Projektni zadatak

iz predmeta

Razvoj programskih rješenja

Aplikacija za biblioteku

Autor:
Mušić Faris

Elektrotehnički fakultet
Sarajevo

Sadržaj

Opis aplikacije

Aplikacija za biblioteku je namjenjena za upravljanje bibliotekom. To uključuje vođenje evidencije o korisnicima biblioteke, zaposlenicima biblioteke koji u sistemu predstavljaju administratore, zatim vođenje evidencije o knjigama i nekim osnovnim njihovim informacijama. Zatim aplikacija služi za vođenje evidencije o iznajmljivanju knjiga od strane korisnika.
U sistemu postoji tri vrste korisnika:

1. Obični korisnik koji se pomoću korisničkog imena i lozinke može ulogovati na svoj profil (nakon registracije), vidjeti koje knjige u tom trenutnu ima podignute i koji je rok za vraćanje tih knjiga. U slučaju da korisnik ne vrati na vrijeme knjigu, sa lijeve strane mu iskače obavijest/upozoranje o tome da je prošao rok za vraćanje knjige i ta obavijest je vidljiva sve dok korisnik ne vrati knjigu u biblioteku. Korisnik ima mogućnost da vidi sve dostupne knjige u biblioteci, njihove podatke i koliko u tom trenutku ima na stanju primjeraka neke knjige.

2. Administrator predstavlja zaposlenika biblioteke koji se također pomoću korisničkog imena i lozinke može ulogovati na svoj profil. Administrator može dodavati, mijenjati i brisati knjige, s tim što u slučaju da neka knjiga još nije vraćena od strane nekog korisnika, iskače obavijest i tu knjigu nije moguće izbrisati iz sistema sve dok se svi primjerci te knjige ne vrate u biblioteku. Također, administrator ima opciju da briše korisnike, s tim što korisnika koji nije izmirio svoje obaveze prema biblioteci nije moguće izbrisati. **Administrator upravlja i iznajmljivanjem knjiga na način da prvo klikne na željenog korisnika i tek onda može da mu dodijeli neku knjigu, da vidi sva zaduženja tog korisnika i da razduži knjigu od korisnika.**

3. ROOT korisnik je ustvari vlasnik sistema koji samo ima mogućnost dodavanja i brisanja administratora.

Baza podataka je inicijalno prazna tj. ne postoji nijedan registrovani korisnik i admin.
Na samom početku korištenja aplikacije potrebno se ulogovati na ROOT profil sa sljedećim pristupnim podacima:

Username: root
Password: %&/123RooT123()=

Nakon logovanja je potrebno registrovati jednog administratora. Nakon toga se možemo odjaviti sa root profila i registrovati jednog običnog korisnika. Nakon toga aplikacija je spremna za testiranje.

Baza podataka

Za realizaciju aplikacije je korištena SQLite baza podataka baza.db. U slučaju da inicijalno baza ne postoji, postoji datoteka sa nazivom baza.db.sql koja služi za generisanje baze i tabela unutar baze podataka. Za svrhu ovog projekta u bazi podataka se nalaze 4 tabele, a to su: rentings, books, admins i users koje čuvaju podatke o , respektivno, iznajmljivanju knjiga, knjigama koje se nalaze u biblioteci, registrovanim korisnicima i administratorima. Klasa BibliotekaDAO je klasa koja ima direktan pristup bazi, pomoću upita dobavlja podatke iz baze i prosljeđuje ih drugim klasama, ažurira podatke u bazi i upisuje nove podatke u bazu.

Grafički interfejs

Za kreiranje grafičkog interfejsa je korištena JavaFX biblioteka. Prilikom kreiranja grafičkog interfejsa aplikacije korišteni su razni GUI elementi kao što su BorderPane, GridPane, MenuBar, Menu, MenuItem, Label, Button, TableView, ListView, ChoiceBox, TextField, PasswordField i sl. Za svaki prozor kreirana je Controller klasa koja je odgovorna za svu interakciju između sistema i korisnika sistema. Prilikom registracije korisnika i admina vršene su razne validacije polja poput dužine lozinke, validne forme emaila, dužine korisničkog imena i sl. Pomoću fajla validation.css se postavlja odgovarajuća boja (zelena ili crvena) na formu u zavisnosti od validnosti forme. Prilikom izrade aplikacije je korištena programska paradigma **programiranje vođeno događajima (event-driven programming)** gdje je većina procedura uzrokovano događajima izvana tj. korisničkim akcijama. Uzeti su u obzir koncepti dobrog dizajna korisničkog interfejsa (Gestalt principi) gdje su svi logički vezani elementi grupisani i na ekranu.

Internacionalizacija

Da bi aplikacija bila dostupna na više jezika uveli smo internacionalizaciju koja nam nudi korištenje engleskog i bosanskog jezika unutar aplikacije. Odmah na početku nam je ponuđeno da izaberemo koji od jezika želimo koristiti i tu odluku možemo promijeniti svaki put kada pokrenemo aplikaciju. Unutar foldera resources je kreiran ResourceBundle Translation koji sadrži tri fajla **.properties** u kojima se nalaze prevodi na bosanski i engleski jezik i fajl u kojem se nalazi defaultni jezik.

Dijagram klasa

Dijagram klasa je generisan uz pomoć Intellij okruženja. Nakon završetka kodiranja desnim klikom na paket selektujemo Diagrams | Show Diagram | Java Class Diagram i dobivamo gotov dijagram klasa za našu aplikaciju. Unutar projekta nalaze se dva fajla UML.png i UML.uml na kojima je prikazan dijagram klasa. Prikazane su sve klase, konstruktori i atributi za svaku klasu i naravno veze između klasa.

Kolekcije podataka, threadovi, streamovi i funkcionalno programiranje

Što se tiče kolekcija podataka u većini slučajeva su korištene liste (List<Object>) za čuvanje objekata. Također, u par navrata je kao kolekcija podataka korišten i Set, tačnije TreeSet u slučaju kad je bilo potrebno da kolekcija podataka bude sortirana po nekom kriteriju. Pri iznajmljivanju knjige nam se pojavljuje lista sa svim knjigama koja je abecedno sortirana i upravo za tu priliku je iskorišten TreeSet uz prethodno implementiran interfejs Comparable u klasi Book i metodu compareTo.
S obzirom da sve potrebne podatke dobivamo iz baze podataka preko odgovarajucih upita kojim mozemo odmah profiltrirati trazene podatke, nije bilo velike potrebe za koristenje petlji i streamova za filtriranje podataka, ali u klasi BibliotekaDAO je na par mjesta demonstrirana upotreba streamova.
U klasi LoginController su iskoristeni threadovi gdje se pokrecu dva threada koja validiraju unesena polja korisnicko ime i lozinku.
Lambda funkcije su korištene u streamovima i pri inicijliziranju određenih kolona u tabelama.

Datoteke

Root ima mogucnost, klikom na button, generisanja .txt datoteka u koje se upisuju svi administratori i svi korisnici sistema (u dvije različite datoteke). Klase korištene za realizaciju su FileOutputStream, ObjectOutputStream i još neke metode klase File koje stvaraju datoteku, provjeravaju da li već postoji datoteka sa tim imenom i sl. Također, da bi ovo funkcionisalo potrebno je u klasama Administrator i User implementirati interfejs Serializable.

Alati za automatsku izgradnju koda

Kreirana je JAR datoteka (datoteka u kojoj su zapakovane sve .class datoteke) kroz okruženje Intellij. Za main klasu je odabrana klasa sample.Main. Naziv JAR datoteke je rpr-projekat.jar i nazazi se u out/artifacts kao ZIP datoteka.