

Manager Karti za letove

Manager karti za letove je Java Swing aplikacija, koja za cilj ima pružiti korisnicima mogućnost kupovine karti, pregled i upravljanje istih. Aplikacija omogućava administratorima istu funkciju, ali im je omogućen i unos zrakoplova, kao i unos letova, čije su karte dostupne za kupovinu. Sve je to napravljeno na način koji bi trebao biti jednostavan i intuitivan za korisnika i admina.

Gotova aplikacija omogućuje krajnjem korisniku rad za grafičkim sučeljem, a preko njega rad s bazom podataka iz koje se podatci čitaju, ažuriraju i stvaraju.

U okviru projekta sam ispunio osnovne smjernice koje su se tražile od nas, poput upravljanja nezavisnostima koristeći Maven, osiguravajući rad s podatkovnom bazom, spremanje datoteka koristeći MVC predložak.

Od ostalih predložaka korišteni su:

Command predložak

Command predložak sam koristio u korisničkom okviru (slika okvira je prikazana dalje u dokumentaciji) kako bih omogućio korisniku (kupcu karte) filtriranje dostupnih letova prema određenim uvjetima. Filteri su:

- Najstarije prvo – prikazuje letove s najstarijim datumom leta prema najnovijim (letovi u 2023. prije letova u 2024. godini)
- Najnovije prvo – prikazuje letove s najnovijim datumom leta prema najstarijim (letovi u 2025. prije letova u 2023. godini)

- Postoji li prva klasa – prikazuje samo letove koji imaju prvu klasu sjedala*
- Postoji li poslovna klasa – prikazuje samo letove koji imaju poslovnu klasu sjedala*
- Postoji li ekonomična klasa – prikazuje samo letove koji imaju ekonomičnu klasu sjedala*

U klasičnoj izvedbi Command predložka omogućen je *redo* i *undo* filtera odabirom tih opcija unutar JMenuBar elementa okvira, ili koristeći odgovarajući prećac.

(*Ukoliko je jedan od ovih filtera uključen, npr. prva klasa, biti će prikazani svi letovi koji imaju dostupnu prvu klasu, čak i ako imaju i poslovnu i/ili ekonomičnu dostupnu u isto vrijeme)

Observer predložak

Observer predložak je također korišten u korisničkom okviru. On je zadužen za prikaz kratkog teksta koji prikazuje podatke o odabranom letu u zato predviđenom JTextArea elementu. Pošto sam omogućio JTable element MouseEventListener, da korisnik može odabrati let klikom po tablici, on je Observable objekt. Dok je JTextArea Observer i mijenja svoje stanje ovisno o odabiru korisnika. (Slika priložena u nastavku)

Strategy predložak

Na predavanju smo ovaj predložak koristili kada korisnik odluči spremiti (ili učitati) podatke koristeći JFileExplorer gdje bi Strategy predložak dobio

informaciju želi li korisnik upravljati datotekom u obliku .bin ili .txt i tom informacijom odabrao koju klasu (strategiju) će koristiti. Međutim, ja u svojoj aplikaciji nisam vidio mogućnost ovakvog pristupa, pa sam odlučio koristiti Strategy predložak koji za informaciju dobiva ActionCommand pritisnutog JButton elementa i time bira koju strategiju će pozvati. U ovom slučaju Strategy predložak je zadužen za pozivanje odgovarajućeg načina za interakciju s bazom podataka.

Singleton predložak

Singleton predložak je jednostavan predložak dizajna u Java programerskom jeziku. Omogućuje instanciranje objekta jednom, tako da možemo pozivati samo refrencu na taj objekt gdje god nam treba, umjesto da instanciramo novi objekt kad god ga trebamo koristiti. Potrebno je napraviti konstruktor objekta s privatnim modifikatorom pristupa i onda ga statički instancirati. Ovaj predložak sam koristio za Kontroler klasu, da ga mogu instancirati samo jednom jer je potreban unutar gotovo svakog okvira, i AuxCLS klasu koja je zadužena za spremanje podataka koji su korišteni dalje u kodu.

SOLID princip

Single responsibility – Dio klase se pridržava ovog principa iako se na nekoliko mjesta primjećuje da klase obavljaju više od jednog zadatka (primarno u JPanel klasama)

Open/Closed princip - U izgradni aplikacije se nisam obazirao na ovaj dio SOLID principa. Open/closed princip je odličan za izbjegavanje bugova za

aplikacije u produkciji, koji će dobivati podršku nakon izlaska da se smanji broj mogućih *bugova*.

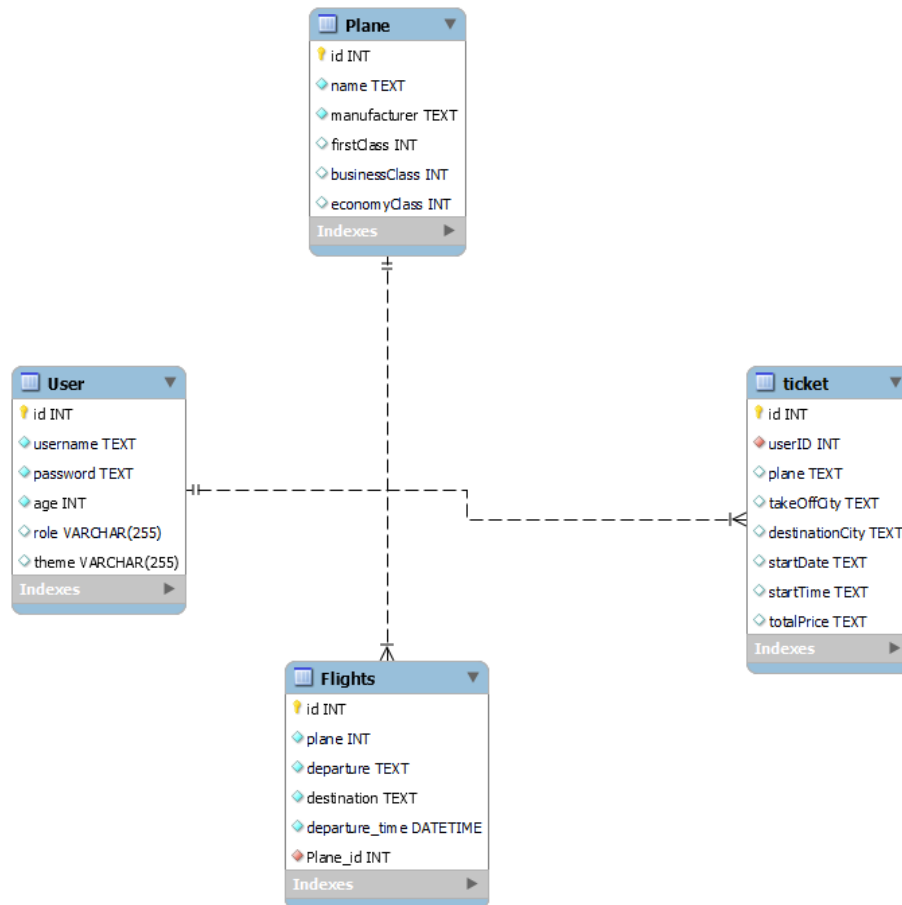
Liskov Substitution – U izgradnji aplikacije ne kršim Liskovu substituciju, ali to je zato što moja aplikacija nije zahtjevala korištenje sučelja (*Interface*) na način gdje bih morao paziti na kršenje Liskove substitucije.

Interface segregation – Princip koji nam objašnjava da je korištenje sučelja koje ima više opisa metoda treba biti smanjeno na više sučelja koje imaju manji broj opisa metoda, tako da klasa koja implementira sučelje koristi sve metode koje je nasljedila. U svojoj aplikaciji sam pazio na ovaj princip na nekoliko mjesta (svi listeneri i command pattern)

Dependency inversion – Koristim ga svugdje po aplikaciji, kršenje ovoga principa je vrlo vidljivo i značajno otežava rad s aplikacijom i testiranjem iste. Praćenjem principa u konstruktor ubacujemo željeni parametar koji očekuje objekt umjesto da instanciramo potpuno novi unutar klase konstruktora.

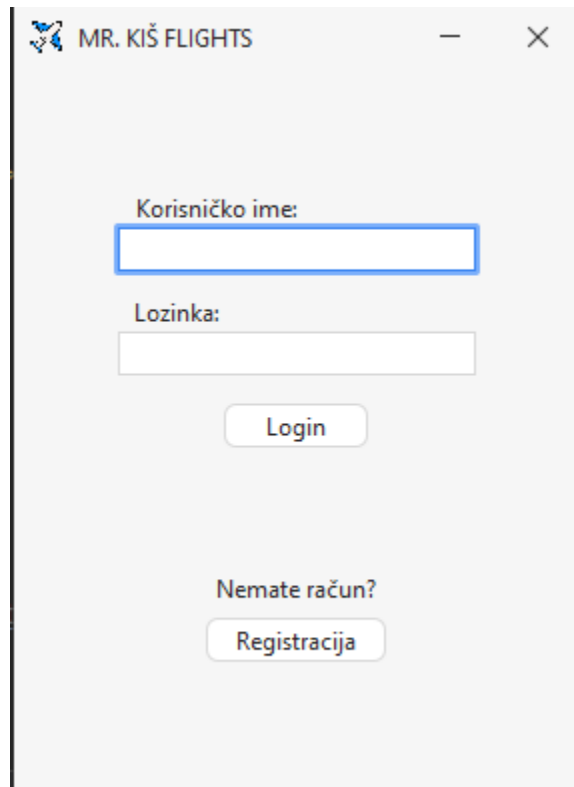
Baza podataka

Za izradu aplikacije rad s bazom podataka bio je obavezan. Za izbor smo imali mogućnost rada sa JDBC ili Hibernate. Ja sam odabrao JDBC (Java DataBase Connectivity). Način na koji sam upravljao bazom podataka je MySql Workbench. Bazu sam napravio u MySql (relacijska baza). Moja baza podataka je spremljena na virtualnoj mašini koja se nalazi na udaljenom serveru u Frankfurtu, kojem sam pristup dobio koristeći uslugu DigitalOcean. Baza podataka je zadužena za spremanje podataka o korisnicima, letovima, zrakoplovima i kupljenim kartama. Relacije svih tablica su objašnjene ispod slike dijagrama baze.



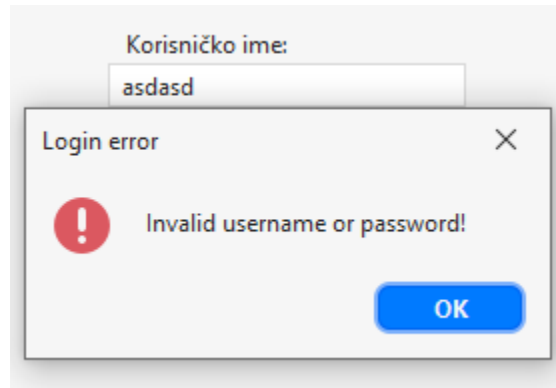
Na dijagramu se vidi jednostavnost baze. Samo 4 tablice koje su povezane u relaciji 1:N (one to many). Npr. Jedan zrakoplov može biti zadužen za obavljanje više letova i jedan korisnik može kupiti više karata, dok suprotno ne vrijedi. Također unutar tablice 'Flights' atribut 'Plane_id' je sekundarni ključ (*Foreign key*) što je način na koji sam povezivao zrakoplov sa letom. Isto vrijedi i za tablicu 'ticket', 'userID' je sekundarni ključ koji refrencira korisnika u tablici 'User'

Snimke zaslona završene aplikacije:




LoginFrame

Okvir u kojem se korisnici prijavljuju s postojećim podacima (korisničko ime i lozinka) ili u slučaju nepostojanja računa, mogućnost kreacije novog računa (opisano naknadno). Netočan unos podatak rezultira greškom koja obavještava korisnika. Uspješna provjera podataka korisnika void na UserFrame. Na istom okviru se prijavljuju i administratori Kojima se na uspješnoj provjeri otvara CreationFrame.




Slika 1: Neuspješna prijava

 IZBORNIK

Mr. KIŠ FLIGHTS

Plane	Departure	Destination	Departure date	Departure time
Boeing 737	Zadar	Zagreb	2024-08-26	11:35:00
F-22	Split	Berlin	2024-09-08	22:15:00
Boeing 737	Zagreb	Cairo	2023-12-12	09:00:00
Cessna SkyHawk	Zadar	Split	2024-06-21	17:55:00
F-15	London	Manchester	2024-09-14	14:30:00
F-35	Munster	Kyev	2024-09-16	13:40:00
Airbus A320	Zadar	Berlin	2024-09-21	13:55:00
Cumulus	London	Paris	2024-09-21	21:00:00
T-33	Zadar	Berlin	2024-09-21	15:07:05
Boeing 737	Zadar	Jeruzalem	2024-09-21	15:16:13

Korisnička forma

Odaberite...

☐ Hand Luggage

☐ Checked Luggage

☐ First Class

☐ Business Class

☐ Economy Class

Ukupna cijena:

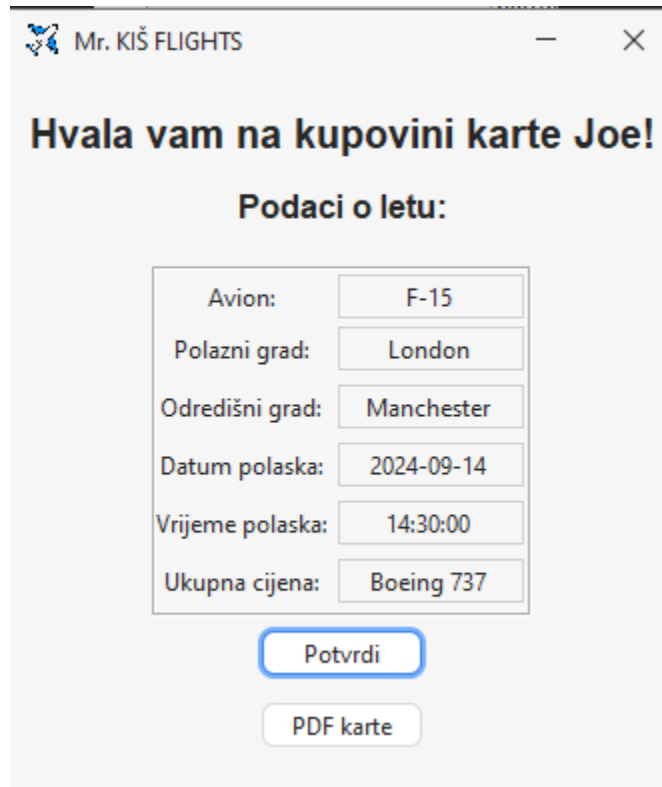
0 €

Book

Korisnička forma

UserFrame je najkompleksniji okvir u ovoj aplikaciji. Posjeduje najviše elemenata, Command i Observer predlošci se nalaze u ovom okviru. Na primjeru slike se vidi da su izbori prijavljene prtljage i klase sjedala zaključane; korisnik ih ne može odabrati. Svi letovi se nalaze u JTable element koji ima MouseEventListener i omogućuje korisniku pritiskanje elemenata po tablici. Klikom na željeni let se otvaraju izbori za prtljagu i ovisno o dostupnosti sjedala – dostupnosti klasa.

ThankYouFrame



Mr. KIŠ FLIGHTS

Hvala vam na kupovini karte Joe!

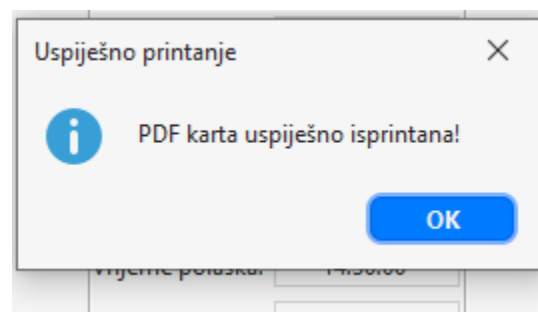
Podaci o letu:

Avion:	F-15
Polazni grad:	London
Odredišni grad:	Manchester
Datum polaska:	2024-09-14
Vrijeme polaska:	14:30:00
Ukupna cijena:	Boeing 737

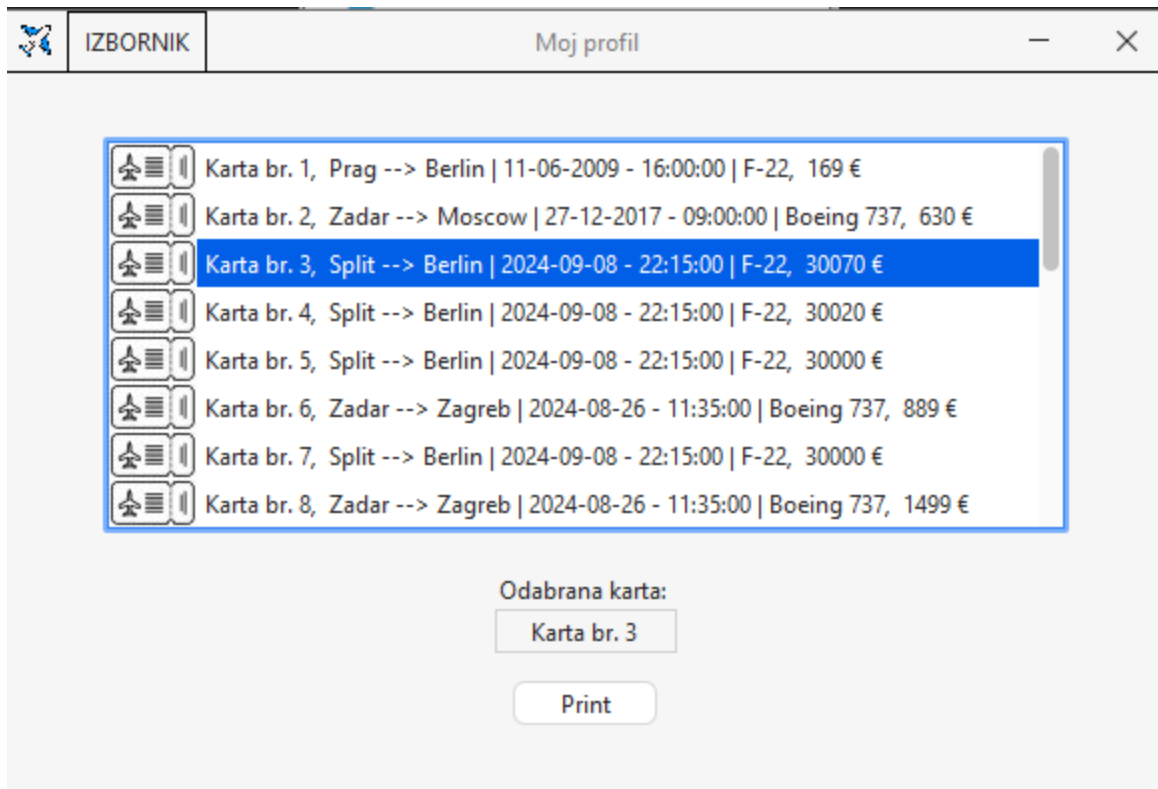
Potvrdi

PDF karte

Okvir koji prikazuje podatke o letu za koji je korisnik kupio kartu. Mogućnosti na ovom okviru su: 'Potvrdi', klasično *confirm* dugme koje korisnika vraća na UserFrame. PDF karte je JButton element koji stvara PDF dokument u kojem su prikazani podatci o letu i zahvala na kupovini.

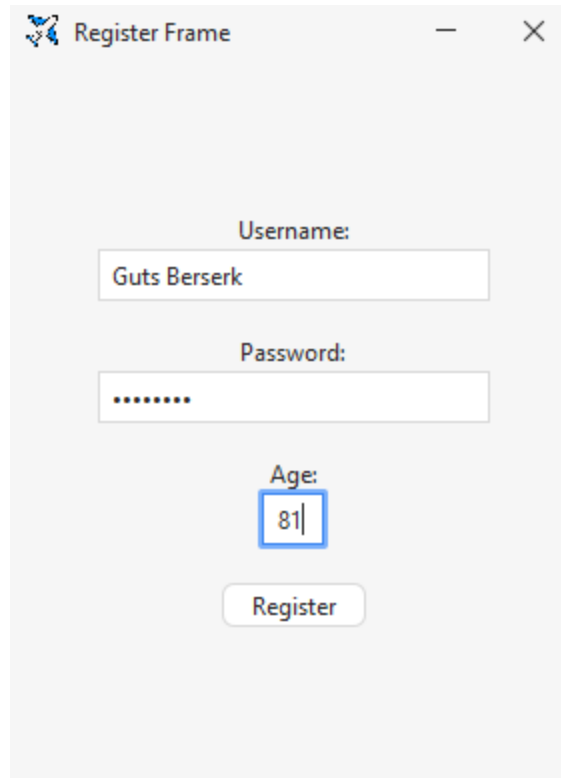


Slika 5: Potvrda o printanju karte



Moj Profil

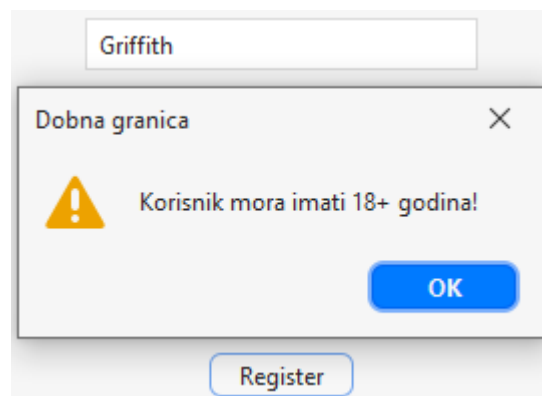
Moj profil je okvir koji sadrži sve karte koje je korisnik kupio unutar JList elementa. Ovaj okvir omogućuje korisniku printanje bilo koje kupljene karte u PDF oblik, u stvarnom svijetu to bi se moglo koristiti ako korisnik želi fizičku kopiju karte ili je želi pokloniti.




A screenshot of a 'Register Frame' window. It contains three input fields: 'Username:' with the text 'Guts Berserk', 'Password:' with masked characters '.....', and 'Age:' with the value '81'. A 'Register' button is located at the bottom.

Register frame

Okvir koji omogućuje korisniku kreiranje novog profila. Jedino ograničenje na ovom okviru je dob korisnika; mora biti 18 ili više, a ne može biti iznad 99.



Slika 6: Pokušaj kreiranja profila s manje od 18 godina!

 IZBORNIK

Admin Frame

Dodavanje aviona

Ime Aviona:

Proizvođač:

☐ Prvi razred?

☐ Poslovni razred?

☐ Ekonomični razred?

Cijena Prvog Razreda:

Cijena Posl. Razreda:

Cijena Eko. Razreda:

Dodaj Avion

Dodavanje leta

Izberi avion:

Datum:

Vrijeme polaska:

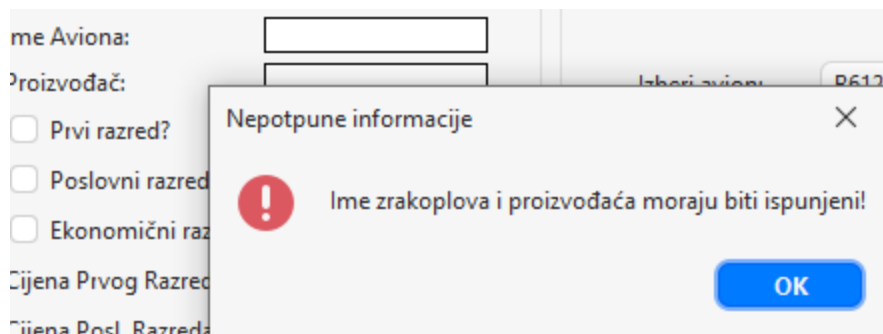
Polazište:

Odredište:

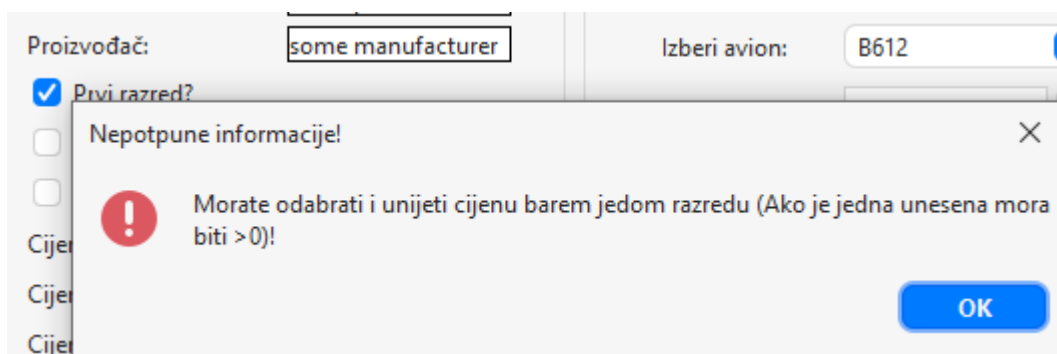
Dodaj let

AdminFrame

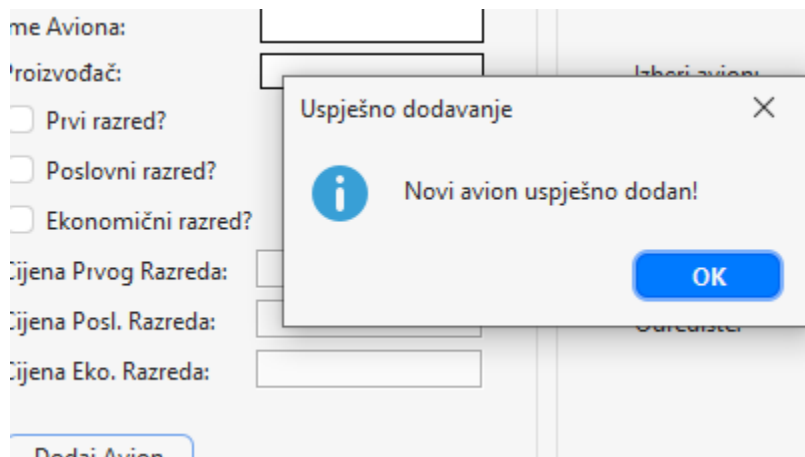
Okvir koji se pojavljuje samo pod uvjetom da korisnik koji se ulogira ima ulogu 'admin' unutar baze podataka. Omogućuje stvaranje novog zrakoplova i unošenje novih letova za korisnike. Prilikom kreiranja zrakoplova ime aviona i proizvođača moraju biti uneseni i barem jedna klasa mora biti postojeća (i njena cijena unesena) inače administrator dobiva odgovarajuće obavještajne greške. Za dodavanje leta imamo JComboBox na kojem su dostupni svi avioni koji nisu dodjeljeni letu (to krši izjavu o bazi podataka da jedan avion može imati više letova, ali ovo je napravljeno da se pokaže poznavanje database 'left join' query-a). Prve tri opcije imaju zadan izbor od kojih se svi mogu mijenjati. Datum ili vrijeme u neispravnom obliku ispisuju grešku i administrator je obavješten ukoliko su polazište ili odredište ne ispunjeni.



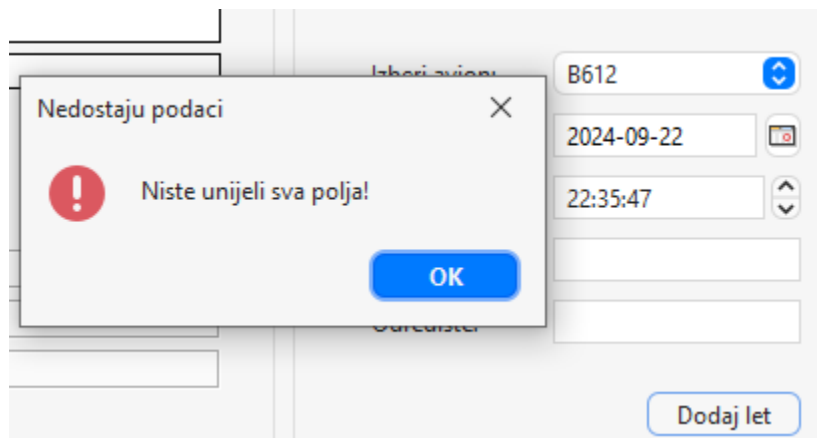
Slika 7: Rezultat stvaranja aviona bez unosa imena aviona i proizvođača



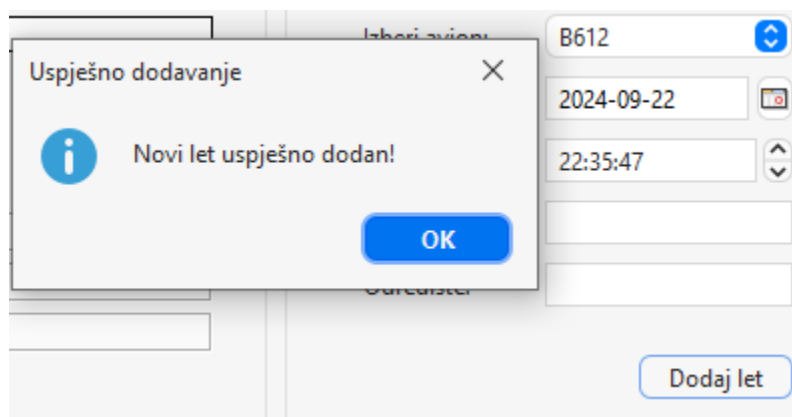
Slika 8: Unos imena nije dovoljan, potrebno je unijeti i cijenu za stvaranje aviona!



Slika 9: Kada su zadovoljeni svi uvjeti novi avion se dodaje u bazu podataka. Forma unosa se resetira



Slika 10: Neispravna forma unosa novog leta, odredište i dolazište su prazni!



Slika 11: Uspješno dodavanje leta u bazu, dio forme se resetira

Upravljanje nezavisnostima (Maven)

MySql connector java – Omogućuje komunikaciju Java aplikacija s MySql bazama podataka. Pomoću nje sam postavljao upite na bazu.

- MVN repository url - [mysql-connector-java](#)
- Službena dokumentacija - /
- Ne službeni izvori – StackOverflow, YouTube, Maven Central

FlatLaf – Omogućuje promjenu zadanog izgleda (teme) Java Swing aplikacija. Kroz projekt sam koristio MacLightTheme i MacDarkTheme.

- MVN repository url - [FlatLaf](#)
- Službena dokumentacija - [doc](#)
- Ne službeni izvori – StackOverflow, Youtube

ITextPDF – Omogućuje zapisivanje podataka u PDF datotečni oblik. Postojalo je nekoliko opcija, ali na preporuke članova foruma StackOverflow odabrao sam ovaj.

- MVN repository url - [ITextPDF](#)
- Službena dokumentacija - [doc](#)
- Ne službeni izvori – StackOverflow

Google zxing core i javase – Korišteni su za kreiranje QR koda. To nije nigdje taženo u zadatku, ali s obzirom da je to jednostavan i učinkovit način pohrane male količine podataka odlučio sam ga inkomponirati.

- MVN repository url - [Javase](#) i [core](#)
- Službena dokumentacija – [github repo core](#)
- Ne službeni izvori – Youtube

JCalendar – *Library* koji omogućuje korisniku unos datuma. Potreban pri kreiranju leta gdje admin unosi datum leta.

- MVN repository url - [jcalendar](#)
- Službena dokumentacija - [doc](#)