UNIVERZITET U SARAJEVU

ELEKTROTEHNIČKI FAKULTET

RAČUNARSTVO I INFORMATIKA

IZVJEŠTAJ O PROJEKTU

BILJEŠKE S PREDAVANJA

RAZVOJ PROGRAMSKIH RJEŠENJA

Student: Begović Amila

Broj indeksa: 18608

Predmetni nastavnik: Doc. Dr. Vedran Ljubović

Sarajevo, Septembar 2021.

Sadržaj

1. Zahtjevi klijenta

Tema projekta je „Bilješke s predavanja“. Pošto je klijent naziv aplikacije ostavio na izbor, odlučila sam se da joj dam ime „Nota“. U intervjuu s klijentom dobila sam uvid koje specifikacije ova aplikacija treba da posjeduje.

- Aplikacija bi trebala biti prilagođena bilješkama sa časova, vježbi i tutorijala.
- Korisnici bi trebali imati mogućnost kreiranja novog korisničkog računa, te pristup istom.
- Korisnici ne bi smjeli da pristupaju aplikaciji kao gost.
- Potrebno je omogućiti korištenje aplikacije i na bosanskom i na engleskom jeziku.
- Svaka kreirana bilješka mora pripadati nekoj grupi koju korisnik ima mogućnost da sam kreira.
- Pored grupa i bilješki, korisnik može kreirati i vlastite oznake za bilješke.
- Bilješka treba da posjeduje datum kreacije i datum zadnjeg editovanja. Naziv bilješke je obavezan i automatsko je veliko početno slovo.
- Potrebno je kreirati neku vrstu editora za editovanje bilješki, te dodavanje slike unutar bilješke. Promjena boje bilješke također treba da bude implementirana.
- Treba omogućiti pretraživanje bilješki po grupama i oznakama, kao i sortiranje bilješki.
- Aplikacija treba da sadrži i dugme za postavke.

2. Opis aplikacije

Pri pokretanju aplikacije korisniku je omogućeno ulogovanje u svoj račun. Ukoliko nema kreiran račun, korisnik ima mogućnost registrovanja vlastitog računa. Prilikom registracije potrebno je da unese svoje ime, prezime, korisničko ime, validnu e-mail adresu, te šifru. Email adresa i korisničko ime trebaju da budu jedinstveni. Korisničko ime mora sadržavati barem 5 karaktera, dok šifra treba da bude duža od 8, a kraća od 25. Treba sadržavati barem jedno malo slovo, jedno veliko slovo, jedan broj i jedan specijalni karakter (@#$%). Prilikom uspješne registracije ili ulogovanja, korisnik ima pristup svojim bilješkama.

Na početnom ekranu aplikacije korisnik može da vidi sve svoje kreirane bilješke, grupe i oznake. Grupe i oznake se mogu sortirati po datumu kreiranja, imenu i opisu, dok se bilješke mogu sortirati i po datumu zadnjeg editovanja. Na ekranu postoji i polje za unos teksta koji filtrira sve korisnikove bilješke, tako što pronalazi uneseni tekst u nazivu ili opisu bilješke.

Za kreiranje bilješki korisnik mora posjedovati barem jednu grupu iz razloga što bilješka ne može egzistirati izvan iste. Jedna bilješka može pripadati samo jednoj grupi, ali može podsjedovati više ili ni jednu oznaku. Prilikom kreacije grupa i oznaka korisnik ima na raspolaganju da postavi naziv, opis i boju grupe/oznake. Naziv i boja su obavezni, dok je opis opcionalan. Za naziv se postavlja automatsko veliko slovo, te mora biti jedinstven za grupe i za oznake. Grupe i oznake se mogu i naknadno uređivati, kao i obrisati. Prilikom brisanja grupe, trajno se brišu i sve bilješke koje joj pripadaju. Pored uređivanja i brisanja svaka grupa i oznaka se može spremiti u tekstualni dokument.

Prilikom kreiranja biljeških korisk mora zadati naziv bilješke, boju i kojoj grupi bilješka pripada. Za naziv bilješke se automatski postavlja prvo veliko slovo. Korisnik može da bilješku označi sa bilo kojom oznakom. Korisnik ima i opciju dodavanja teksta za bilješku, te pripadajuću sliku. Korisnik ima na raspolaganju i mali editor za uređivanje teksta biljeških. Nakon kreacije, bilješka će biti vidljiva na ekranu ukoliko je izabrana grupa ili oznaka kojoj bilješka pripada. Baš kao i grupe i oznake, bilješke je moguće editovati, brisati i spremiti u tekstualni dokument, što se obavlja preko menija koji je dostupan prilikom pregledavanja bilješke.

Na glavnom ekranu se pored svih navedenih polja, nalazi i dugme za odlogovanje, te dugme za postavke aplikacije. U postavkama aplikacije moguće je promijeniti jezik, te ažuriranje korisnikovih podataka. U postavkama je moguće izvršiti i brisanje računa.

Brisanje računa se može izvršiti i preko menija na glavnom ekranu. Pored brisanja postoje i opcije spremanja svih korisnikovih podataka u tekstualni dokument, printanje biljeških, grupa i oznaka preko pdf dokumenta, pristup postavkama, te pristup korisničkom vodiću i stranici about. Preko menija se mogu također kreirati i nove bilješke, grupe i labele.

3. Osnovne ideje pri implementaciji

U osnovi aplikacije nalaze se klase koje prate takozvanu Java Bean specifikaciju. Java Bean klase (Account, Group, Label i Note) povezuju podatke iz baze i objekata unutar aplikacije. Pored njih kreirane su tri enum klase: GroupColor, LabelColor i NoteColor koje predstavljaju boju koju posjeduje određena grupa, oznaka i bilješka respektivno. Svaki element enuma predstavlja boju, te sadrži heksadecimalni broj te boje. Heksadecimalni broj se može promijeniti veoma lahko, jer nije vezan za ostatak programa, s čime je ostvarena enkapsulacija. Pored te prednosti, veoma je lahko dodati nove boje. U aplikaciji su kreirana tri enuma za boje, da boje grupa, oznaka i bilješki mogu biti drugačije nijanse, ukoliko je za tim želja i potreba. Kreirana je i enum klasa za editovanje bilješki.

Pri implementaciji projekta korištena je SQLite baza podataka. Veza između baze podataka i projekta postignuta je klasom DatabaseConnection. To je singleton klasa koja se bavi kreacijom baze podataka iz sql dump datoteke, brisanja svih podataka iz baze, te kreira konekciju na bazu podataka. Klasa DatabaseConnection nema više uloga, te se podacima iz baze podataka pristupa preko odgovarajućih model klasa. Model klase posjeduju potrebne upite za pristup podacima.

Baza podataka projekta posjeduje šest tabela, od kojih su četiri glavne tabele, jedna međutabela i jedna pomoćna. Za svaku od četiri glavne tabele kreirana je pripadajuća model klasa: AccountModel, GroupModel, LabelModel i NoteModel. Prva model klasa pored upita vezanih za svoju tabelu posjeduje i statički atribut koji predstavlja trenutno logovanog korisnika, te se trenutnom korisniku može pristupiti iz svih klasa. Ostale tri model klase posjeduju Observable atribute, koji se koriste u kontrolerima zbog konzinstencije podataka.

Kreacijom modela i DatabaseConnection klase postignuto je odvajanje podataka iz baze od ostatka aplikacije.

Pored DatabaseConnection klase, projekat posjeduje i dvije dodatne singleton klase. Prva singleton klasa je MyResourceBundle, koja se koristi za pristup ResourceBundle, te

omogućava da svaki novokreirani prozor koristi isti Locale. Druga kreirana singleton klase je MyHostService. Klasa je kreirana zbog about prozora koji zahtjeva HostService za otvaranje url linkova. Pošto je HostServicu moguće pristupiti samo u Main klasi, ova singleton klasa je omogućila da se atribut ne mora držati kao parametar u kontrolerima.

U aplikaciji je moguće kreirati i tri različita izvještaja, što je urađeno preko klase PrintReport i Jaspersoft studia. Za validaciju podataka kreiran je i vlastiti izuzetak AccountValidationFailedException. Postoje i dodatne model klase za vrijednosti unutar ChoiceBox.

4. Implementacija

U nastavku slijede detaljnjiji opisi izabranih dijelova aplikacije, kako bi se objasnila zastupljenost gradiva na predmetu Razvoj programskih rješenja.

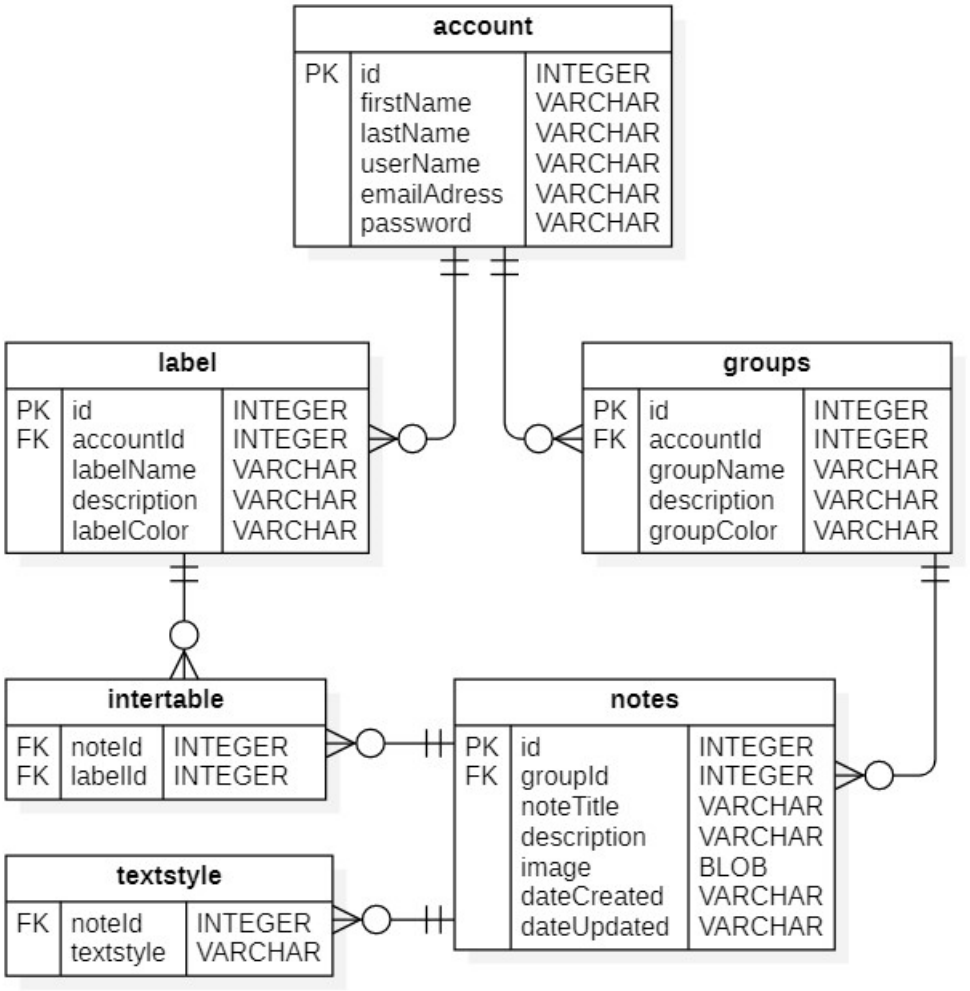
4.1. Korišteni patern

Tokom kreiranja klasa i pogleda unutar aplikacije, korišten je DAL + MVC patern. Za potrebe ovog paterna kreirane su slijedeće cjeline:

- DTO paket:
    - Sadrži Data Transfer Objekte, tj obične podatkovne JavaBean klase u kojima će se držati podaci iz baze.
- DAL paket:
    - Sadrži DatabaseConnection klasu – jedna singleton klasa koja uspostavlja konekciju na bazu, te se bavi očuvanjem baze podataka.
    - Sadrži Data Access Objekte – singleton model klase za pojedinačne tabele unutar baze podataka. U model klasama se nalaze upiti za pristup podacima iz baze podataka, a atributi su tipa ObservableList ili SimpleObjectProperty.
    - Sadrži model klase za korištenje ChoiceBox polja unutar kontrolera. Ti modeli sadrže samo Observable objekte, bez ikakvog pritupa bazi podataka.
- Controller paket:
    - Sadrži sve Controller klase. Svaki kreirani pogled posjeduje svoj vlastiti kontroler koji upravlja njegovim poljima.
- Utilities paket
    - Sadrži pomoćne klase za rad projekta: PrintReport, MyResourceBundle, MyHostServices i AccountValidationFailedException.

4.2. Baza podataka

Za potrebe ovog projekta korištena je SQLite baza podataka pod nazivom projectdatabase.db, koja sadrži četiri glavne tabele: account, notes, groups i label, te međutabelu intertable i pomoćnu tabelu textstyle. Međutabela služi kao veza više na više između tabela label i notes, dok pomoćna tabela čuva podatke o editovanju bilješki.



Veza sa bazom počinje u singleton klasi DatabaseConnection, čiji je cilj da uspostavi vezu sa bazom i da regeneriše bazu ukoliko je potrebno. Unutar DatabaseConnection se postavlja konekcija za sve DAO modele, te se instancama model klasa pristupa iz DatabaseConnection. Nakon što model preuzme konekciju iz DatabaseConnection on kreira sve PreparedStatements potrebne za rad. Ukoliko se unutar neke model klase desi izuzetak, DatabaseConnection će uhvatiti izuzetak i obraditi ga.

4.3. Koncepti objektno orijentisanog programiranja

U objektno orijentisanom programiranju postoje tri koncepta: nasljeđivanje, enkapsulacija i polimorfizam.

Nasljeđivanje u ovom projektu je korišteno pri kreiranju ListView grupa i oznaka. ListView ima poseban izgled i to je ostvareno na način da je prvo kreiran poseban fxml dokument koji predstavlja jedno polje unutar ListView. Nakon toga je bilo potrebno kreirati dva kontrolera GroupListCellController i LabelListCellController. Oba kontrolera su naslijedila klasu ListCell, te preklopili metodu updateItem. S tim je postignuto da se kontroleri mogu povezati sa ListView i time promijeniti izgled čelije liste. Na ovom mjestu je vidljiv i polimorfizam, jer se umjesto ListCell objekta koriste naši kreirani kontroleri.

Enkapsulacija je postignuta u kreiranim enum klasama, o čemu se govorilo u poglavlju 2. Pored toga, enkapsulacija je postignuta i DAL + MVC paternom (Iako se unutar model klasa nalaze Observable objekti, listeneri su postavljeni u kontrolerima, te su dvije cjeline odvojene). Baza podataka se može promijeniti, a izmjene bi se dešavale samo unutar DAL (i DTO) cjeline. Možemo promijeniti dešavanja unutar kontrolera, a da to se ne odražava na DAL cjelinu. Enkapsulacija je vidljiva i u DTO klasama. Iako su atributi tipa SimpleStringProperty, to ostatku programa nije vidljivo.

4.4. Grafički interfejs

Prilikom izrade grafičkog interfejsa korištene su mnoge javafx kontrole:

- Kontenjeri: GridPane, Vbox, Hbox, ScrollPane, FlowPane itd.
- Kontrole: Buttons, ChoiceBox, CheckBox, RadioButton, ImageView, HyperLink, ListView, Label, TextField, TextArea, PasswordField, Separator itd.
- Menu: Menu, MenuItem, SeparatorMenuItem

Na glavnom ekranu, kao i na ekranima za bilješke, grupe i oznake postoje meniji koji sadrže smislene radnje. Na ekranu za bilješke nalazi se i Toolbar za editovanje biljeških.

Primijenjeni su slijedeći dobri principi dobrog dizajna:

- Korišteni su Gestalt princip prikom grupisanja elemenata:
    - Udaljenost: Objekti koji su međusobno bliski izgledaju kao grupa
        - Npr: u gornjem desnom čošku glavnog ekrana se nalaze podaci o trenutno prijavljenom korisniku

    - Sličnost i simetrija
        - Npr: dugmad za kreaciju biljeških, oznaka i grupa izgledaju kao jedna grupa
    - Figura/pozadina:
        - Npr: na glavnogm ekranu u prvi plan dolaze bilješke zbog kontrasta između boja biljeških i pozadine

- Grafički dizajn je konzinstentan:
    - Koriste se sistemski dijaloški prozori (Alert: Confirmation i Error)
    - Sve poruke greške izgledaju i ponašaju se isto
    - Pomoću CSS fajlova je postignuto da sve labele, dugmad, choiceBox izgedaju isto
    - Meniji za oznake, grupe i bilješke imaju iste elemente, koji rade iste stvari
    - Meniji koriste dobro poznate akceleratore (npr: Ctrl + S)
    - Sličice na menijima i na prozorima su jasne oko toga šta predstavljaju
- Navigacija tastaturom:
    - Prilikom Enter-a izvršava se Ok dugme
    - Prilikom kretanja kroz ekran tastaturom kroz polja formulara se ide odozgo prema dole

Pored navedenih principa dobrog dizajna prilikom kreiranja grafičkog interfejsa pažnja je posvećena da aplikacija bude „*responsive*“. Prilikom korištenja radio dugmadi korištene su Toogle grupe i kreirana je pomoć korisniku (Help).

4.5. Validacija podataka

Pošto u ovoj aplikaciji postoji mnogo stvari koje se kreiraju, validacija je bila izuzetno potrebna. Za korisnički račun postoji validacija za email adresu, korisničko ime i šifru što je postignuto preko regexa unutar DTO klasa. Prilikom registracije vrši se i validacija da li je šifra ista kao i ponovljena šifra, te da li već postoji korisnik s tom email adresom ili korisničkim imenom. Iste te validacije se rade i prilikom ažuriranja korisničkog računa.

Prilikom izabira boje za bilješke, oznake i grupe koristi se ChoiceBox, kao i za izabir grupe prilikom kreiranja bilješke. Za izabir oznaka koriste se CheckBox. Prilikom unosa datuma u bazu podataka koristi se formater koji je definiran unutar model klase, da ne bi došlo do greške pri čitanju. Ukoliko dođe do nekog pogrešnog unosa od strane korisnika (pogrešno korisničko ime) to se provjerava tek nakon klika na dugme Ok, nakon čega se jasno prikazuje gdje je nastao problem.

4.6. Resources

Pored fxml datoteka u resources folderu su smještene css datoteke koje se koriste za izgled ekrana, postavljanje sličica na dugmad, te za pomoć pri validaciji polja. Unutar images foldera se nalaze sve slike korištene u aplikaciji, a reports folder služi za .jrxml fajlove o kojima će se govoriti u poglavlju 4.9.

4.7. Datoteke

U aplikaciji postoji opcija da se grupa, oznaka i bilješka, te svi korisnikovi podaci spreme unutar tekstualne datoteke, što je moguće postići preko Save menija. Pored toga, prilikom kreiranja bilješke postoji opcija izabira slike sa korisnikovog računara.

4.8. Funkcionalno programiranje

Kroz cijeli program se protežu lambda funkcije, koje se koriste za sortiranje i filtriranje listi, pronalaska određenog elementa unutar liste, za postavljanje listenera za polja itd.

4.9. Izvještavanje

U aplikaciji su korišteni Jaspersoft Studio alati za kreiranje izvještaja. Korisnik ima opciju da kreira 3 vrste izvještaja sa glavnog menija, koristeći Print opciju. Moguće je kreirati izvještaj o svim grupama, o svim bilješkama i o svim oznakama koje korisnik posjeduje. Pomoću klase PrintReport i parametra „AccountID“ unutar jrxml dokumenta, obezbjeđeno je da izvještaj bude samo za trenutno logovanog korisnika. Korišten je već postojeći parametar „REPORT_RESOURCE_BUNDLE“ za prosljeđivanje ResourcesBundle, da bi i se i izvještaj mogao koristit na dva jezika.

4.10. Lokalizacija

Za potrebu lokalizacije kreirana je singleton klasa MyResourceBundle, te Resource Bundle „Translation“ koja sadrži default, bosanski i engleski jezik. Za bosanski prijevod su iskorišteni unicode karakteri za ispravno prikazivanje neengleskih karaktera.

4.11. Izuzeci

Za potrebu validacije podataka prilikom kreacije i ažuriranja korisnikovih podataka kreiran je vlastititi Runtime exception AccountValidationFailedException.

4.12. Maven

Kreiran je Maven pom.xml fajl preko kojeg je kreiran i jar-with-dependencies za projekat. Maven je kreiran tako da ne uključuje testove, nego samo aplikaciju.

4.13. Testiranje

Za potrebe testiranja aplikacije kreirana je pomoćna singleton Utility klasa koja u sebi sadrži statički atribut „*testAccount*“ tipa Account. Taj korisnik je kreiran u bazi podataka sa ID *0* i on se koristi za testiranje aplikacije. Kreiran je iz razloga da se prilikom testiranja ne trebaju brisati svi podaci iz baze, te s tim obrisati svi podaci za sve korisnike unutar aplikacije, nego se trebaju samo obrisati podaci za testnog korisnika.

Podaci za testnog korisnika su slijedeći:

- Ime: Firstname
- Prezime: Lastname
- Korisničko ime: username
- E-mail: email@gmail.com
- Lozinka: Password10#

Za testiranje projekta korišten je testfx-junit5 Maven import, te junit.api import. Korištene su anotacije @BeforeAll, @BeforeEach i @AfterAll, te razne vrste anotacija iz junit.api.

Testovi su grupisani u 3 paketa.

- Testovi za DTO klase koje testiraju settere, gettere i kontrolere za JavaBean klase.
- Testovi za DAL klase koje testiraju modele i bazu
- Testovi za kontrolere koji testiraju sveukupnu aplikaciju koristeći @ExtendWith(ApplicationExtension.class) anotaciju.

4.14. Dokumentacija

Unutar repozitoija na githubu se nalazi ER-dijagram za bazu podataka i četiri klas dijagrama:

- Klas dijagram sa svim klasama unutar src foldera zajedno sa svim metodama
- Klas dijagrama sa svim klasama bez metoda
- Klas dijagram za DTO klase zajedno sa svim metodama
- Klas dijagram za DTO klase bez metoda

5. Dijelovi projekta koji nedostaju

U poglavlju 4 se govorilo o dijelovima gradiva predmeta koji su uklopljeni u projekat, ali neki dijelovi kao mrežno programiranje, tredovi, interfejsi i abstraktne klase nisu dio projekta. Pored tih dijelova programa sačuvanje slike za bilješke u bazi podataka nije implementirano. Font za Jasper izvještaje nije podešen za neengleska slova.

6. Prilike za unaprjeđenje

Jedan dio aplikacije gdje bi se kod mogao poboljšati je unutar kontrolera, jer tu postoji dosta koda koji se ponavlja, kao npr: metode za otvaranje drugih prozora. Aplikacija bi se poboljšala ukoliko bi se riješio dio sa slikom za bilješke, kao i riješio problem s fontom na izvještajima.Izlistavanje bilješki bi se mogao ubrzati koristeći tredove.