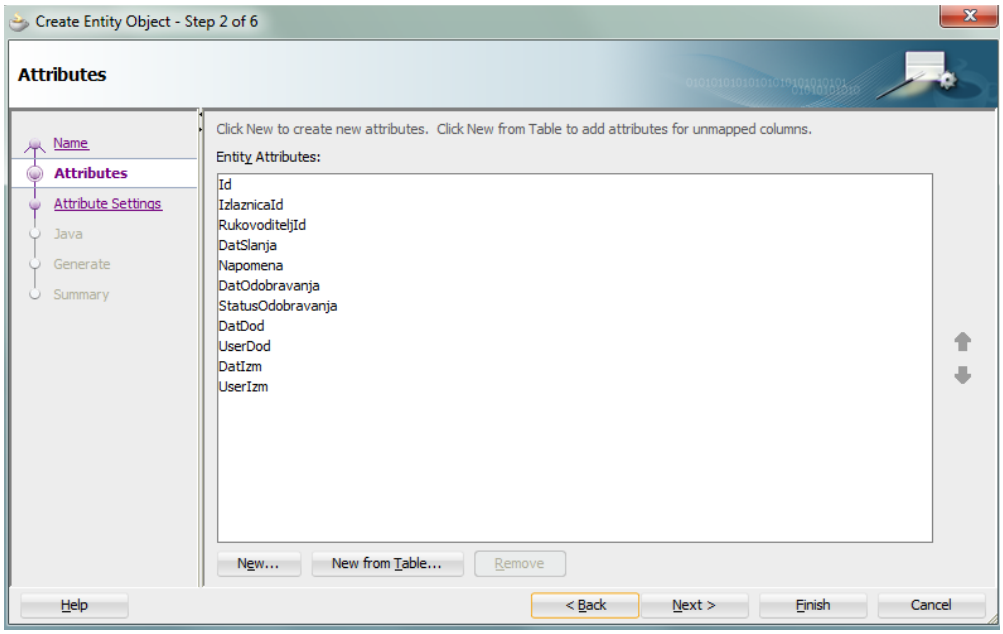


Slika 20. Definiranje imena entitet objekta

U donjem dijelu ekrana definiramo shemu i tablicu na kojoj je baziran entitet. U ovom slučaju shema je “WORKBOSS” , a tablica “RR_IZL_POSLAN_MAIL” .

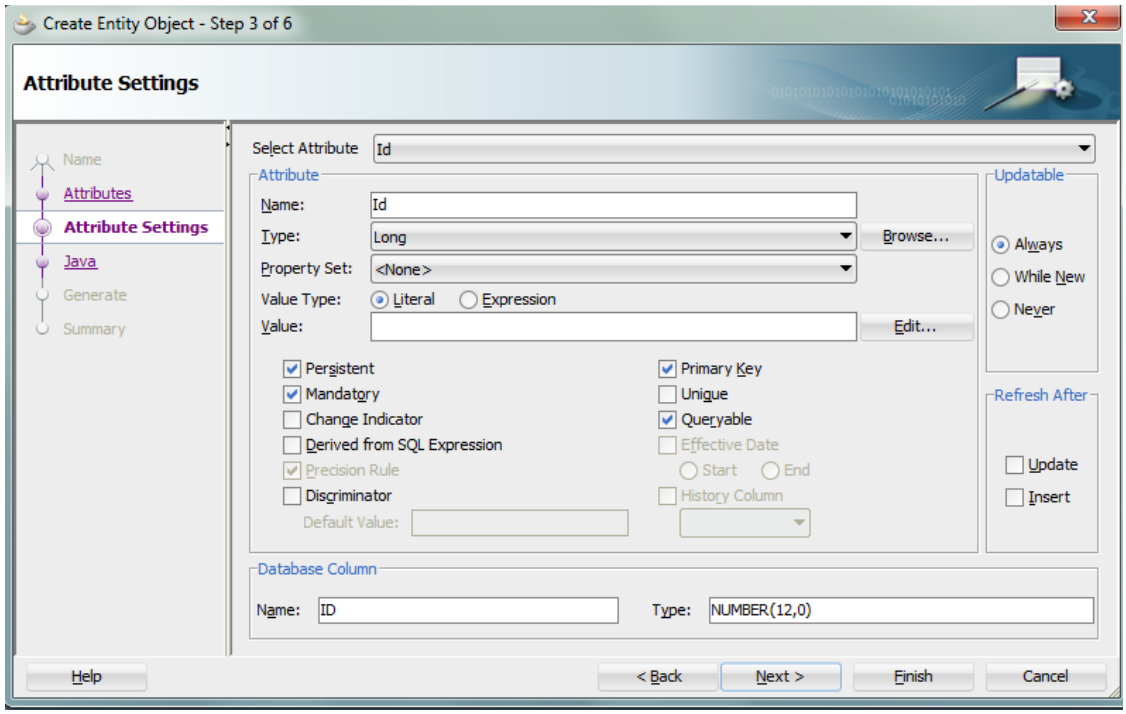
- **Korak 2:** Definiranje atributa (slika 21)

U ovom koraku definiramo atribute koje ćemo koristiti u entitetu. Ponuđeni su nam atributi tablice koju smo naveli u prethodnom koraku. Pored tih postojećih atributa možemo stvoriti i atribute koji nisu vezani za kolonu u tablici, a koristiti će nam za izradu logike aplikacije. U našem slučaju dovoljni su nam atributi definirani tablicom.



Slika 21. Definiranje atributa

- **Korak 3:** Definiranje svojstva atributa (slika 22.)

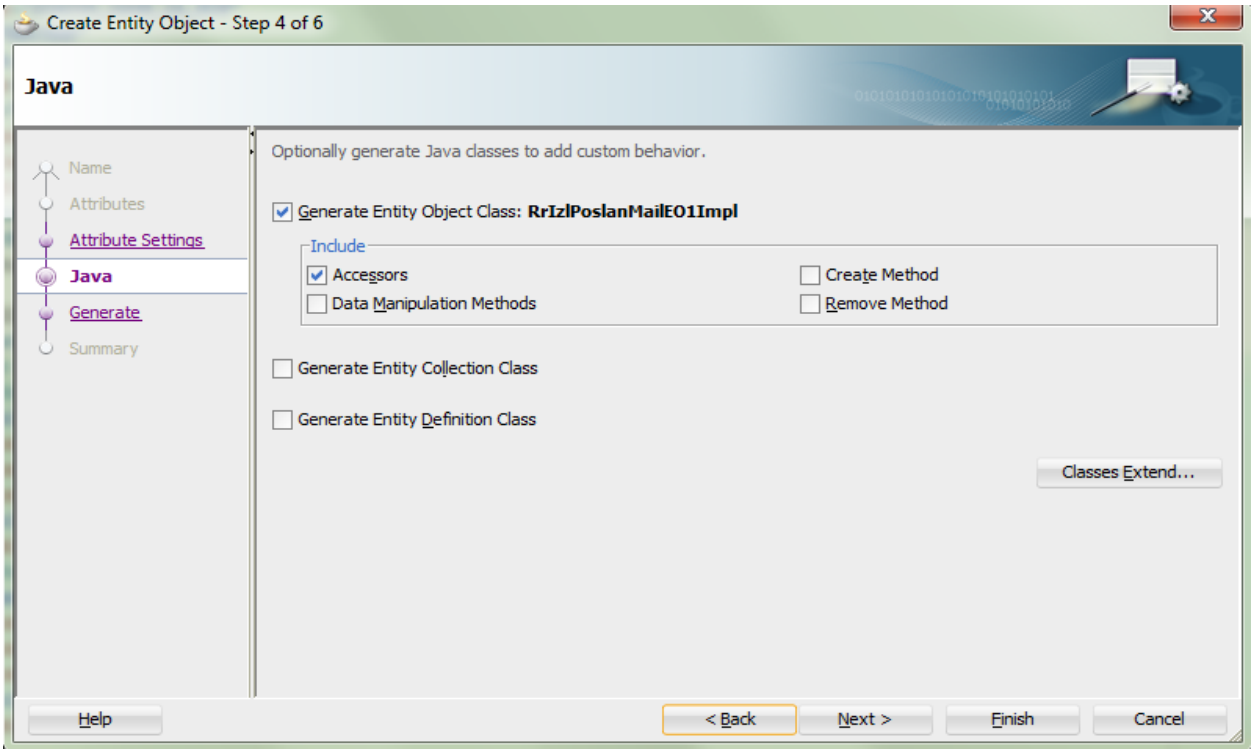


Slika 22. Definiranje svojstva atributa

U ovom koraku definiramo svojstva atributa, kao što su:

- kojeg je tipa atribut (integer, long, string, timestamp, itd.)
- da li je atribut obavezan
- da li je atribut primarni ključ
- da li se atribut može ažurirati (eng. *Updatable*)
- za koju kolonu tablice je atribut vezan
- da li možemo po tom atributu vršiti upit (eng. *Queryable*)
- da li se vrijednost atributa mora osvježiti nakon dodavanja ili ažuriranja

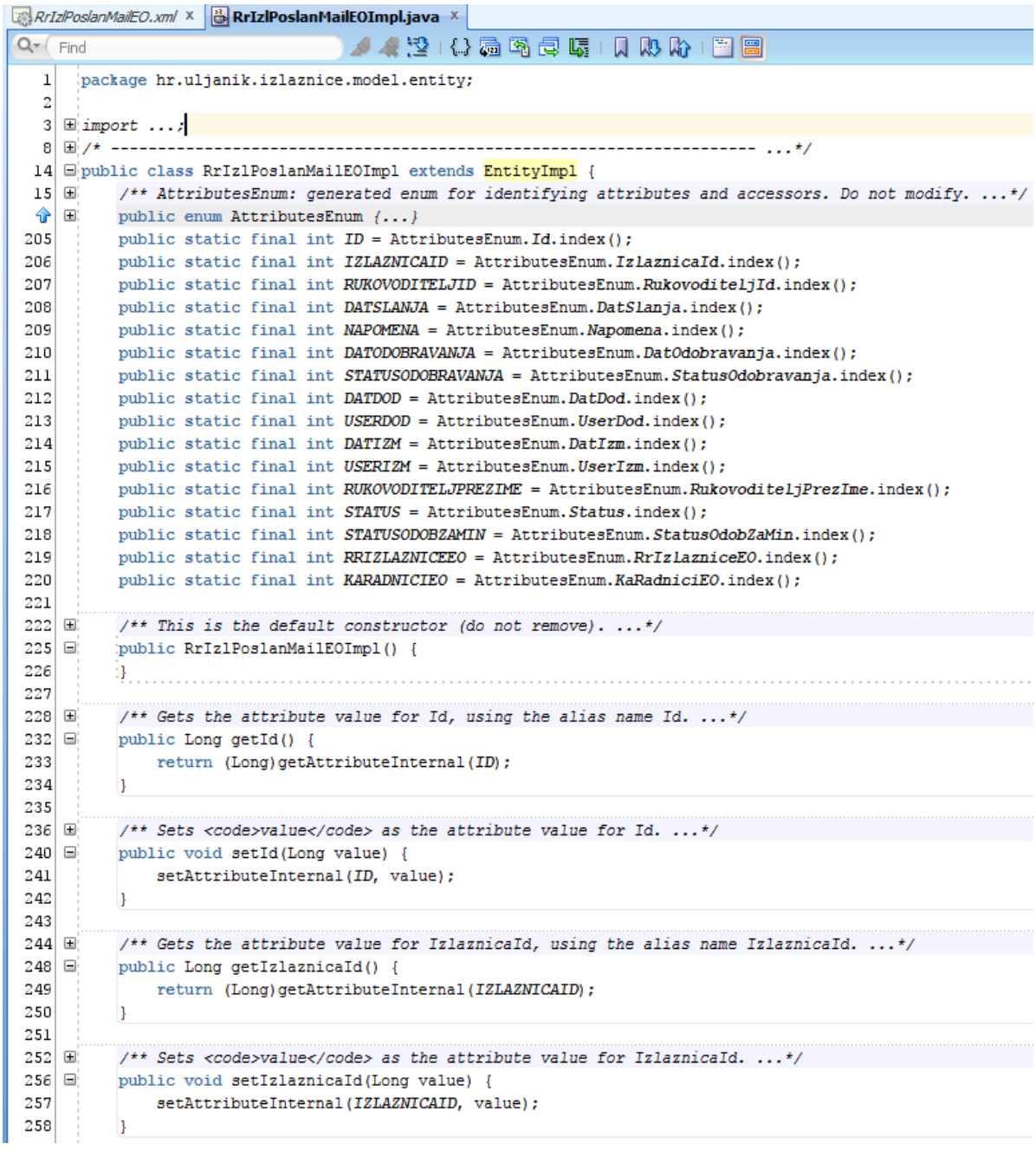
- **Korak 4:** Generiranje Java klase (slika 23.)



Slika 23. Generiranje Java klase

U ovom koraku definiraju se karakteristike generirane Java klase odnosno metode koje će Java klasa sadržavati.

Slijedeća dva koraka su nam informativnog karaktera tj. sumarni prikaz odabranih svojstva i generiranje entitet objekta. Nakon generiranja objekt se pojavljuje u navigatoru na lijevoj strani ekrana te ga možemo detaljnije uređivati. Uz entitet objekt generirana je i Java klasa. Dio te klase prikazan je na slici 24.



Slika 24. Generirana Java klasa

Iz slike vidimo da se unutar Java klase nalaze svi pripadajući atributi i da je za svaki atribut generirana metoda za dohvaćanje i postavljanje vrijednosti tog atributa (eng. *get and set*).

4.1.2 Izrada pogled objekata

ADF pogled objekti (eng. *view objects*) su poslovne komponente koje služe za prikupljanje podataka iz izvora podataka, oblikovanje podataka prilagođenih za prikaz te omogućuju korisniku izmjenu podataka. Pogled objekti mogu biti dinamički ili statički. Dinamički pogled objekti bazirani su na entitet objektima ili SQL upitu, a statički pogled objekti bazirani su na fiksnim vrijednostima. U ovom radu korištena su oba tipa pogled objekata. Za svaki entitet objekt generiran je njegov pogled objekt. Pogled objekti bazirani na SQL upitu kao što ćemo vidjeti kasnije korišteni su za prikaz i dohvaćanje rukovoditelja radnika i za određivanje tipa korisnika. Statički pogled objekti korišteni su definiranje liste vrijednosti za tip izlaza (P - privatni izlaz, S - službeni izlaz).

Pomoću pogled objekata definiramo korisnički prikaz npr. koji atributi će biti vidljivi korisniku, na koji način će biti formatirani podaci. Također, unutar pogled objekata možemo definirati i dodatne atribute koji će biti rezultat neke funkcije, a možemo definirati i metode koje će izvršavati neke kompleksne operacije nad podacima.

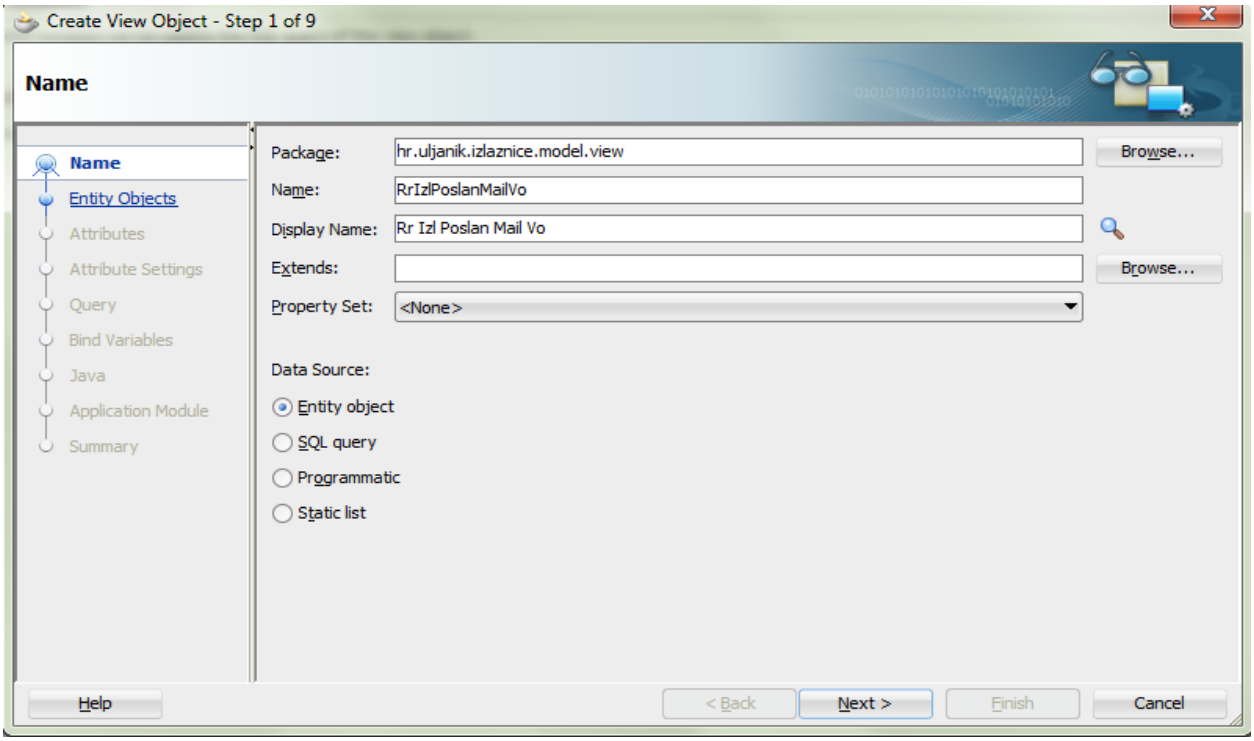
Kroz slijedećih nekoliko koraka pokazati ćemo izradu pogled objekata za primjer iz prethodnog podpoglavlja tj. za entitet RrIzlPoslanMailEO.

Koraci izrade:

- **Korak 1:** Definiranje imena pogled objekta (slika 25)

U ovom koraku definirano paket u kojem će se nalaziti pogled objekt. Smjestiti ćemo ga u paket hr.uljanik.izlaznice.model.view. Entitet objekte smještali smo u hr.uljanik.izlaznice.model.entity, a pogled objekte smještati ćemo u paket hr.uljanik.izlaznice.model.view. Kod velikog broja

objekata takva organizacija omogućuje nam brže snalaženje prilikom razvoja i održavanja aplikacije.

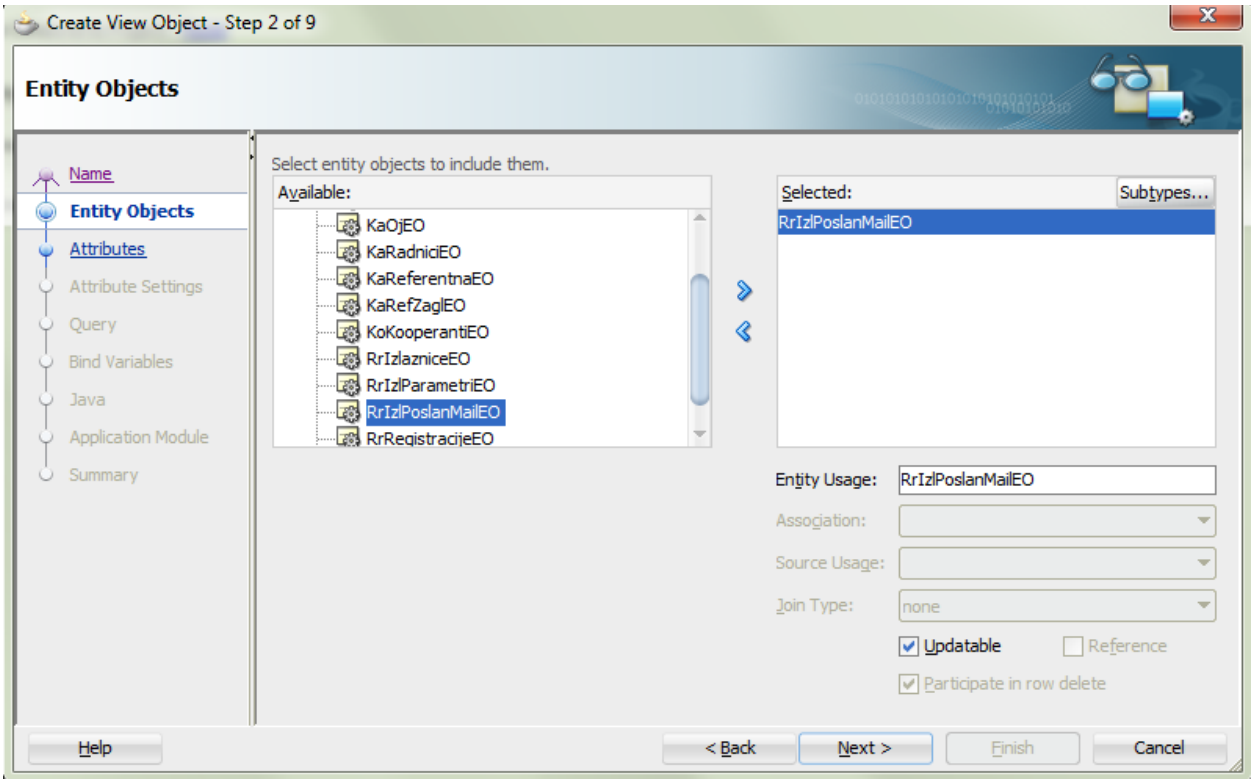


Slika 25. Definiranje pogled objekta

U slijedeće polje na ekranu upisujemo ime pogled objekta. To je u ovom slučaju RrIzlPoslanMailVO. Kao i kod entitet objekata dogovoreno je da pogled objekti imaju nastavak VO prema engleskom nazivu "*View object*".
Nadalje izabiremo izvor podataka na kojem će biti bazirani pogled objekt, u ovom slučaju to će biti entitet objekt.

- **Korak 2:** Izbor izvornog objekta (slika 26)

U ovom ekranu izabiremo izvor podataka odnosno u ovom slučaju entitet objekt na kojem će biti baziran pogled objekt. Ako se radi o pogled objektu baziranom na SQL upitu u ovom koraku upisujemo upit, a ako se radi o objektu baziranom na statičkoj listi, ovdje upisujemo statičke vrijednosti.



Slika 26. Izbor izvornog objekta

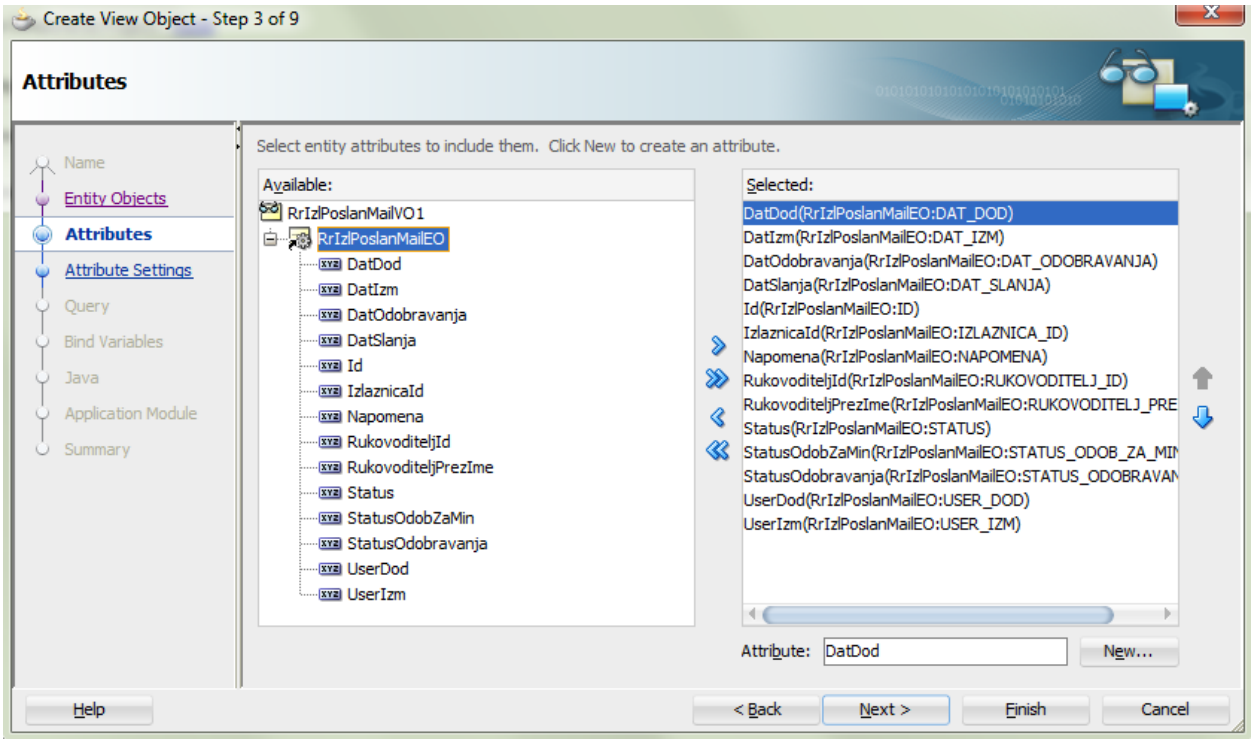
Pogled objekt može biti baziran na više od jednog entitet objekta. U tom slučaju na slici 26. imali bi sa desne strane više entitet objekata te bi morali definirati vezu između njih. Na to možemo gledati kao na spajanje dviju ili više tablica unutar SQL upita.

- **Korak 3:** Izbor atributa pogled objekta (slika 27.)

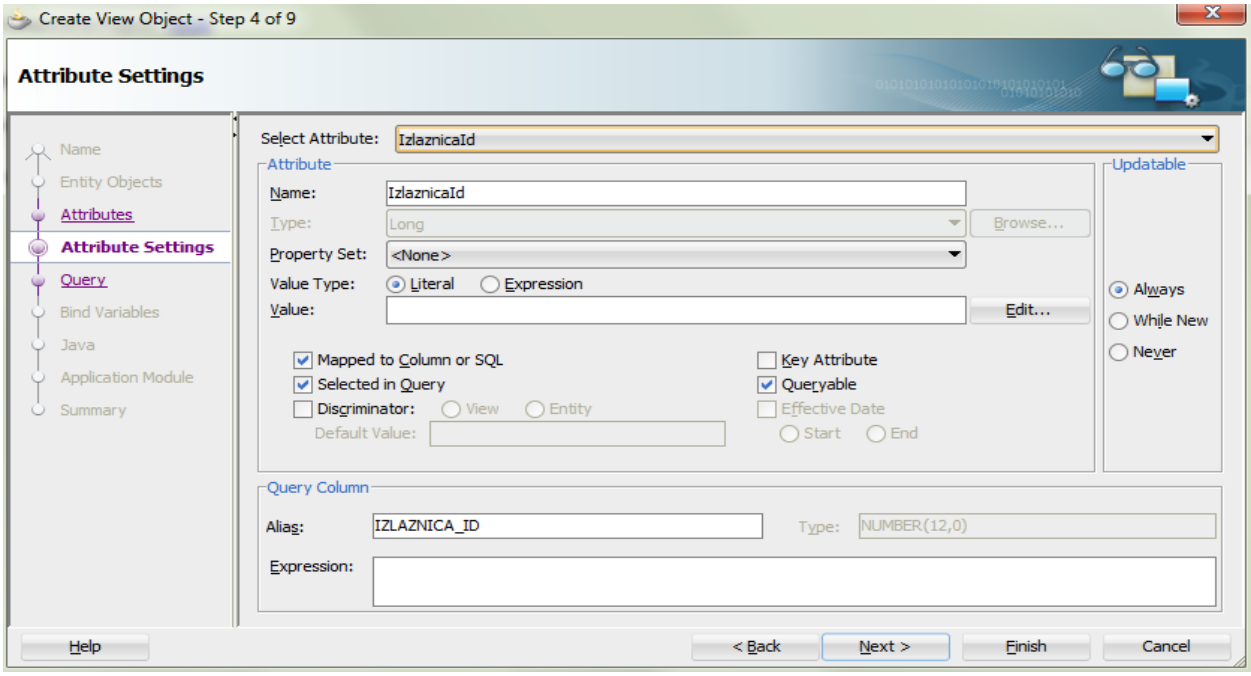
U ovom koraku definiramo koji će se atributi koristi u pogled objektu. Unutar pogled objekta možemo dodati sve atribute koje je imao pripadajući entitet odnosno SQL upit ili statička lista ili možemo definirati neki novi atribut (tipka *"New"* dole desno) koji je rezultat neke funkcije.

- **Korak 4:** Postavke atributa (slika 28.)

U ovom koraku postavljamo svojstva atributa. Ovdje postavljamo:



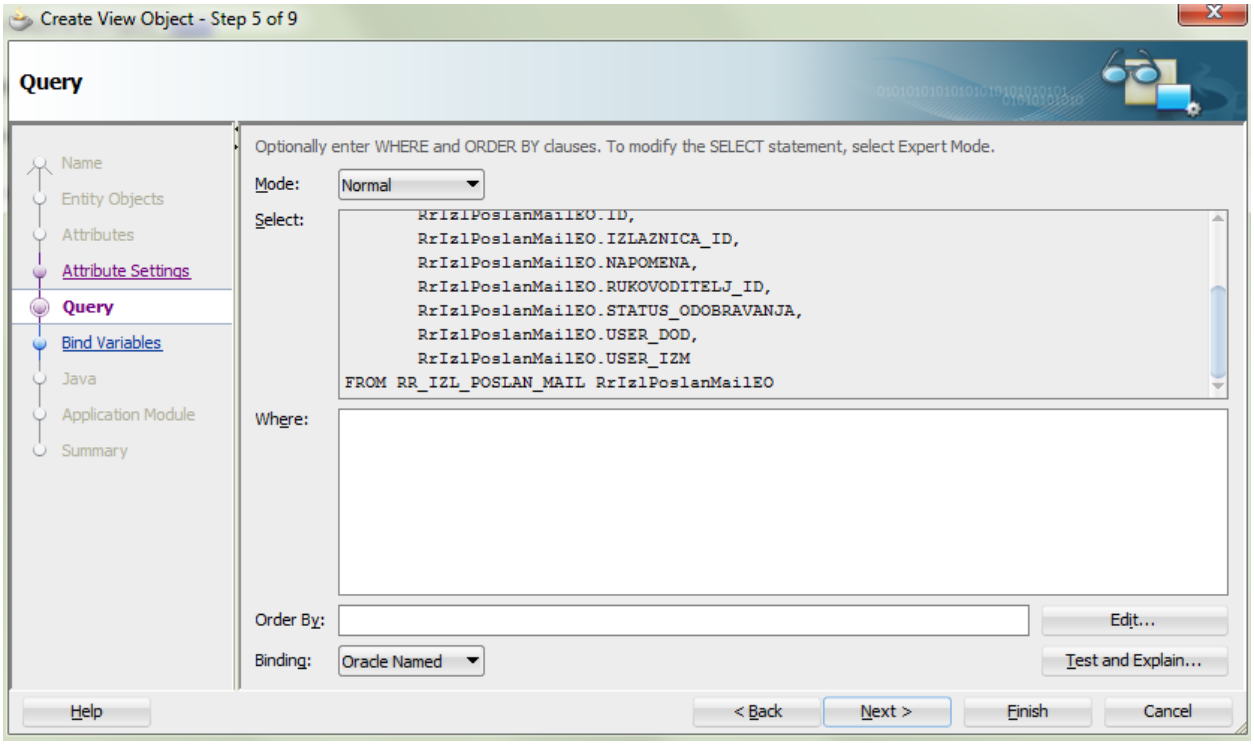
Slika 27. Izbor atributa



Slika 28. Postavke atributa

- da li se atribut može ažurirati
- da li je atribut vezan za neki atribut iz entitet objekta ili je rezultat neke funkcije
- da li se po određenom atributu može pretraživati

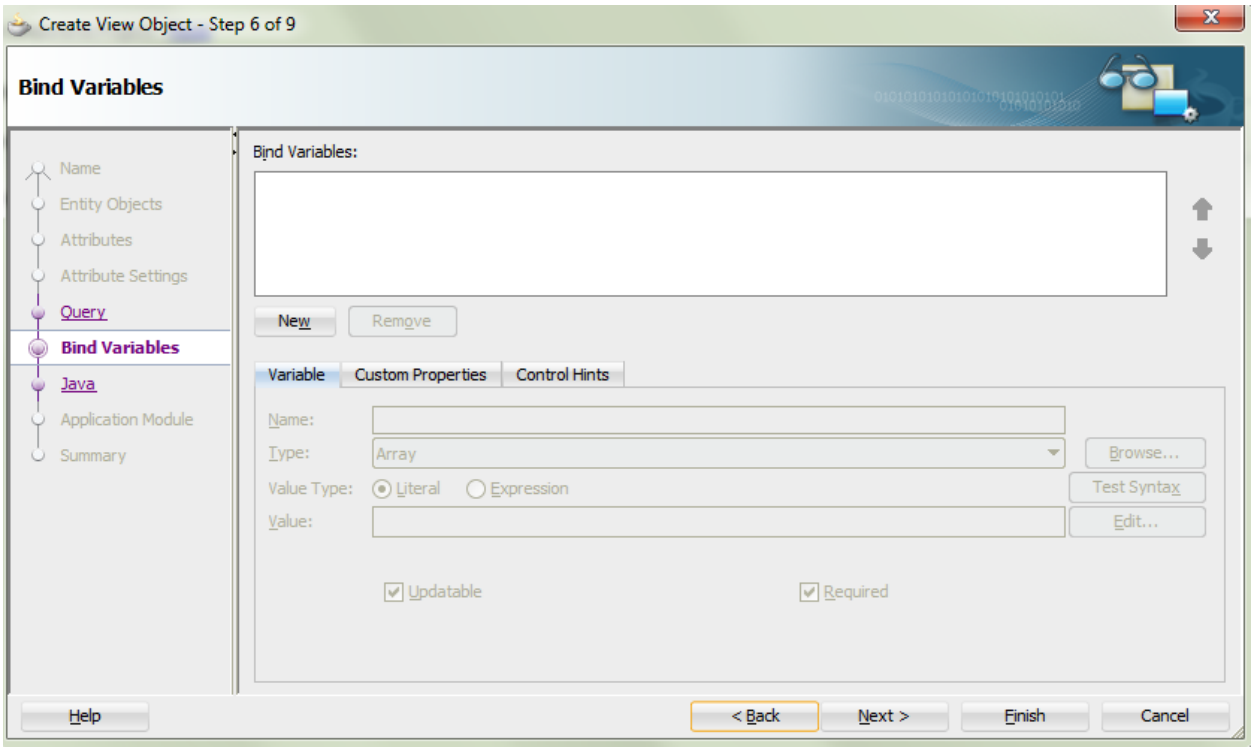
- **Korak 5:** Ograničavanje prikaza i postavljanje sortnog pojma (slika 29)



Slika 29. Postavljanje uvjeta prikaza

Ako postoji potreba za ograničavanjem prikaza tj. za sužavanjem rezultirajućeg seta podataka npr. želimo prikazivati samo odobrene poruke. To podešavamo u ovom koraku. Također ovdje podešavamo i ako želimo po nekim atributima sortirati izlazni prikaz. Za sužavanje rezultirajućeg seta podataka koristimo polje "*Where*", a za sortiranje prikaza koristimo polje "*Order by*". Sintaksa sa kojom popunjavamo ta polja istovjetna je Oracle SQL sintaksi.

- **Korak 6:** Postavljanje parametra (slika 30.)

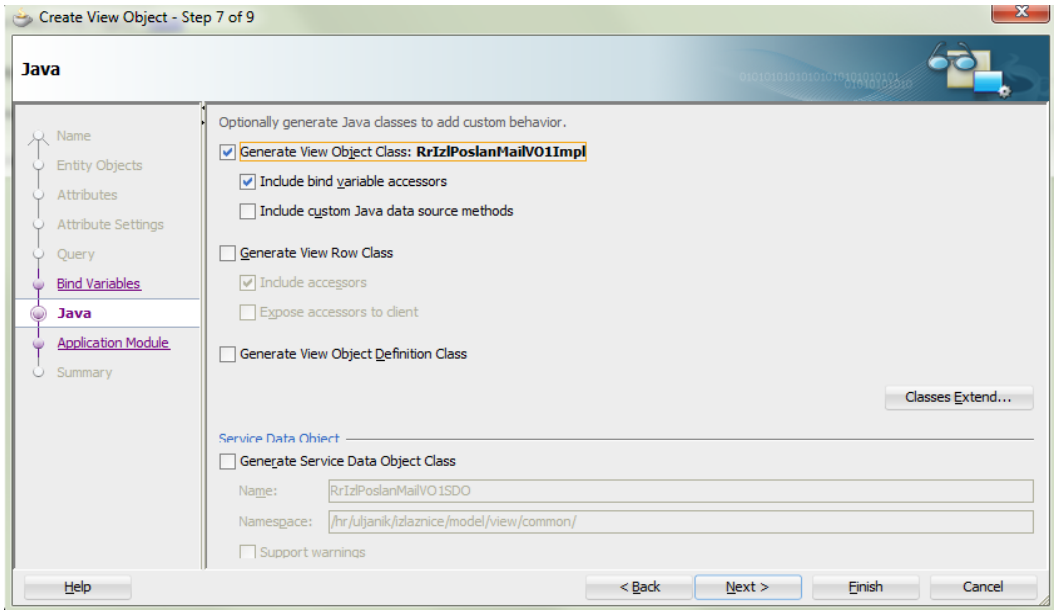


Slika 30. Postavljanje parametra

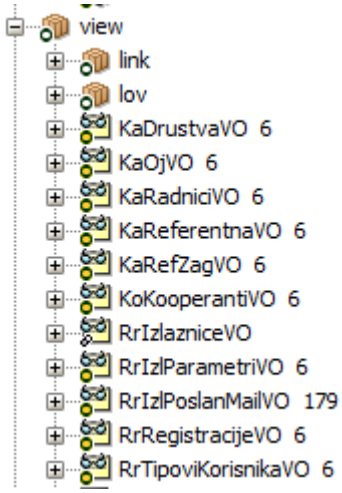
U ovom ekranu ako postoji potreba definiramo parametar koji će pogled objekt primati, te sa tim parametrom ograničavamo izlazni set podataka (koristimo ga u uvjetu). U ovom konkretnom slučaju nemamo potrebe za uvođenjem parametra, ali prilikom izrade pogled objekta RrIzlazniceVO uvesti ćemo parametar koji će nam služiti za ograničavanje podataka za točno određenog korisnika, budući da je definirano da korisnik može vidjeti samo svoje izlaznice.

- **Korak 7:** Parametri generiranja Java klase (slika 31.)

Ovdje kao i kod kreiranja entitet objekata izabiremo postavke generiranja pripadajuće Java klase.



Slika 31. Generiranje Java klase



Slika 32. Pogled objekti aplikacije e-izlaznice

U slijedeća dva koraka vidimo sumarne informacije prethodnih koraka, te slijedi generiranje pogled objekta. Isti postupak ponovili smo i za ostale pogled objekte koje ćemo koristiti u aplikacije e-izlaznice (slika 32).

4.1.3 Validacije

Implementacija poslovnih pravila vrlo je bitan dio procesa u izradi aplikacije zato što o njemu ovisi ispravnost podataka unutar aplikacije. Validacija je upravo to, kontrola ispravnosti unesenih podataka. Poslovna pravila su često dvosmislena ili nejasno definirana, ali su jako bitna za uspjeh cijelog projekta te je implementacija poslovnih pravila glavni zadatak većine projekata razvoja aplikacija.

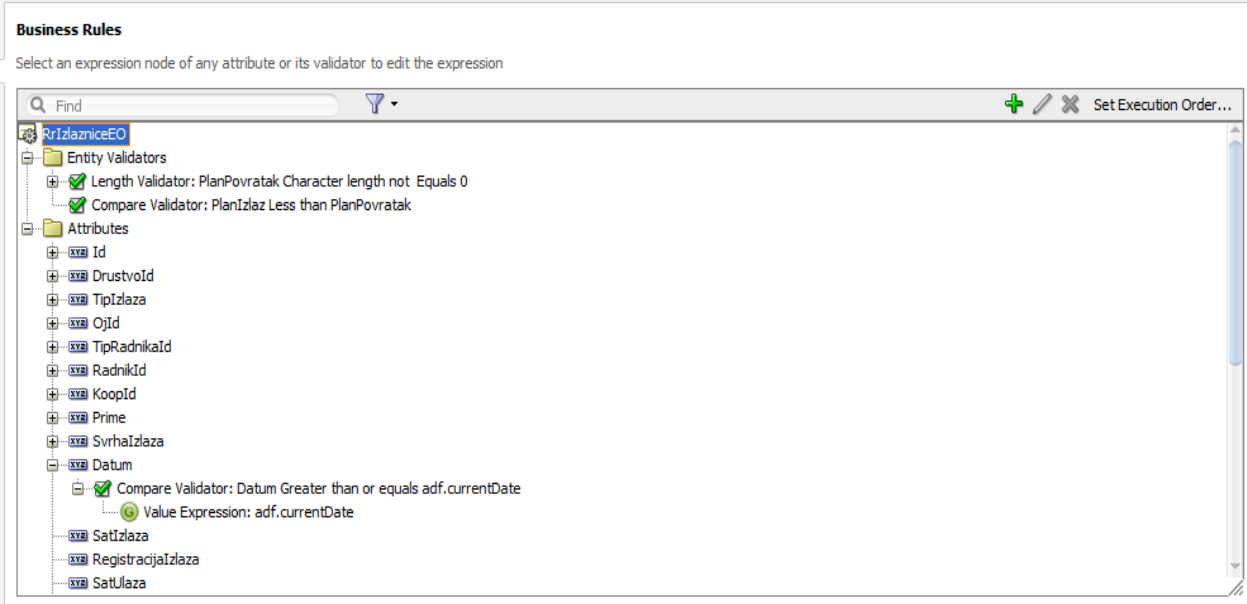
U Oracle ADF-u validacija se implementira najčešće u model sloju. Implementacija poslovnih pravila može biti deklarativna kroz već pripremljene ekrane JDeveloper-a ili može biti ugrađena u kod Java klasa generiranih entiteta ili pogled objekata. U ovom radu koristimo oba načina izrade validacija. Validaciju u ADF model sloju vežemo za određeni atribut odnosno polje na određenoj stranici, kada korisnik to polje promjeni i potvrdi promjene okida se validacija kroz niz pravila i uvjeta koje smo naveli prilikom izrade validacije. Ako uneseni podatak ne zadovolji neko od unesenih pravila ili uvjeta, poruka sa greškom pojavljuje se na ekranu.

Neka poslovna pravila riješili smo samim modelom baze podataka. Ako se prisjetimo korisničkog zahtjeva iz prvog poglavlja vidimo da je jedno od poslovnih pravila bilo to da izlaznica može biti ili privatna (P) ili službena (S). To poslovno pravilo implementirano je kroz dizajn baze podataka kao ograničenje (eng. *check constraint*) tako da je na polju TIP_IZLAZA u tablici RR_IZLAZNICE stavljeno ograničenje da vrijednost polja može biti P ili S. Da bi olakšali korisniku unos toga polja, a samim tim i na tom sloju ograničili unos na P ili S kao komponentu za unos toga polja izabrana je padajuća lista (eng. *list item*) sa vrijednostima P i S kao što je vidljivo iz prototipa izgleda (slika 7).

Neka pravila nisu eksplicitno navedena, ali su sama po sebi logična npr. da datum izlaza ne može biti veći ili jednak datumu povratka.

Poslovna pravila mogu se implementirati na razini atributa ili na razini entiteta. Na razini atributa definiramo pravila koja su vezana za taj atribut i njihova validacija izvršava se odmah prilikom

promjene vrijednosti polja. Na razini entiteta definiramo validacije koje ovise o više atributa i provjera pravila se pokreće prilikom potvrde i spremanja promjena. Na slici 33. vidimo pravila implementirana na razini entiteta (eng. *Entitiy Validators*) i pravilo implementirano na razini atributa (atribut datum).

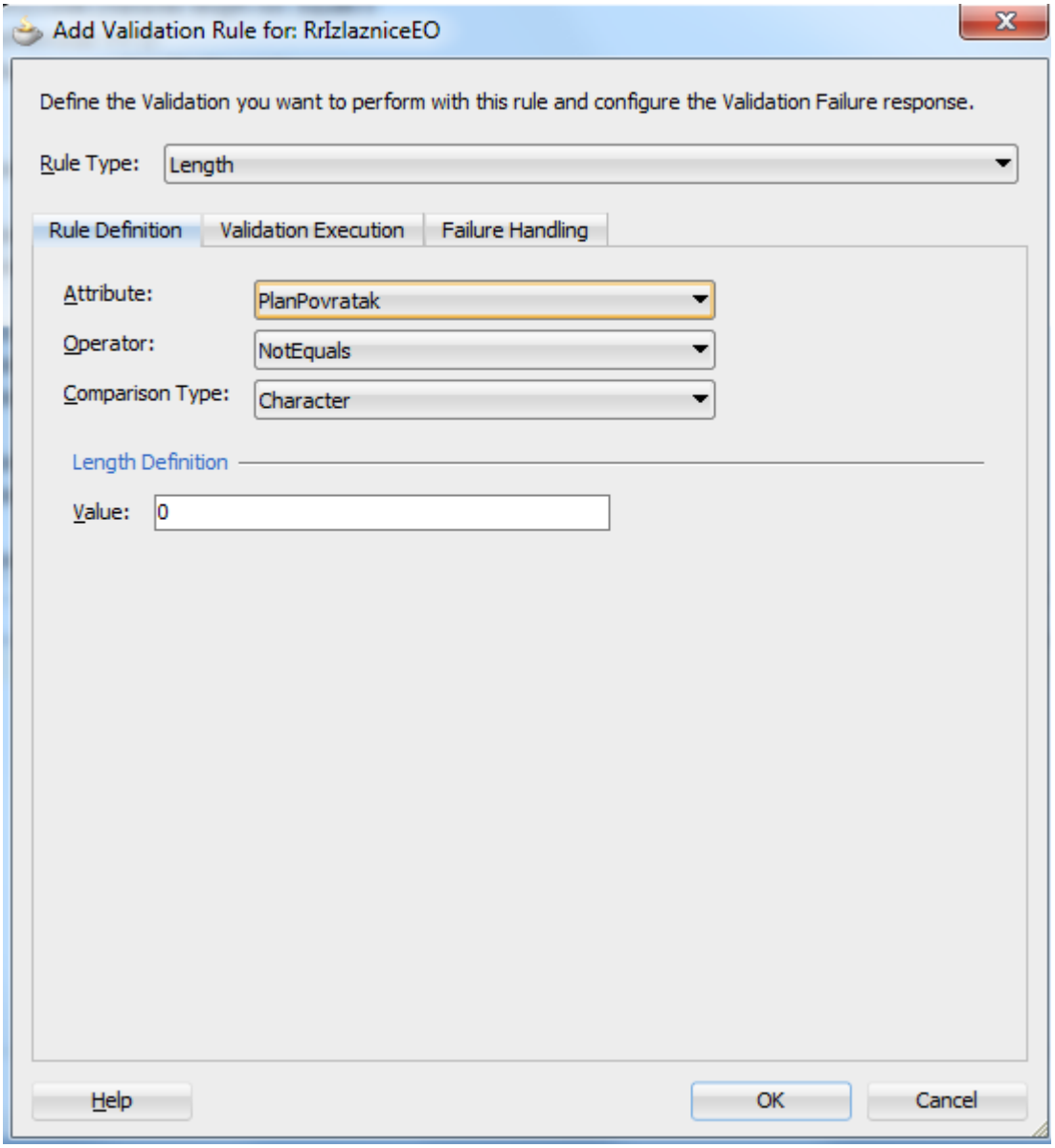


Slika 33. Poslovna pravila RrIzlazniceEO

Popis pravila koja su implementirana u aplikaciji e-izlaznice:

1. Tip izlaza P ili S - privatno ili službeno
2. Vrijeme planiranog izlaza treba biti manje od vremena planiranog povratka
3. Ako nije unesen planirani povratak onda je indikator povratka D (Da)
4. Vrijeme za koji se unosi izlaznica mora biti veće od trenutnog vremena
5. Ako je tip izlaza S - službeni, mora biti navedena svrha izlaza
6. Ne mogu se brisati izlaznice manje od tekućeg datuma tj. već iskorištene izlaznice
7. Ne mogu se ažurirati izlaznice koje su kod rukovoditelja na odobravanju ili koje su već odobrene
8. Ako je indikator planiranog povrataka N (nije planiran povratak) onda popuniti planirani povratak sa tekućim datumom, a za vrijeme staviti 15:00 npr.15.09.2014 15:00

U nastavku ćemo pokazati izradu nekih tipova validacija korištenih u ovom radu.



Slika 34. Validacija za polje planirani povratak

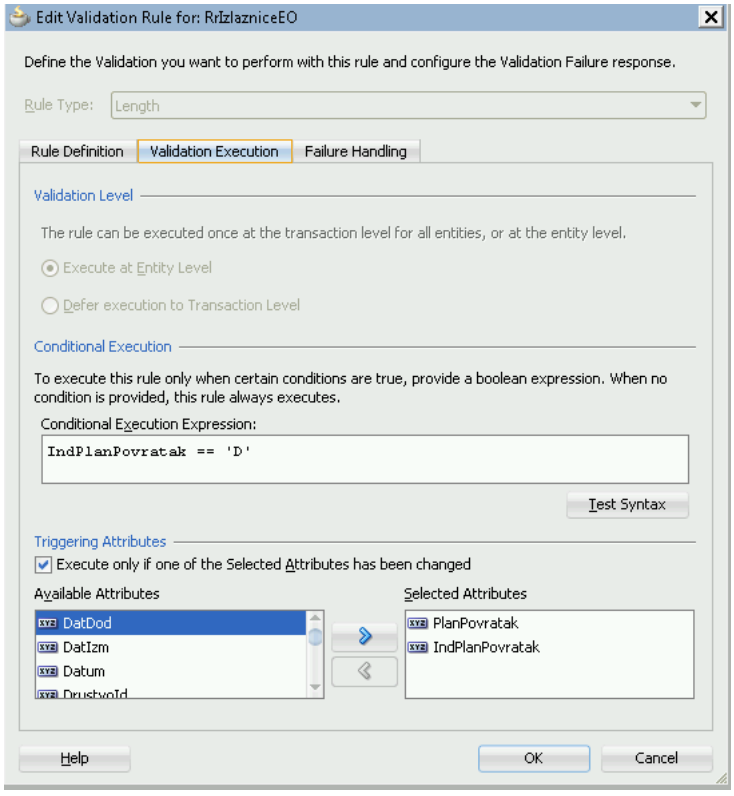
Izrada validacije započinje izborom tipa pravila (slika 34). Postoje nekoliko vrsta pravila odnosno načina na koji možemo testirati korisnikov unos. Vrsta pravila su:

- usporedba sa nekom vrijednošću odnosno drugim poljem
- duljina unesene vrijednosti

- lista vrijednosti
- poziv Java metode u čijem kodu vršimo provjere
- pojas vrijednosti npr. sat treba biti između 0 - 23

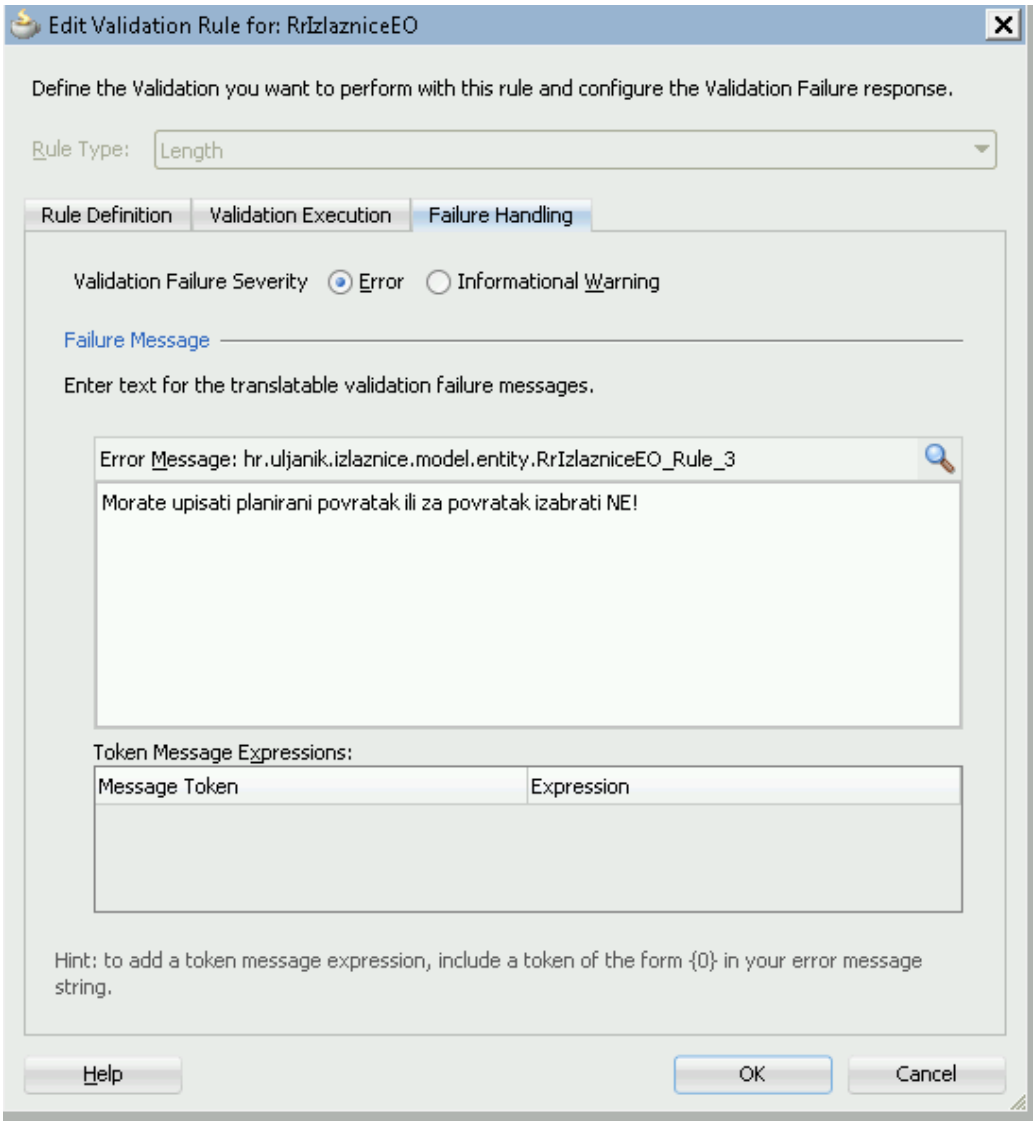
U ovom slučaju kod izrade validacije za polje planirani povratak koristiti ćemo testiranje na duljinu da vidimo da li je korisnik unio podatak o planiranom povratku. Uspoređujemo sa vrijednošću 0. Ako je korisnik unio podatak duljina će biti veća od 0, a ako nije duljina će biti jednaka 0.

U slijedećem koraku (slika 35.) izabiremo pod kojim uvjetima se vrši provjera. Upisuje se pravilo koje treba biti logički istinito da bi se provjera izvršavala. U konkretnom slučaju zanima nas da li je korisnik nešto unio tj. ako je postavio oznaku na indikator povratka (zadnje polje na slici 7.). U donjem dijelu ekrana odabiremo atribute za koje će se provjera okidati kada se njihove vrijednosti promjene. Kako su polja planirani povratak i indikator povratka ovisna jedna o drugom, njih stavljamo na popis atributa koji će okidati provjeru.



Slika 35. Uvjet okidanje provjere

U slijedećem koraku (slika 36) postavljamo tip upozorenja i sadržaj poruke koji će korisnik vidjeti na ekranu ako se ne zadovolji uvjet iz prvog koraka. Poruka može informativnog karaktera ili može biti greška. Poruke informativnog karaktera pojavljuju se na ekranu i omogućavaju korisniku daljnji rad. Dok poruke o grešci zaustavljaju rad korisnika, te se sa radom može nastaviti tek nakon što se greška ispravi. U našem primjeru ako je korisnik označio da se planira vratiti, a nije unio podatak (vrijeme) o planiranom povratku javlja mu se poruka na ekran "Morate upisati planirani povratak ili za povratak izabrati NE!", odnosno ne može nastaviti sa radom dok ne upiše vrijeme povratka.



Slika 36. Izbor vrste pogreške i sadržaj poruke

Ovaj tip izrade validacije naziva se deklarativni tip. Kao što smo vidjeli za definiranje validacije postoji nekoliko ekrana, te se kroz te ekrane definira pravilo "okidanja" greške, tip greške i pripadajuća poruka. Time je izrađena validacija odnosno implementirano poslovno pravilo bez programiranja. Na ovakav način ugrađuju se jednostavnija poslovna pravila. Postoji i implementacija poslovnih pravila kroz kod aplikacije, točnije u konkretnom slučaju u Java klasu entitet objekta. U primjeru aplikacije e-izlaznice, pravilo pod broj 6 implementirano je na taj način. Pravilo glasi: "Ne mogu se brisati izlaznice manje od tekućeg datuma tj. već iskorištene izlaznice". Ova kontrola ne odnosi se na unos izlaznice nego kao kontrola kada se izlaznica pokuša obrisati. Kod generiranja entitet objekta kako je ranije u poglavlju opisano, generirali smo i pripadajuću Java klasu. JDeveloper nam je automatski kod generiranja kreirao i između ostalog, metode koje se pozivaju prilikom dodavanja, ažuriranja i brisanje slogova. Mijenjajući te metode možemo utjecati na ponašanje tih akcija. Na taj način ugraditi ćemo kontrolu na akciju brisanja, odnosno provjeru datuma da vidimo da li se taj slog zaista može obrisati ili ne zadovoljava zadano pravilo.

Generirana metoda koja se poziva kod brisanja izgleda ovako:

```
public void remove() {
        super.remove();
}
```

U tu metodu ugrađujemo kontrolu provjere datuma i nakon promjene metoda za brisanje "*remove*" izgleda ovako:

```
public void remove() {
    Date date = new Date();
    Timestamp datumTs = getDatum();
    SimpleDateFormat formatter = new SimpleDateFormat("yyyy-MM-dd");
    Date date1 = new Date();
-
```

```
        try {
            date = formatter.parse(formatter.format(date1));
        } catch (ParseException e) {
        }
        Date datum = null;
        try {
            datum = datumTs.dateValue();
        } catch (SQLException f) {
        }
        if (datum.compareTo(date) < 0) {
            throw new JboException("Nije dozvoljeno brisanje izlaznica za datume manje od tekućeg
datuma!");
        } else {

            super.remove();
        }
    }
```

Učitavamo tekući datum, te ga formatiramo na oblik u kojem je vrijednost datuma spremljena u entitet objektu. Nakon toga dohvaćamo datum tekućeg sloga, te ga uspoređujemo sa tekućim (današnjim) datumom. Ako je dohvaćeni datum manji od tekućeg datuma na ekranu prikazujemo odgovarajuću poruku, a ako nije brišemo slog iz baze.

Na ovakav način, direktno utječući na programski kod možemo implementirati kompleksna poslovna pravila, pozivajući ih u odgovarajućem trenutku.

Također i ostala poslovna pravila implementirana su u aplikaciju koristeći deklarativni i programski pristup.

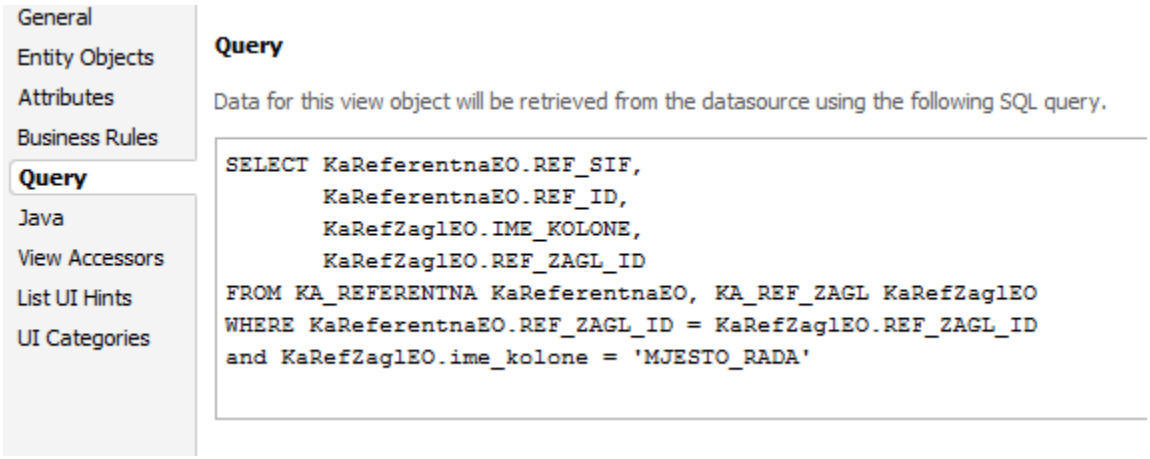
4.1.4 Liste vrijednosti

Lista vrijednosti je lista koja nam ograničava izbor pojedinog atributa tablice na dozvoljene vrijednosti. Na primjer atribut SPOL možemo ograničiti na muško/žensko. Liste vrijednosti najčešće se vežu za atribute tablice koje predstavljaju veze na druge tablice odnosno strane ključeve (eng. *foreign key*) ili na atribute tablice nad kojima postoje ograničenja.

Postoje dvije vrste lista vrijednosti:

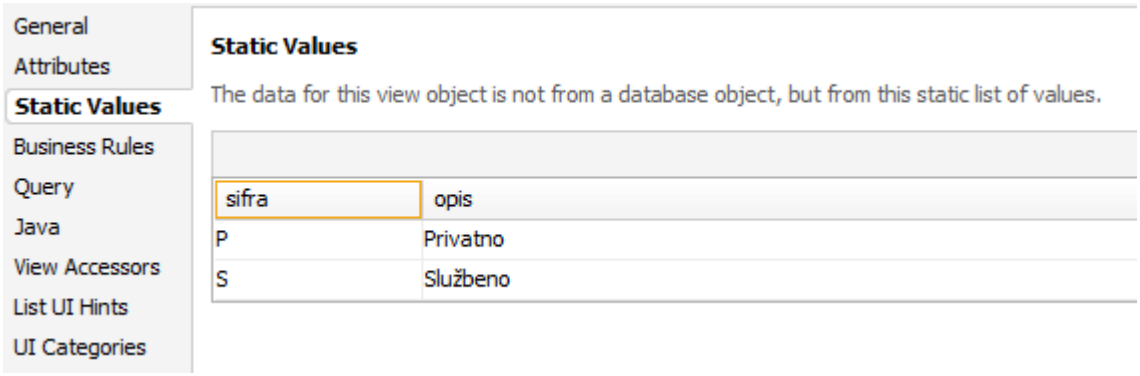
1. Dinamičke liste vrijednosti
2. Statičke liste vrijednosti

Dinamičke liste vrijednosti ovisne su o upitu prema bazi podataka. Njihov sadržaj je upravo rezultat upita. Takve liste vezane su za strane ključeve tablice te se u njima nalazi upit prema tablici na koju pokazuje strani ključ (slika 37.)



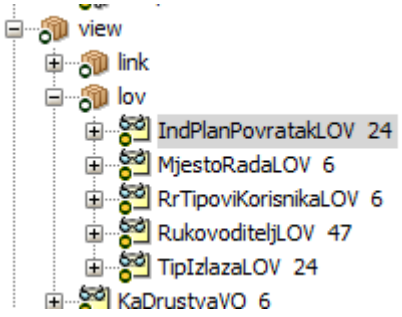
Slika 37. Upit dinamičke liste vrijednosti (lista vrijednosti mjesta rada)

Statička lista vrijednosti je lista u kojoj se vrijednosti sa vremenom ne mijenjaju odnosno statične su. Te vrijednosti definiramo kod izrade liste i takve liste vežemo uz atribute tablice nad kojima su definirana ograničenja (slika 38.).



Slika 38. Vrijednosti statičke liste vrijednosti (lista vrijednosti tipa izlaza)

U Oracle ADF-u liste vrijednosti definiramo kao pogled objekte koje vežemo za atribut tablice. U praksi, radi bolje organizacije projekta najbolje je liste vrijednosti staviti u poseban paket unutar pogled objekata i nazvati ih sa sufiksom LOV od engleskog naziva "*list of values*" kao što je prikazano na slici 39.

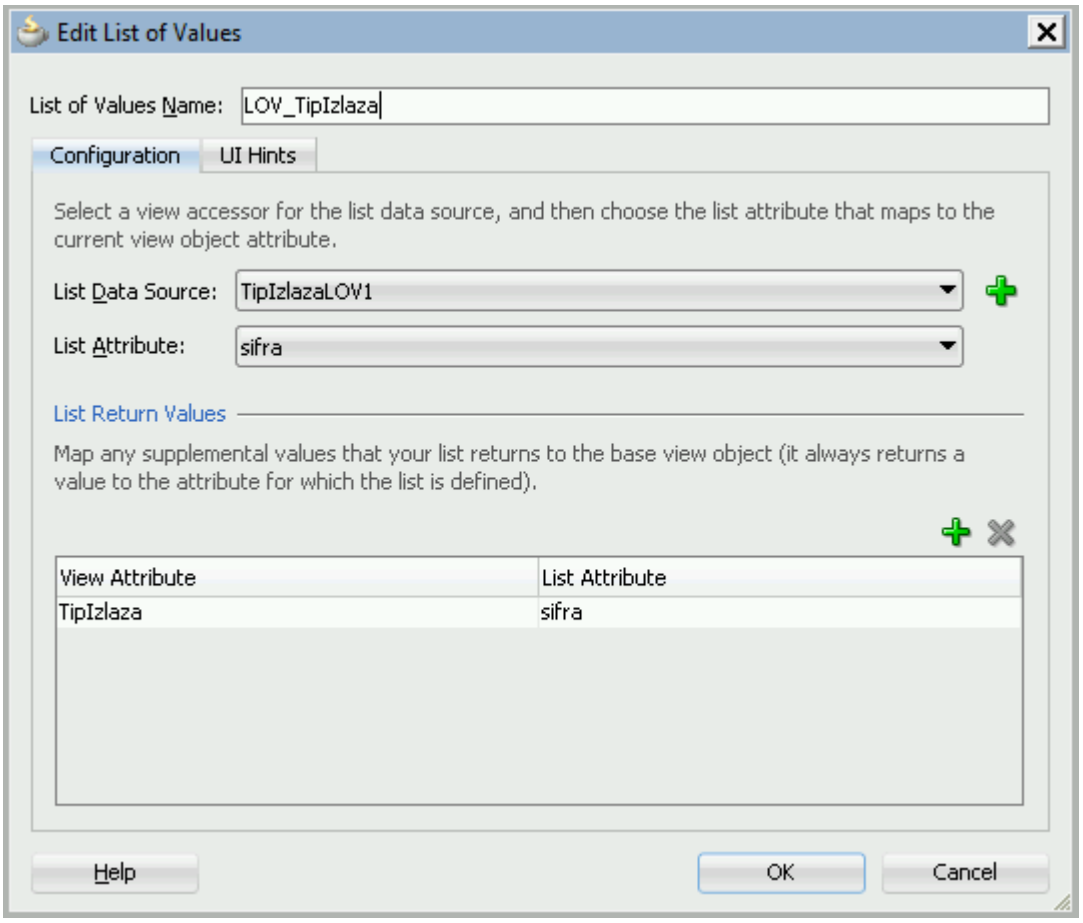


Slika 39. Organizacija listi vrijednosti unutar pogled objekata

Obzirom da su liste vrijednosti pogled objekti njih izrađujemo na isti načina kao što se izrađuju pogled objekti opisanih u potpoglavlju "Izrada pogled objekata". Razlika je samo u tome što tu listu vežemo za određeni atribut unutar pogled objekata.

Slijedi prikaz vezivanja liste vrijednosti za atribut pogled objekata. U konkretnom primjeru biti će pokazano vezivanje liste vrijednosti za atribut TipIzlaza. Tip izlaza ima dvije dozvoljene vrijednosti P - privatno i S - službeno. Nakon kreiranja statičkog pogled objekata sa dozvoljenim vrijednostima listu vežemo za atribut na slijedeći način:

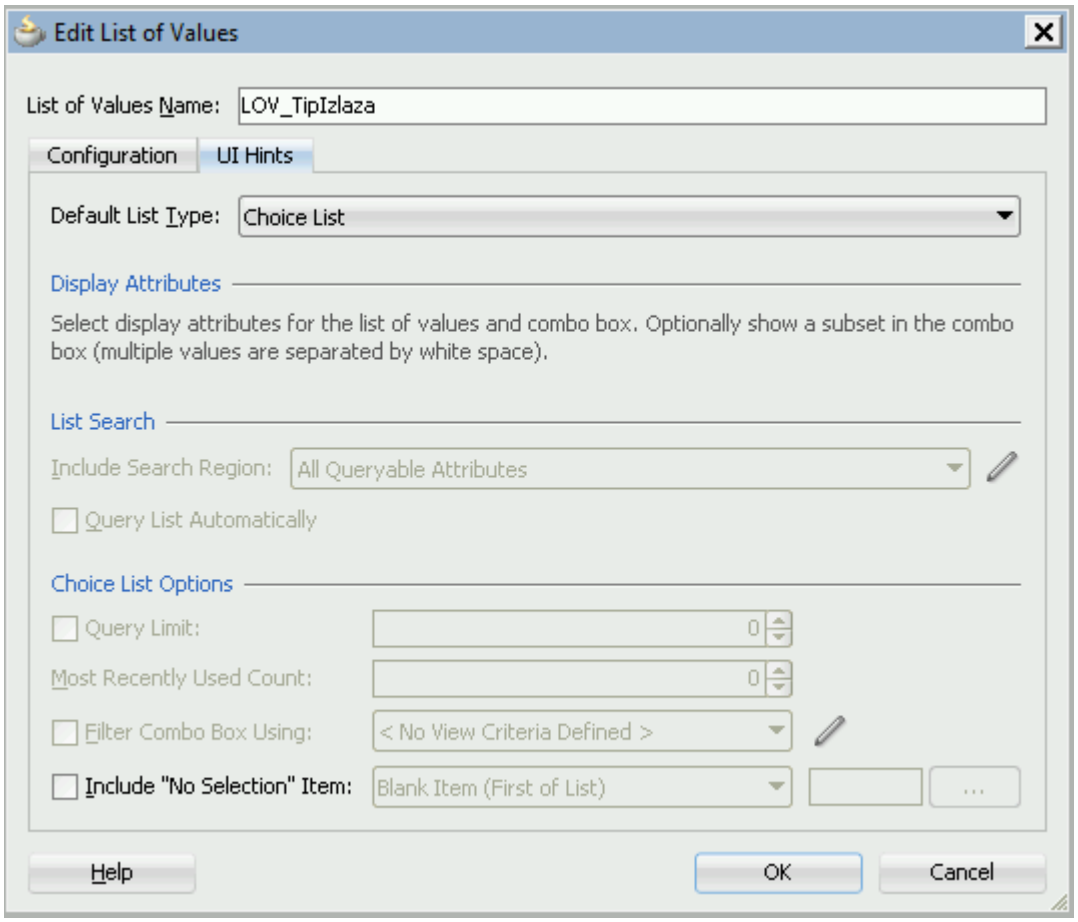
U svojstvima atributa TipIzlaza pogled objekta RrIzlazniceVO otvaramo detalje liste vrijednosti i nakon toga dobivamo ekran sa slike 40. U tom ekranu izabiremo izvor liste vrijednosti odnosno pogled objekt koji smo ranije izradili i koji će biti osnova listi vrijednosti u ovom konkretnom slučaju TipIzlazaLOV. Također u ovom dijelu izabiremo i koje polje iz liste vrijednosti povezujemo sa kojim atributom pogled objekta tj. mapiramo polje iz liste vrijednosti sa atributom pogled objekta. Budući da možemo imati listu vrijednosti koja nam "puni" više atributa pogled objekta u donjem dijelu ekrana na slici 40., mapiramo i preostale atribute. U ovom primjeru nemamo takav slučaj.



Slika 40. Vezivanje liste vrijednosti za pogled objekt

U dijelu ekrana pod nazivom "*UI Hints*" (slika 41.) podešavamo način na koji će se lista vrijednosti prikazivati na korisničkom sučelju. Tu odabiremo tip prikaza liste koji može biti:

- lista izbora
- polje za unos sa formom za izbor vrijednosti
- padajuća lista
- izborni gumb (eng. *radio button)*



Slika 41. Način prikaza liste vrijednosti

Kod dinamičkih listi vrijednosti ovdje ograničavamo broj dohvaćenih slogova u jednom trenutku iz baze. To je dobra značajka prilikom izvršavanja upita koji dohvaćaju veliki broj slogova tako da korisnik ne treba dugo čekati na prikazivanje liste vrijednosti.

4.2 Izrada toka zadataka

U poglavlju koji govori o kontroler sloju vidjeli smo da je taj sloj zadužen za komunikaciju između model i pogled sloja, te za tok izvršavanja pojedinih zadataka između dijelova aplikacije. Upravo tu navigaciju između pojedinih dijelova aplikacije definiramo tokom zadataka bez obzira da li su to ekrani aplikacije, java klase ili drugi objekti. Uobičajeno je da unutar jedne aplikacije postoji više tokova zadataka međusobno povezanih koje se izmjenjuju ovisno o korisničkim akcijama.

U Oracle ADF-u tok zadataka prikazujemo dijagramom. Dijagram se sastoji od aktivnosti i veza između aktivnosti. Aktivnosti mogu biti ADF stranice aplikacije, metode unutar java klase, pozivi Internet adresa, komponente za preusmjeravanje toka aplikacije (eng. *router*), komponente za povratak u prethodni tok zadataka pa čak i sami dijagrami toka zadataka. Tim dijagramom definiramo tok aplikacije odnosno način na koji će se korisnik kretati kroz aplikaciju. Svaki dijagram toka zadataka ima jednu ili više ulaznih točaka (aktivnosti) i nula ili više izlaznih točaka. Između ulaznih i izlaznih točaka postoji više aktivnosti.

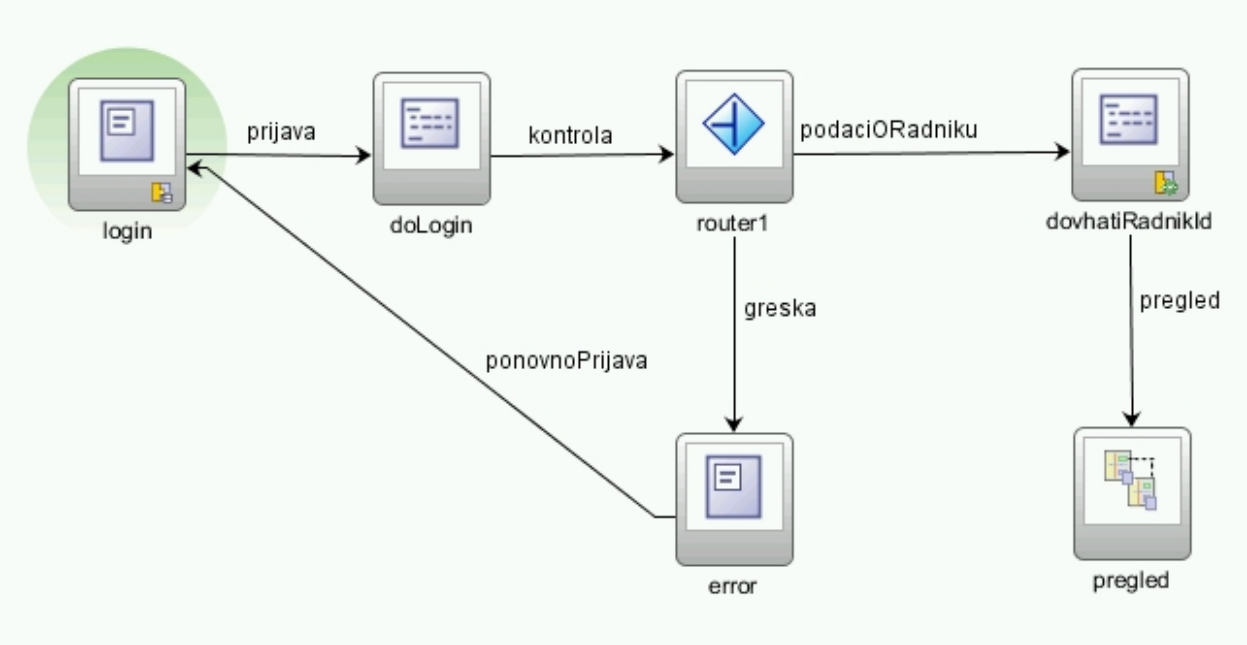
Aplikacija e-izlaznice sastoji se od nekoliko dijagrama toka zadataka koji obuhvaćaju glavne funkcionalnosti sustava:

1. Prijava u sustav e-izlaznice
2. Pregled izlaznica
3. Unos i ažuriranje izlaznica
4. Unos i slanje e-mail-a rukovoditelju odnosno odgovornoj osobi

U slijedećim potpoglavljima biti će obrađeni gore navedeni dijagrami toka zadataka.

4.2.1 Prijava u sustav e-Izlaznice

Prijavu u sustav definirali smo tokom zadataka prikazanim na slici 42.



Slika 42. Dijagram toka zadataka za prijavu u sustav

Prijava u sustav zamišljena je prikazivanjem ADF stranice na kojoj će korisnik unijeti korisničko ime i lozinku. Upravo ta ADF stranica predstavlja ulaznu aktivnost u tok zadataka za prijavu u sustav. Ulazna aktivnost na dijagramima toka zadataka prikazana je obojanom kružnicom oko aktivnosti. U našem slučaju aktivnost "login". Nakon unosa korisničkih podataka sljedeća aktivnost je "doLogin" aktivnost. Ta aktivnost predstavlja metodu Java klase koju pozivamo za provjeru ispravnosti unesenih podataka odnosno usklađenost korisničkog imena i lozinke. Metoda "doLogin" komunicira sa aplikacijskim serverom na kojem su definirani korisnici te vraća podatak tipa *"boolean"* tj. "1" ili "0".

Slijedeća komponenta dijagrama je usmjerivač koja služi za preusmjeravanje toka zadataka u ovisnosti o ulaznom parametru. U ovom slučaju usmjerivač je postavljen na način da ako je

ulazni parametar “0” odnosno da su uneseni podaci za prijavu u sustav krvi, tok se preusmjerava na stranicu sa informacijama o grešci i sa te stranice tokom “ponovoPrijava” dovodi se korisnika ponovo na početnu aktivnost tj. početnu stranicu. Ako je na usmjerivaču ulazni parametar “1” tok zadataka se preusmjeruje na aktivnost “dohvatiRadnikId”. To je također metoda Java klase koja nam služi za dohvaćanje primarnog ključa radnika iz baze podataka, da bi se na stranici za pregled e-izlaznica vidjeli samo podaci koji pripadaju tom radniku. Stranica za pregled izlaznica postavljena je na način da se preko primarnog ključa radnika vide samo izlaznice tog radnika.

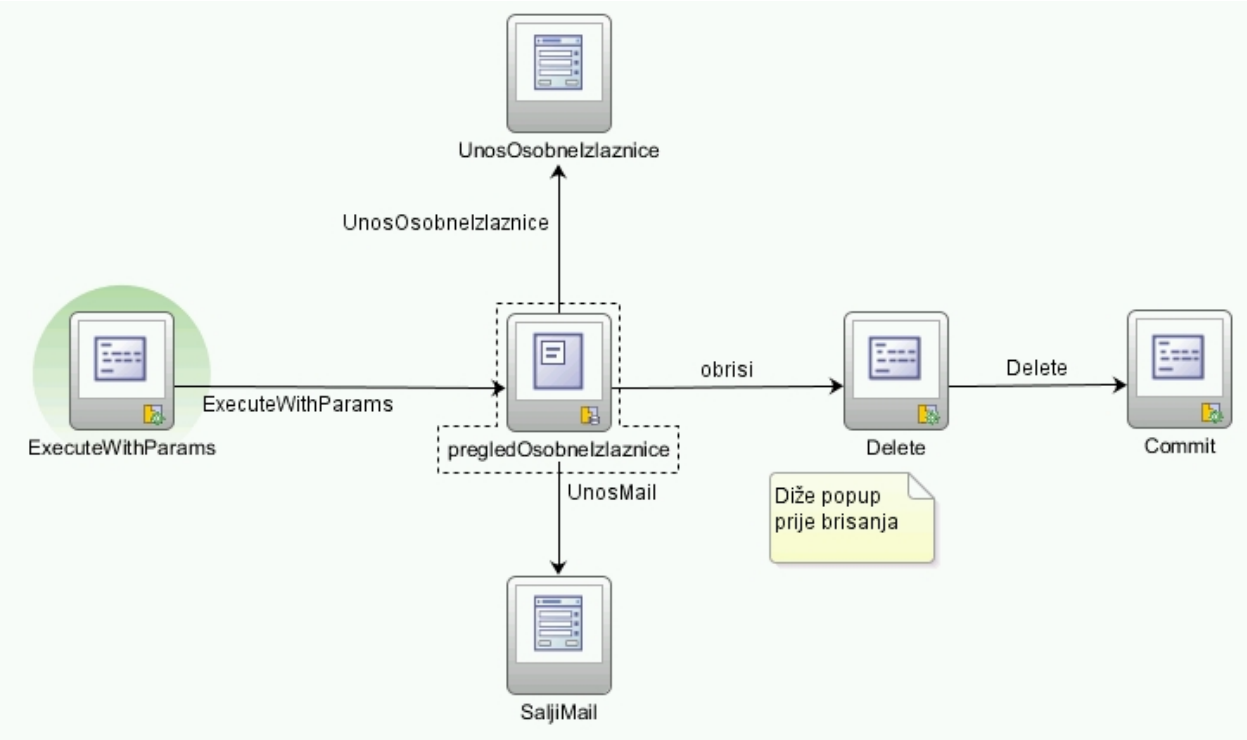
Posljednja aktivnost u dijagramu na slici 42. je aktivnost “pregled”. Ta aktivnost je zapravo slijedeći dijagram toka zadataka, odnosno tok zadataka koji nam služi za pregled i ažuriranje podataka o izlaznicama.

4.2.2 Pregled, unos, ažuriranje i brisanje izlaznica

Nastavak zadnje aktivnosti dijagrama toka zadataka iz prethodnog potpoglavlja dana je na slici 43. Taj dijagram toka zadataka predstavlja pregled, unos, ažuriranje i brisanje e-izlaznica.

Na ovom dijagramu ulazna aktivnost je standardna ADF aktivnost “ExecuteWithParams”. Ovoj aktivnosti proslijeđen je primarni ključ radnika iz prethodnog dijagrama toka tj. iz aktivnosti “dohvatiRadnikId”. Aktivnost “ExecuteWithParams” izvršava upit prema bazi sa parametrom primarnog ključa radnika. Kao rezultat dobiva set slogova izlaznica prijavljenog radnika koje prosljeđuje na sljedeću aktivnost odnosno na stranicu za prikaz podataka - aktivnost ”pregledOsobneIzlaznice”. Kao što smo vidjeli u poglavlju o prototipu izgleda, na toj stranici postoje tipke za unos, ažuriranje, brisanje slogova e-izlaznica te tipke za slanje elektroničke pošte (slika 6.). U ovisnosti o tim tipkama određuje se slijedeća aktivnost na toku zadataka.

Tok ”unosOsobeIzlaznice” vodi na slijedeći dijagram toka zadatka koji ćemo obraditi u slijedećem potpoglavlju. Pod unosom osobne izlaznice podrazumijeva se i unos i ažuriranje podataka.



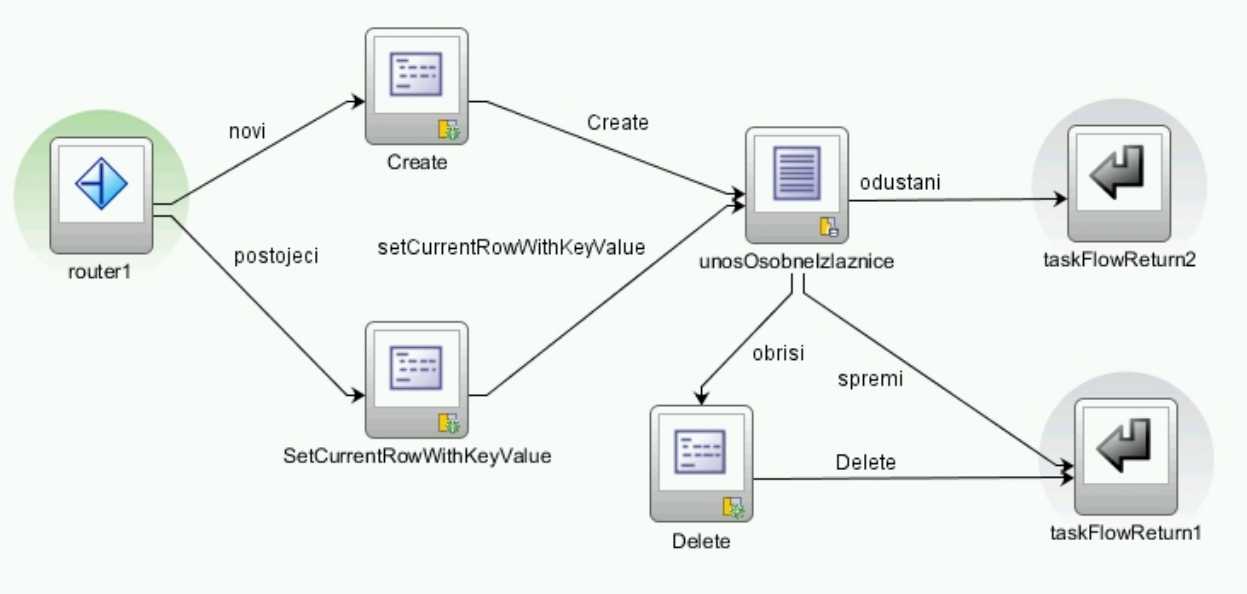
Slika 43. Tok zadataka pregleda, ažuriranja i brisanja e-izlaznice

Tok “UnosMail” vodi nas do aktivnosti koja šalje elektroničku poštu rukovoditelju odnosno traži odobrenje za unesenu izlaznicu. To je također jedan od dijagrama toka zadataka koji ćemo obraditi kasnije.

Na ovom dijagramu ostao nam je još tok 'obriši' koji vodi do aktivnosti “Delete” a koja služi za brisanje izlaznica (ako je brisanje dozvoljeno). Za akciju brisanje nije napravljen novi tok zadataka jer operacija ne zahtjeva od korisnika nikakav unos nego samo potvrđivanje brisanja. Potvrđivanje brisanja javlja se porukom u skočnom prozoru koji se prikazuje na sredini ekrana. Nakon potvrđivanja brisanja briše se izlaznica i spremaju se promjene - aktivnost “Commit”.

4.2.3 Unos i ažuriranje osobne izlaznice

Kao što je vidljivo iz prototipa izgleda pregleda izlaznica (slika 6.) na vrhu postoje tipke za unos i ažuriranje. Za akcije koje se izvršavaju pritiskom na te tipke zadužen je tok zadataka prikazan na slici 44.



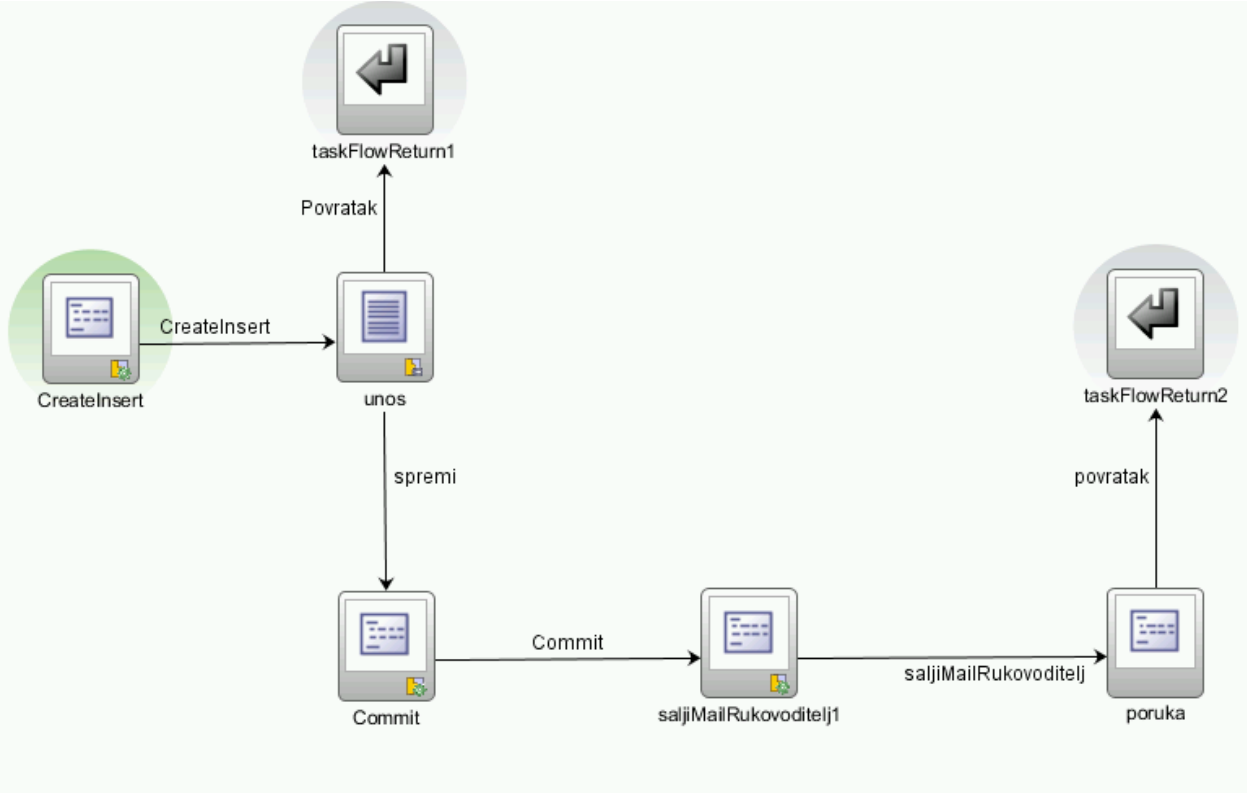
Slika 44. Prikaz dijagrama toka zadataka za unos i ažuriranje e-izlaznica

Ulazna aktivnost toka zadataka u ovom slučaju je usmjerivač koji ima zadatak odrediti da li se u tok zadatka ušlo kroz akciju "unos" ili kroz akciju "ažuriranje". Usmjerivač je programiran tako da ako se došlo kroz akciju "ažuriranje" onda je parametar koji se prenosi u tok popunjen vrijednošću primarnog ključa izlaznice tj. atribut ID tablice RR_IZLAZNICE, a ako se u tok ušlo sa praznim parametrom onda usmjerivač zna da se radi o unosu novog sloga u tablicu RR_IZLAZNICE. Usmjerivač se u ovisnosti o tome preusmjerava na dvije standardne ADF aktivnosti "Create" i "SetCurrentRowWithKeyValue". Ove aktivnosti vode na skočni prozor u kojem se prikazuje forma koja je ili prazna ukoliko se radi o unosu ili popunjena sa podacima tekućeg sloga ukoliko se radi o ažuriranju. Ovaj tok zadataka ima dvije izlazne aktivnosti koje su označene crvenom strelicom i obojanom kružnicom. Aktivnost pod nazivom "taskFlowReturn1" poziva se nakon što izvršimo promjene i želimo spremiti te promjene (tipka

"Spremi" na slici 7.). Ta aktivnost podešena je tako da sprema promjene izvršene prema bazi tj. izvršava SQL naredbu "COMMIT". Međutim, ako promjene ne želimo spremiti onda se izvršava aktivnost "taskFlowReturn2" koja promjene ne potvrđuje prema bazi nego vraća prethodne vrijednosti tj. izvršava SQL naredbu "ROLLBACK". Izlazne aktivnosti rade na način da tok zadataka vraćaju na tok koji ga je pozvao, konkretno u našem slučaju na tok zadataka za pregled.

4.2.4 Unos i slanje elektroničke pošte

Dijagram toka zadataka prikazan na slici 45. ima funkciju slanja elektroničke pošte sa sadržajem e-izlaznice rukovoditelju na odobravanje.



Slika 45. Dijagram toka zadataka za slanje elektroničke pošte

Na ovom dijagramu ulazna aktivnost je aktivnost “CreateInsert” koja je također jedna od standardnih ADF aktivnosti. Ta aktivnost nam služi za dodavanje sloga u tablicu na bazi odnosno za pripremanje pripadajućeg entiteta u način rada za dodavanje. U ovom konkretnom slučaju radi se o tablici RR_IZL_POSLAN_MAIL. Preko toka “CreateInsert” dolazimo do slijedeće aktivnost tj. do unosa vrijednosti atributa kroz formu u skočnom prozoru koji se podiže na sredini ekrana. Na prototipu izgleda tog skočnog prozora (slika 8.) vidimo tipku “Slanje e-mail-a” koja poziva tok “spremi”, te se promjene spremaju na bazi preko aktivnosti “Commit”. Nakon uspješnog spremanja, poziva se metoda klase “saljiMailRukovoditelj1” koja poziva baznu proceduru te šalje obavijest rukovoditelju. Aktivnost “poruka” predviđena je za ispisivanje poruke na korisničkom sučelju ako je iz bilo kojeg razloga došlo do greške prilikom slanja elektroničke pošte. Nakon uspješnog završetka preko aktivnosti “taskFlowReturn2” vraćamo se u prethodni tok zadataka.

Za prekid unosa odnosno za odustajanje od slanja elektroničke pošte ovdje nije predviđena izrada posebne tipke nego se koristi “X” iz gornjeg desnog kuta skočnog prozora.

4.3 Izrada korisničkog sučelja

Kao što smo ranije vidjeli, korisničko sučelje nalazi se unutar pogled sloja, a prema organizacijskoj strukturi ADF projekta, korisničko sučelje nalazi se u projektu “ViewController”. Aplikacija e-izlaznice je aplikacija koja će se izvoditi u Internet pretraživaču te je kao tehnologija izrade korisničkog sučelja izabrana "ADF faces" tehnologija. U JDevoloper-u izrada korisničkog sučelja prilično je intuitivna i svodi se na smještanje komponenti prikaza iz palete komponenti na radnu površinu te vezivanje tih komponenti za pogled objekte model sloja, Java klase, aktivnosti na dijagramima toka zadataka, itd. U ADF-u za izradu korisničkog sučelja na raspolaganju nam je preko 150 gotovih komponenti uz mogućnost izrade i vlastitih komponenti. Pri izradi sučelja uzimamo u obzir prototip sučelja koji više ili manje odstupa od krajnjeg izgleda.

Ovaj sustav obuhvaća nekoliko korisničkih sučelja:

- sučelje za prijavu u sustav
- sučelje za pregled osobnih izlaznica
- sučelje za unos osobne izlaznice
- sučelje za unos i slanje elektroničke pošte
- sučelje za pregled osobne statistike izlaza(grafički prikaz)

Ova sučelja se izmjenjuju u ovisnosti o korisničkim akcijama. Te izmjene odnosno sam slijed operacija definiran je dijagramima toka zadataka. Svaki tok na dijagramu toka zadataka vezan je za jednu korisničku akciju na sučelju.

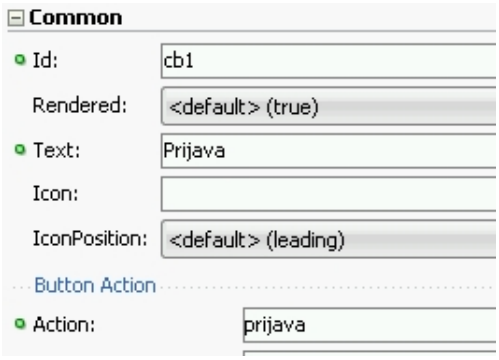
U daljnjem tekstu pobliže će se opisati svaki od gore navedenih sučelja.

4.3.1 Izrada sučelja za prijavu u sustav

Za izradu korisničkog sučelja za prijavu u sustav potrebna su nam polja za unos korisničkog imena te polje za unos lozinke. Također nam je potrebna i tipka za prijavu u sustav. ADF komponenta koja služi kao polje za unos vrijednost naziva se "Input text", a za potrebe tipke za prijavu koristiti ćemo ADF komponentu "Button". Sve te komponente smjestiti ćemo na uzdignuti posjenčani okvir odnosno ADF komponentu "Panel box". Obzirom da je ovo korisničko sučelje malih dimenzija u odnosu na veličinu ekrana smjestiti ćemo ga ma sredinu ekrana. Nakon dovlačenja i smještanja svih komponenti na radnu površinu JDeveloper-a, sučelje izgleda kao što je prikazano na slici 46.

Slika 46. Izgled korisničkog sučelja za prijavu u JDeveloper-u

Ovo što vidimo na slici 46. nije korisničko sučelje koje vidi korisnik, to je korisničko sučelje kako ga vidi programer. Vidimo da komponenta "Panel box" ima mogućnost dodavanje alatne trake (eng. *toolbar*) no tu mogućnost nećemo iskoristiti jer nam ovdje nije potrebno. Ostalo nam je još vezivanje tipke "prijava" na akciju iz dijagrama toka zadataka. Akciju koja se izvodi na tipku postavljena je u paleti postavki te komponente (slika 47.)



Slika 47. Postavke tipke za prijavu

Akcija koja se poziva je tok “prijava” iz dijagrama toka zadataka za prijavu u sustav. Nakon pokretanja aplikacije pojavljuje se forma za prijavu kakva je prikazana na slici 48.

Slika 48. Izgled forme za prijavu u sustav

Nakon što korisnik upiše korisničko ime i lozinku te pritiskom na tipku “Prijava” pokreće se tok prijave u sustav kroz dijagram toka zadataka zaduženog za prijavu u sustav.

4.3.2 Sučelje za pregled osobnih izlaznica

Sučelje za pregled osobnih izlaznica temeljno je sučelje aplikacije e-izlaznice. Sa ovog sučelja kreću sve daljnje akcije unosa, ažuriranja, brisanja, poziv ostalih sučelja, itd. I kod izrade ovog sučelja pokušavamo se uz manje ili veće promjene držati predloženog izgleda (slika 6). Ako pogledamo prototip izgleda vidimo da ovo sučelje možemo podijeliti na četiri dijela:

1. zaglavlje - sadrži logo i naziv aplikacije te akciju za odjavu iz sustava
2. tabelarni pregled osobne izlaznice
3. detalji pregled osobne izlaznice
4. tabelarni pregled poslane elektroničke pošte

Kod izrade zaglavlja koristili smo ADF komponentu “Image” koja služi za prikaz ikona, fotografija i slično. Tu komponentu vezali smo za datoteku koja sadrži logo aplikacije. Datoteka nije smještena u bazi podataka nego na datotečnom sustavu aplikacijskog servera.

Za sam ispis naziva aplikacije koristili smo komponentu “Output text”, dok za izradu poveznice koja poziva akciju odjavljivanja iz sustava koristili smo komponentu pod nazivom “Link”.

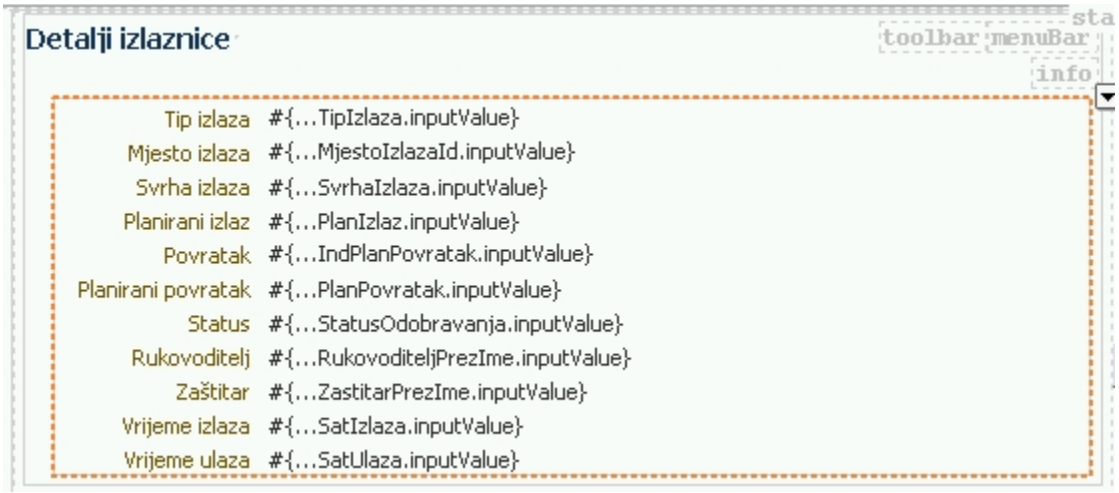
U drugom dijelu aplikacije prikazan je pregled osobnih izlaznica za osobu koja je prijavljena u sustav. To je definirano, kao što smo ranije vidjeli u aktivnostima dijagrama toka zadataka koji služi za pregled osobnih izlaznica. Komponenta ADF-a koja nam omogućuje tabelarni prikaz naziva se “Table”. Bitno je naglasiti da su komponente koje prikazuju podatke iz baze vezane za pogled objekte model sloja. Tabelarni prikaz podataka vezan je za objekt RrIzlazniceVO koji je preko entitet objekta RrIzlazniceEO vezan za tablicu na bazi RR_IZLAZNICE.

Kao što je vidljivo iz prototipa sučelja iznad tablice postavljene su tipke za akcije unosa, ažuriranje, brisanje te slanja elektroničke pošte. U sam izgled tablice ne možemo dodati te tipke, stoga sve skupa moramo smjesti u panel koji nam to omogućava. U standardnim ADF komponentama postoji više panela, ali za ovaj rad odlučili smo se na “Panel collection”. Komponenta “Panel collection” ima u gornjem dijelu mogućnost dodavanja alatne trake koju smo iskoristili upravo za postavljanje tipki. Na slici 49. prikazan je panel sa pripadajućim tipkama i tabelarnim prikazom tabele u fazi razvoja korisničkog sučelja.

Slika 49. Komponenta “Panel collection”

Komponeta za tabelarni prikaz u standarnoj funkcionalnosti ima mogućnost filtriranja i sortiranja prikazanih podataka, te prikaz i razmještaj kolona po želji korisnika.

U trećem dijelu sučelja prikazani su detalji osobne izlaznice označene u tabelarnom prikazu u drugom dijelu ekrana. Ovakav pregled podataka u obliku forme predstavlja jednostavne čitanje podataka od onog u tabelarnom prikazu jer se prikazuju podaci jedne osobne izlaznice. Za prikaz podataka u obliku forme u ADF-u koristi se komponenta "Panel form layout" koje u ovisnosti o postavkama prikazuje podatke jedan ispod drugog u jednoj ili više kolona. Ova forma vezana je za isti pogled objekt model sloja Rr_IzlazniceVO kao i tablica u prethodnom dijelu. Radi se o istim podaci samo o drugačijem prikazu. Radi boljeg dizajna i preglednosti i ovaj dio sučelja staviti ćemo u panel sa naslovom. Komponenta koja nam to omogućava je "Panel header" komponenta. Izgled sučelja u trenutku dizajna prikazan je na slici 50.



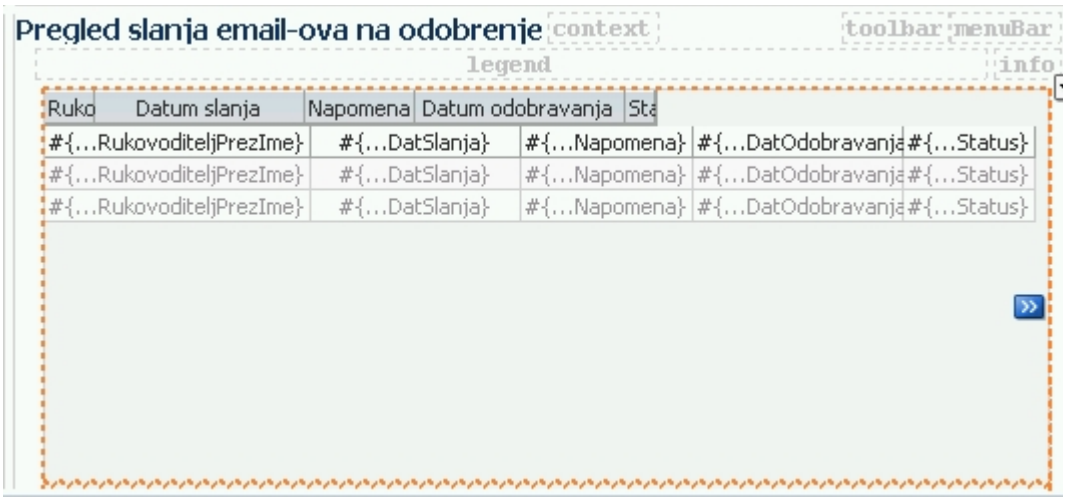
Slika 50. Forma za pregled osobne izlaznice

Na slici vidimo da i u "Panel header" komponentu možemo staviti alatnu traku. Međutim, to nam ovdje nije potrebno jer smo tipke za akciju stavili u prethodni dio sučelja.

Četvrti dio ovog sučelja je dio koji prikazuje podatke o poslanim elektroničkim porukama. Prikazuju se podaci o rukovoditelju, datumu slanja, sadržaju poruke datumu odobravanja i statusu. Podatke prikazujemo u obliku tablice sa istom komponentom koju smo koristili u drugom dijelu sučelja, samo ova komponenta vezana je za drugi pogled objekt model sloja, a to je RrIzlPoslanMailVO koja prikazuje podatke iz tablice RR_IZL_POSLAN_MAIL. Ovdje se

prikazuju poslane elektroničke poruke za odabranu izlaznicu iz tabelarnog prikaza osobnih izlaznica. Kako i u trećem dijelu i ovu komponentu smještamo u panel sa naslovom "Pregled slanja email-ova na odobravanje". Slika 51. prikazuje ovaj dio ekrana u trenutku dizajna.

Nakon izrade, sve dijelove objedinjujemo u jedan zajednički panel. Za tu potrebu izabrana je komponenta "Panel Tabbed" (slici 52) koja na vrhu ima kartice (eng. *tab*) te nam na taj način



Slika 51. Pregled poslanih elektroničkih poruka

omogućava lakše buduće nadograđivanje aplikacije koje predstavlja dodavanje novih kartica.

Slika 52. Kartice komponente "Panel Tabbed"

Konačni izgled sučelja kako ga korisnik vidi dobivamo nakon pokretanja aplikacije (slika 53.).

Slika 53. Izgled glavnog korisničkog sučelja

4.3.3 Sučelje za unos i ažuriranje osobnih izlaznica

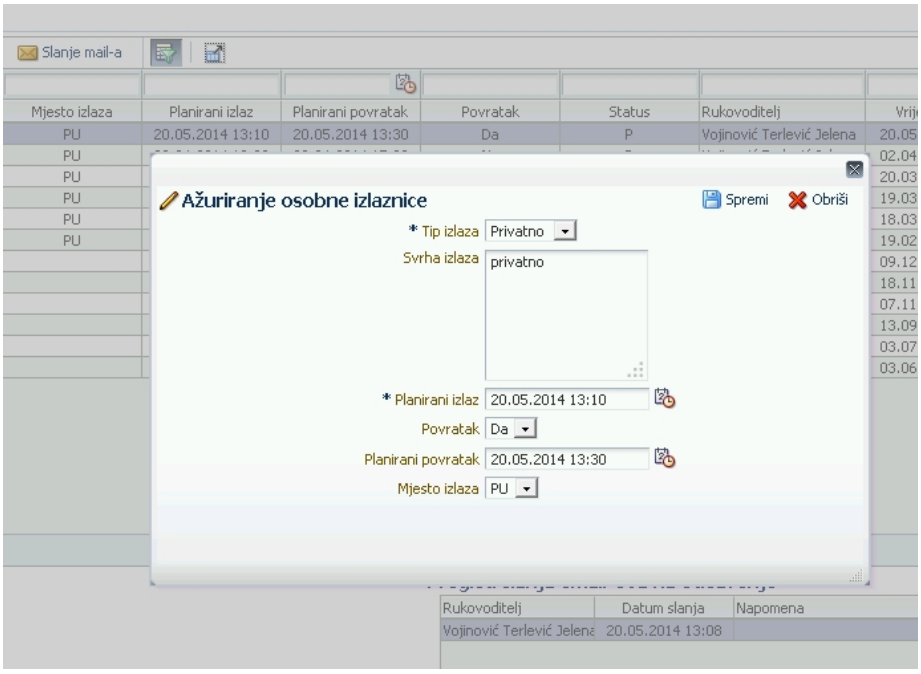
Sučelje za unos i ažuriranje osobnih izlaznica je forma u kojoj se ispunjava zahtjev za izlaz. Ista ta forma koristi se i za ažuriranje osobne izlaznice na način opisan u dijagramu toka zadataka za unos osobne izlaznice. Na sučelju za pregled postoje tipke "Unos" i "Ažuriranje" koji vode na ovo sučelje. Sučelje je zamišljeno na način da se podiže u skočnom prozoru iznad prozora sa pregledom osobnih izlaznica. Ovakvo otvaranje sučelja ne zahtjeva od programera nikakvo kodiranje, već se radi se o postavljanju određenih svojstva na aktivnost u dijagramu toka zadataka. Ta svojstva su "Run As Dialog" koju postavljamo na vrijednost "true" i svojstvo "Display Type" koje postavljamo na "inline - popup" (slika 54). Na ovaj jednostavan način dobili smo prikazivanje sučelja u skočnom prozoru.

Slika 54. Podešavanje prikazivanja unosa u skočnom prozoru

Za potrebe ove forme također kao i kod pregleda detalja osobnih izlaznica koristimo ADF komponentu "Panel form layout". Za razliku od forme koja je korištena u detaljima osobne izlaznice ova forma omogućuje unos odnosno promjenu podataka. Za jednostavnije korištenje sučelja za neka polje koristiti ćemo dinamičku listu vrijednosti. To je lista vrijednosti koja se može puniti dinamički iz pogled objekta. Dinamička lista vrijednosti iskorištena je za polja tip izlaza, povratak i mjesto izlaza. Polje tip izlaza može poprimiti vrijednosti P ili S tj. privatno ili službeno. Polje povratak također može poprimiti dvije vrijednosti Da (D) ili Ne (N), dok mjesto izlaza može biti PU (Pula) ili RI (Rijeka). ADF komponenta koja nam omogućuje prikazivanje takve liste vrijednosti je komponenta "Choice".

Za jednostavniji unos datumskih polja ADF automatski kod takvih polja postavlja ikonu koja poziva kalendar sa mogućnošću izbora datuma (komponenta "Input date").

Slika 55. Dizajn forme za unos osobnih izlaznica.

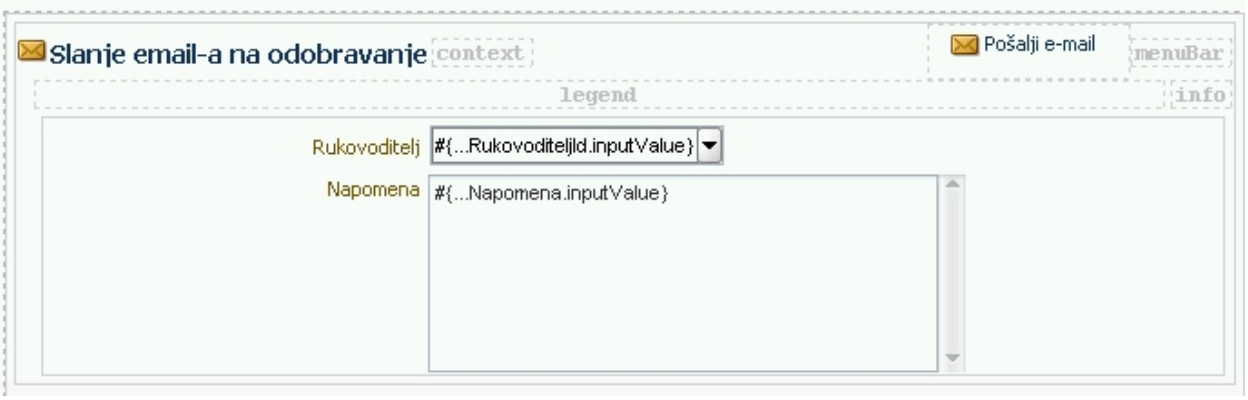


Slika 56. Forma za ažuriranje osobne izlaznice

U gornjem desnog dijelu skočnog prozora postavljena je alatna traka sa tipkama “Spremi” i “Obriši” koje su vezane za tokove na dijagramu toka za unos osobne izlaznice. Izgled forme u trenutku dizajna prikazan je na slici 55., a izgled u aplikaciji prilikom pokretanja prikazan je na slici 56.

4.3.4 Sučelje za unos i slanje elektroničke pošte

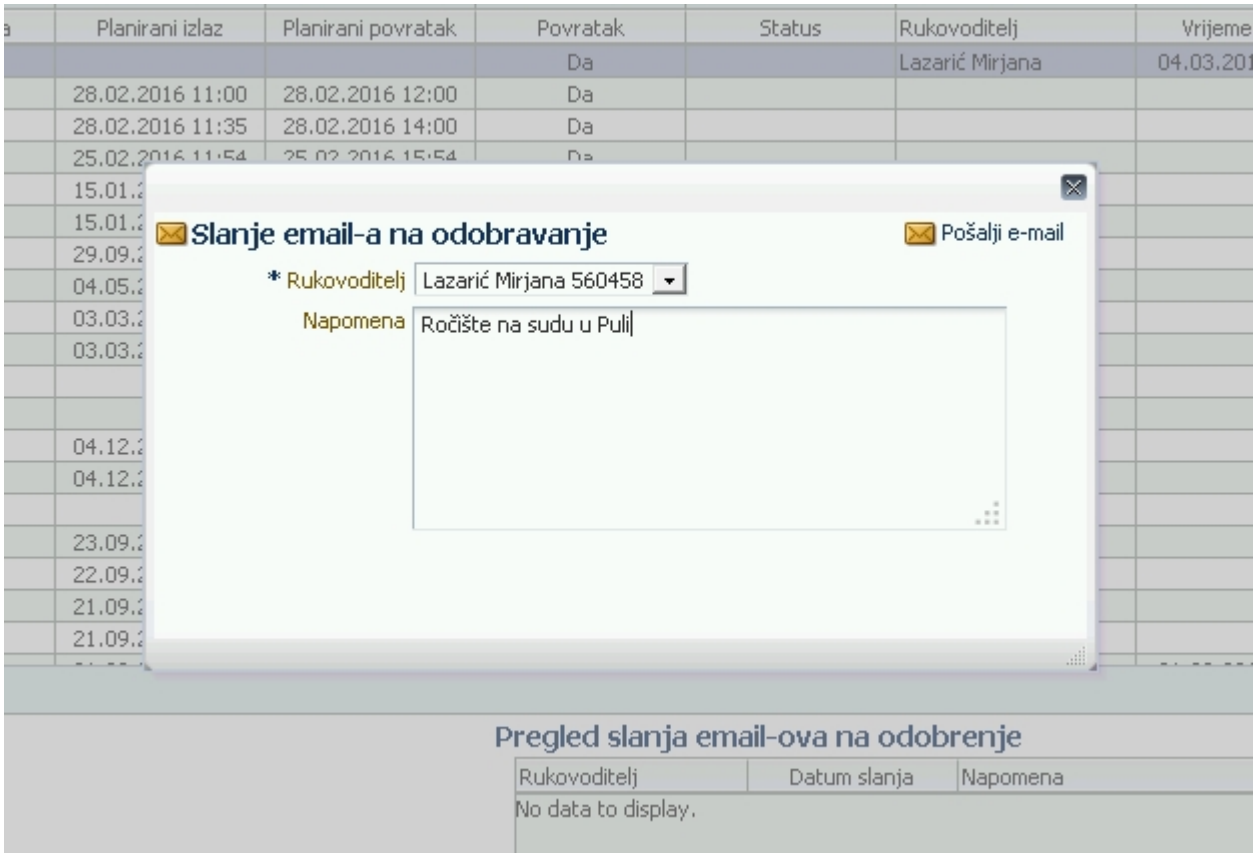
U korisničkom zahtjevu traži se, da se svaka osobna izlaznica odobri od strane rukovoditelja elektroničkom poštom na osobnom računalu ili mobilnom telefonu. Za tu potrebu napravljeno je ovo korisničko sučelje koje omogućava korisniku da nakon što unese osobnu izlaznicu pošalje elektroničku poštu rukovoditelju ili odgovornoj osobi. Sučelje za unos elektroničke pošte poziva se tipkom "Slanje mail-a" na osnovnom prozoru aplikacije. Za slanje elektroničke pošte potrebna su nam minimalno tri podatka: adresa pošiljatelja, adresa primatelja i sadržaj poruke. Adresu pošiljatelja nećemo na sučelju niti prikazivati zato što se radi o korisnikovoj adresi elektroničke pošte, dok će se adresa primatelja izabirati iz liste vrijednosti. Lista vrijednosti popunjena je pogled objektom model sloja RukovoditeljLOV. Upit koji se izvršava u pozadini tog pogled objekta napravljen je tako da se na prvom mjestu prikaže rukovoditelj prijavljene osobe, a nakon toga hijerarhijski ostale odgovorne osobe za prijavljenog zaposlenika.



Slika 57. Sučelje za unos i slanje elektroničke pošte u fazi dizajna

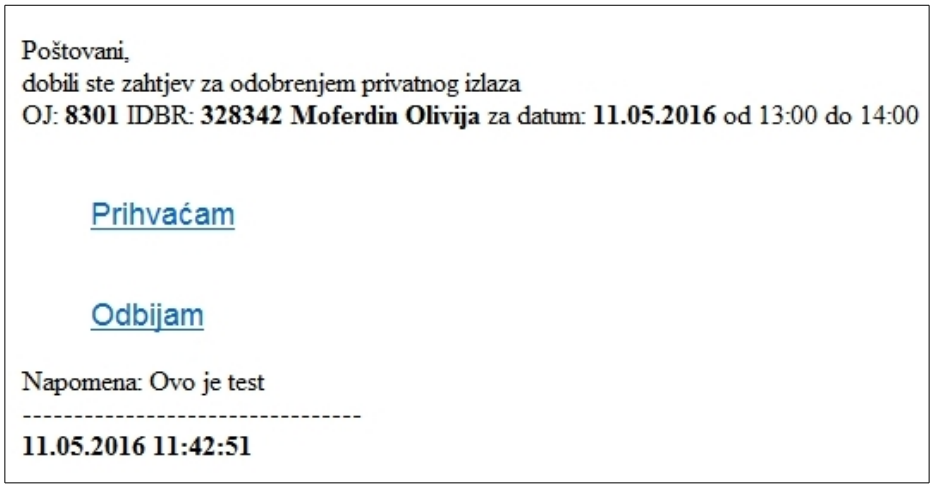
Komponente korištene kod ovog sučelja već smo ranije upoznali, a to su padajuća lista vrijednosti i polje za unos teksta. Obzirom da se u polju za unos teksta radi o unosu sadržaja elektroničke pošte, polje smo povećali i omogućili unos više linija (eng. *multiline*). Sve skupa je smješteno u ADF komponentu "Panel form layout" te je na vrhu dodana alatna traka sa akcijom "Pošalji e-mail". Kao što je vidljivo na dijagramu toka zadataka (slika 45.) ova akcija izvršava spremanje sloga na bazu (tok "Commit") i pozivanje procedure za slanje elektroničke pošte (tok

"saljiMailRukovoditelj"). I ovo sučelje podiže se u skočnom prozoru iznad osnovnog prozora. Izgled sučelja u fazi dizajna prikazan je na slici 57., dok je na slici 58. prikazano sučelje kakvo korisnik vidi kroz aplikaciju.

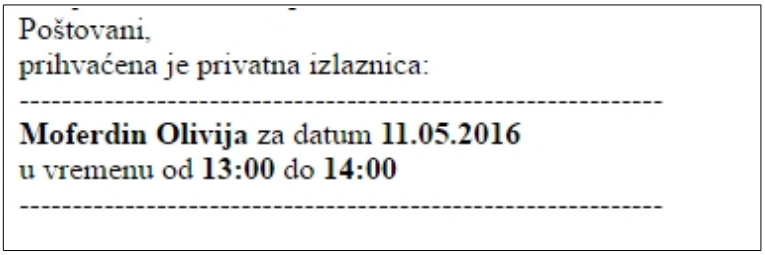


Slika 58. Sučelje za unos i slanje elektroničke pošte

Procedura koja se nalazi u pozadini aktivnosti "saljiMailRukovoditelj1" (slika 45) kreira tijelo elektroničke pošte sa sadržajem prikazanim na slici 59. Na slici vidimo da je napomena koju je unio korisnik pridružena tijelu elektroničke pošte kreirane u proceduri. Također su vidljive i dvije poveznice "Prihvati" i "Odbiji". Pritiskom na njih odgovorna osoba odobrava ili odbija izlaz.



Slika 59. Generirana poruka elektroničke pošte



Slika 60. Povratna poruke elektronske pošte ukoliko je izlaznica prihvaćena

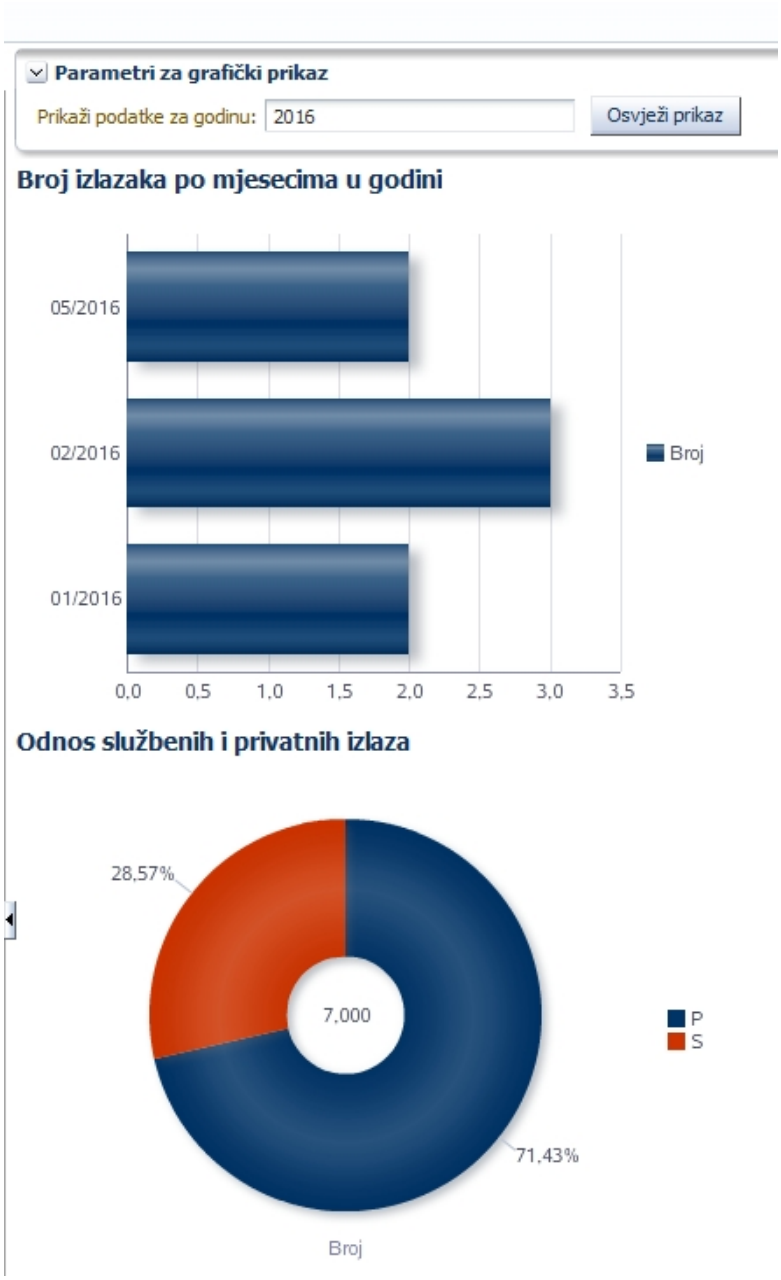
Tim pritiskom generira se odgovarajuća poruke elektroničke pošte koja se šalje korisniku na njegov pretinac elektroničke pošte (slika 60) i ažurira se status pripadajućeg sloga u tablici RR_IZL_POSLAN_MAIL, te se osvježava prikaz na sučelju pregleda osobnih izlaznica.

4.4 Grafički pregled

Uz osnovno korisničko sučelje aplikacije odnosno uz karticu osobne izlaznice dodali smo još jednu karticu pod nazivom grafički pregled. Kartica grafički pregled podijeljena je po vertikali na dva dijela. Za dijeljene kartice iskoristili smo mogućnosti ADF komponente "Splitter" koja dijeli panel na kojem se nalazi po vertikalnoj ili horizontalnoj osi u omjeru definiranom u svojstvima komponente. Na lijevoj strani ekrana pozicioniran je kalendar koji prikazuje pregled osobnih izlaznica iz prve kartice, ali u dnevnom, tjednom i mjesečnom obliku u ovisnosti o pritisnutoj tipki na vrhu kalendarske komponente (slika 61.).

Slika 61. Kalendarski prikaz

Kalendar je također baziran na istom pogled objektu kao i tabelarni prikaz iz prve kartice - RrIzlazniceVO. Komponenta za kalendarski prikaz je standardna ADF komponenta "Calendar" koja za standardnu funkcionalnost ne iziskuje nikakvo dodatno programiranje nego samo postavlja svojstva kao što su izvor podataka, period za koji se žele prikazati podaci, i odabir podatka koji se želi prikazati u samom kalendaru.



Slika 62. Prikaz u obliku grafova

Na desnoj strani kartice “grafički prikaz” postavljeni su grafovi kojima se prikazuju statistički podaci prijavljenog zaposlenika. Pokazuju se podaci o broju izlaza po mjesecu za godinu navedenu u polju “Prikazi podatke za godinu” na vrhu desne strane ekrana (slika 62.). Također se pokazuju i podaci o odnosu službenog i privatnog izlaza radnika na razini godine navedene na vrhu ekrana. Polje “Prikazi podatke za godinu” inicijalno je popunjeno tekućom godinom. Za potrebe svakog od grafova napravljeni su pripadajući pogled objekti. Pogled objekti bazirani su na jednostavnom upitu koji broji izlaze po mjesecima za godinu (slika 63.), odnosno za potrebe drugog grafa broji tipove (S,P) izlaza na razini godine (slika 64.).

Query

Data for this view object will be retrieved from the datasource using the following SQL query.

```
SELECT to_char(t.plan_izlaz,
               'yyyymm') sort,
       to_char(t.plan_izlaz,
               'mm/yyyy') mj_god,
       COUNT(*) broj
  FROM rr_izlaznice t
 WHERE t.radnik_id = :pRadnikId
   AND to_char(t.plan_izlaz,
               'yyyy') = :pGodina
 GROUP BY to_char(t.plan_izlaz,
                  'yyyymm'),
          to_char(t.plan_izlaz,
                  'mm/yyyy')
order by to_char(t.plan_izlaz,'yyyymm') desc
```

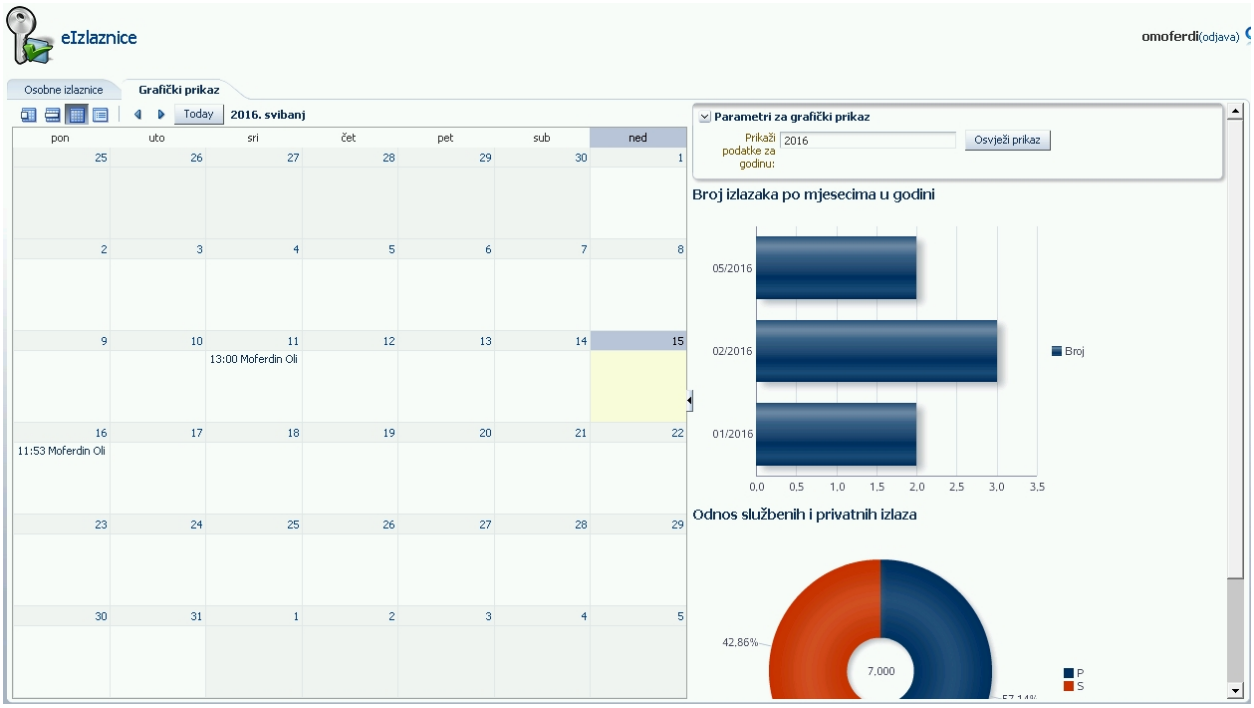
Slika 63.Upit za prikaz broja izlaza po mjesecima u godini za zaposlenika

Query

Data for this view object will be retrieved from the datasource using the following SQL query.

```
SELECT t.tip_izlaza,
       COUNT(*) broj
  FROM rr_izlaznice t
 WHERE t.radnik_id = :pRadnikId
   AND to_char(t.plan_izlaz,
               'yyyy') = :pGodina
 GROUP BY t.tip_izlaza
```

Slika 64. Upit za prikaz broja tipova izlaza u godini



Slika 65. Kartica grafički prikaz

Komponenta za prikaz grafova također je standardna ADF komponenta “Graph”. Kod te komponente isto tako nije potrebno programiranje već je potrebno postaviti svojstva kao što su tip grafa, izvor podataka te izabrati koji će se podaci prikazivati na X i Y osi. Izgled kartice u pokrenutoj aplikaciji prikazan je na slici 65.

5 ZAKLJUČAK

U ovom je radu prikazan razvoj sustava e-izlaznice u Oracle ADF tehnologiji koristeći Oracle JDeveloper razvojni alat. Kako smo vidjeli kroz JDeveloper podržani su svi nivoi razvoja aplikacije od analize poslovnog procesa, modeliranja baze podataka pa sve do izrade aplikacije. Zbog mogućnosti prenošenja objekata i svojstva objekta između različitih faza razvoja aplikacije ovaj nam alat u pojedinim fazama pojednostavljuje proces razvoja aplikacije.

JDeveloper razvojni alat u potpunosti je integriran sa ADF tehnologijom, te nam omogućuje pristup svim funkcijama i objektima ADF-a na vizualan i deklarativan način. Ukoliko je potrebno moguće je i svaku funkcionalnost prilagoditi specifičnim zahtjevima.

ADF tehnologija bazirana je na slojevitoj MVC arhitekturi što nam omogućuje razvoj i održavanje svakog sloja odvojeno i neovisno jedan od drugome. Iz tog razloga ovakav način arhitekture pogoduje i timskom radu.

Posebno treba naglasiti uporabu dijagrama toka zadataka koji nam na vizualan način prikazuju slijed operacija unutar aplikacije odnosno sam tok izvođenja aplikacije, a predstavljaju i vezu model i pogled sloja. Dijagramima toka zadataka riješena je prijava korisnika u aplikaciju, unos i pregled osobnih izlaznica te unos i slanje elektroničke pošte.

Za izradu korisničkog sučelja koji je osnova interakcije sa korisnikom na raspolaganju su brojne, gotove vizualne komponente od osnovnih polja za unos podataka, komponenti za odabir do složenijih kalendarskih prikaza i grafova. Te komponente se relativno lako ugrađuju u aplikaciju i vežu za razne funkcionalnosti i aktivnosti na dijagramima toka zadataka. Zbog velikog broja raspoloživih komponenti i funkcionalnosti, moguće je pokriti širok spektar zahtjeva korisnika u odnosu na korisničko sučelje i načine interakcije i upravljanja aplikacijom.

6 LITERATURA

a) KNJIGE

1) Vinod Krishnan, (2013): Oracle ADF 11gR2 Development Beginner's Guide, Packt Publishing Ltd., UK
2) Duncan Mills, Peter Koletzke, Avrom Roy-Faderman, (2010):Oracle JDeveloper 11g Handbook, The McGraw-Hill Companies, Inc. , USA
3) Ralph Gordon, Peter Jew, Dave Mathews, Landon Ott, and Robin Whitmo, (2008) : Oracle Fusion Middleware: Fusion Developer's Guide for Oracle Application Development Framework, Oracle Inc., USA
4) Frank Nimphius, Lynn Munsinger: Building Rich Internet Applications with Oracle ADF Business Components and Oracle ADF Faces, The McGraw-Hill Companies, Inc., USA
5) Alan Dennis, Barbara Haley Wixom, Roberta M. Roth, (2012): System analysis and design, John Wiley & Sons, Inc., USA

b) INTERNET IZVORI

1) http://www.uacg.bg/filebank/acadstaff/userfiles/publ_bg_397_SDP_activities_and_steps.pdf , posjećeno siječanj 2015
2) http://www.se.rit.edu/~se361/Slides/se361_Chapter_06.pdf , posjećeno siječanj 2015
3) http://www.ics.uci.edu/~kobsa/courses/ICS205/06F/RequirementsGathering.ppt, posjećeno siječanj 2015
4) http://autopoiesis.foi.hr/wiki.php?name=KM%20-%20Tim%2027&parent=37530&page=Dijagram%20slu%C4%8Dajeva%20kori%C5%A1tenja , posjećeno siječanj 2015
5) http://www.veleri.hr/files/datoteke/nastavni_materijali/k_informatika_s1/uml-rooa.pdf , posjećeno siječanj 2015
6) http://e-student.fpz.hr/Predmeti/B/Baze_podataka/Materijali/Auditorne_vjezbe_2.pdf, posjećeno siječanj 2015

7) http://www.pfri.uniri.hr/~tudor/materijali/Informacijski%20sustavi,%20baze%20podataka.htm , posjećeno siječanj 2015
8) http://www.oracle.com/technetwork/developer-tools/adf/adf-11-overview-1-129504.pdf , posjećeno lipanj 2015
9) http://projectmanagementdud.blogspot.com/2013/03/model-view-controller-mvc-simply.html , posjećeno lipanj 2015
10) https://doc.odoo.com/6.0/developer/1_3_oo_architecture/mvc/ , posjećeno lipanj 2015
11) https://docs.oracle.com/middleware/1212/adf/ADFCG/model.htm#ADFCG129, posjećeno studeni 2015
12) https://docs.oracle.com/middleware/1212/adf/ADFCG/controller.htm#ADFCG217, posjećeno studeni 2015
13) http://www.oracle.com/technetwork/developer-tools/jdev/adf-task-flow-design-132904.pdf , posjećeno studeni 2015
14) http://download.oracle.com/otndocs/tech/ias/portal/files/RG/WhitePaper/jdev10g_overview.pdf , posjećeno studeni 2015
15) https://docs.oracle.com/cd/B14099_19/web.1012/b14022/oracle/jbo/ViewObject.html , posjećeno siječanj 2016
16) http://www.oracle.com/technetwork/developer-tools/jheadstart/overview/businessrulesinadfbctechnicalwp-128066.pdf, posjećeno siječanj 2016
17) http://www.kingtraining.com/confdownloads/downloads/King_Oracle_ADF_TaskFlowBeyond10MinuteDemo3.pdf , posjećeno ožujak 2016
18) http://www.kingtraining.com/confdownloads/downloads/King_Oracle_ADF_TaskFlowBeyond10MinuteDemo3.pdf , posjećeno ožujak 2016