IZVJEŠTAJ

Student: Elma Šeremet, 18318

Predmetni nastavnik: Doc. dr. Vedran Ljubović

Tema: Aplikacija za upravljanje nastavnim materijalom

Elektrotehnički fakultet, Sarajevo

Sadržaj

1. Opis aplikacije
   1. Admin
   2. Profesor
   3. Student
2. Realizacija
   1. Java paketi
   2. Izuzeci
   3. Pravila imenovanja i pravila formatiranja
   4. Objektno orjentisano programiranje
   5. Metode klase Object
   6. Enum
   7. Unit testovi u Javi
   8. TestFX
   9. Kolekcije
   10. Funkcionalno programiranje
   11. Objektno orjentisani dizajn
   12. Grafički korisnički interfejs
   13. Organizacija koda kroz resurse
   14. Rad sa datotekama
   15. Mrežno programiranja
   16. Rad sa bazom podataka
   17. Višenitno
   18. Izvještavanje
   19. Internacionalizacija
   20. Alati za automatsku izgradnju koda
3. Opis aplikacije

U nastavku je objašnjena realizacija projekta na temu „Aplikacija za upravljanje nastavnim materijom“. Projekat je na samom početku baziran idejom C2, dok kroz realizaciju sam postavila dosta ograničenja i izmjena. Prilikom pokretanja aplikacije otvara se forma za prijavu, što znači da za pristup funkcionalnostima aplikacije korisnik mora imati napravljen račun.



Zbog toga, postoji Admin koji započinje „život“ aplikacije (username: admin, password: admin).

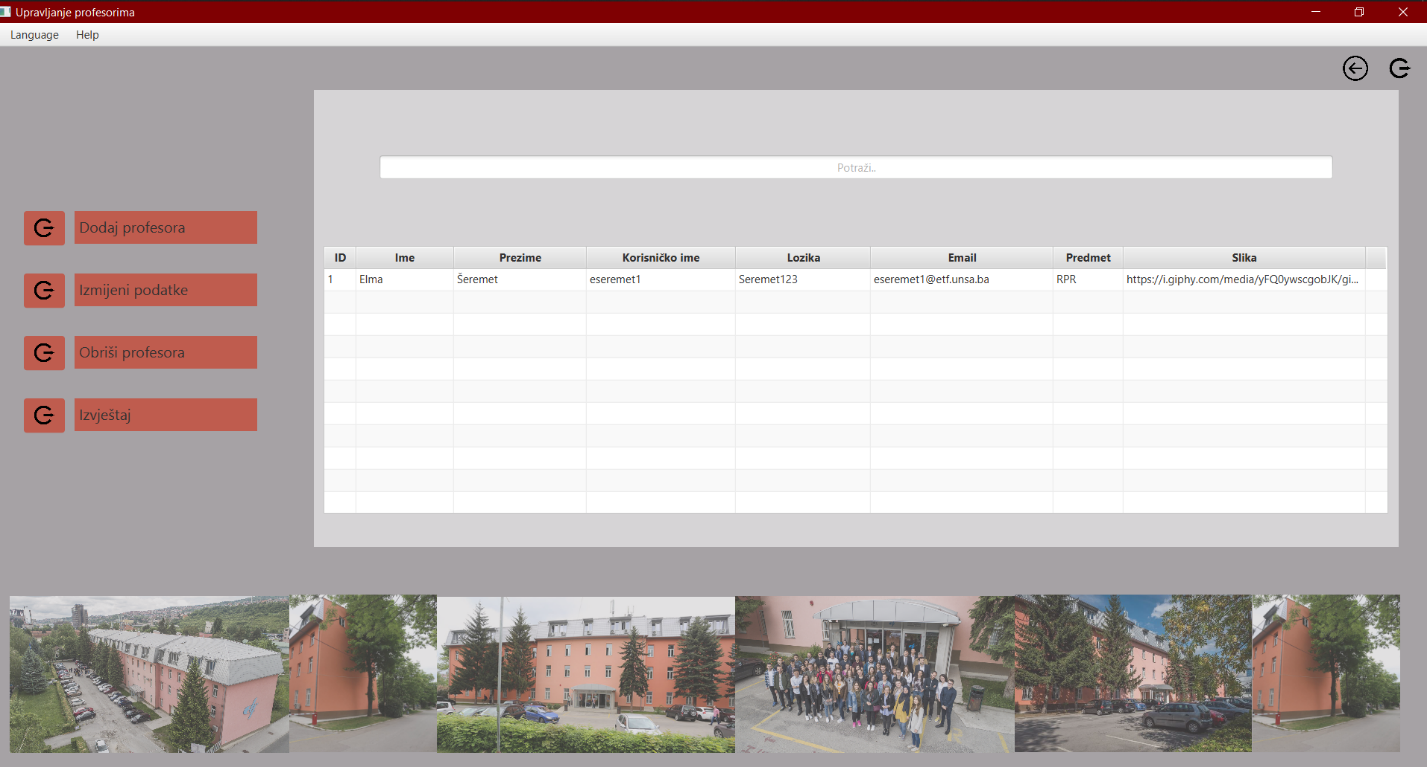
* 1. Admin

Admin je korisnik koji upravlja korisničkim računima profesora i studenata, te predmetima, što se može vidjeti odmah prilikom prijave sa gore unesenim podacima.



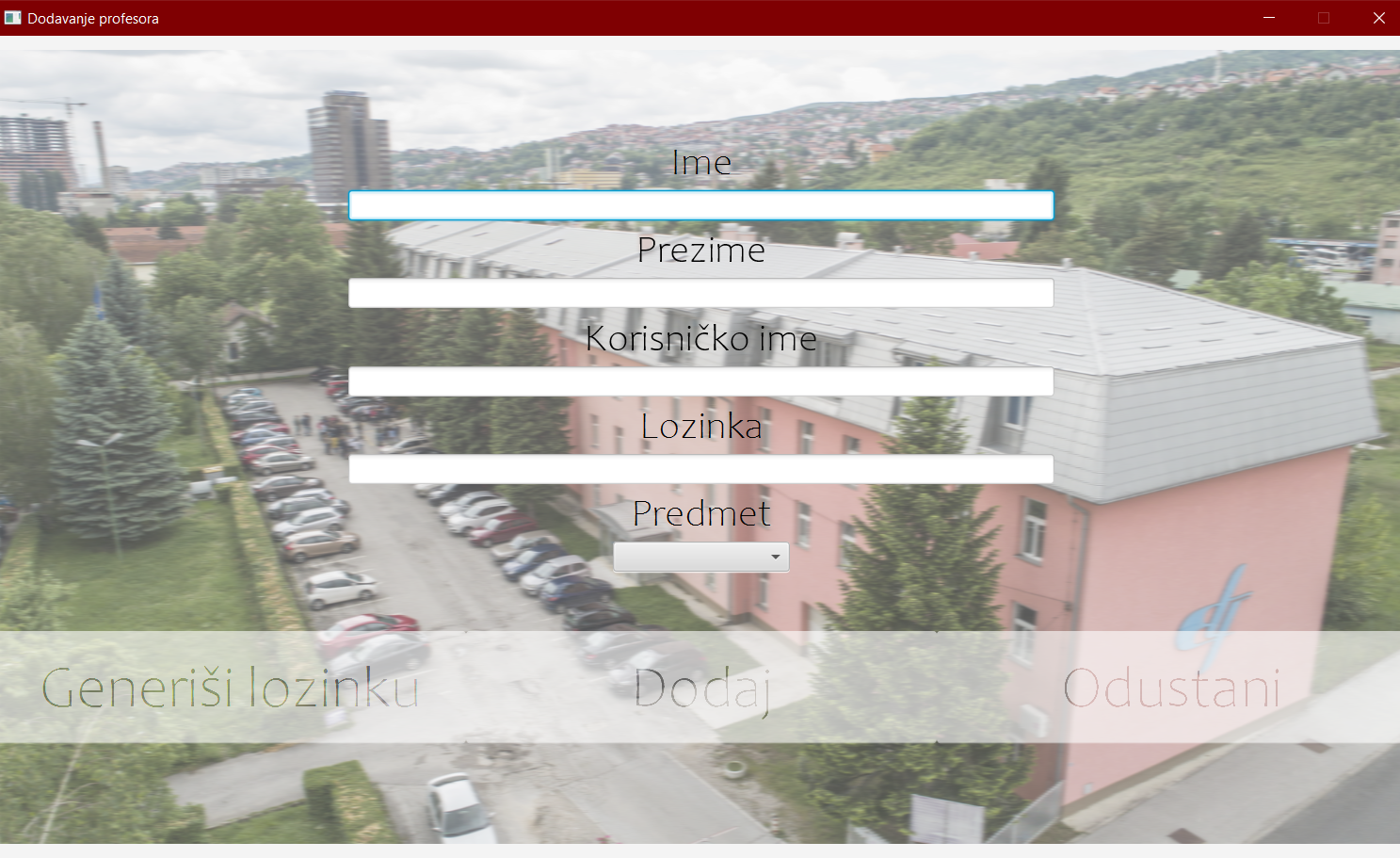
Budući da su funkcionalnosti za upravljanje bilo kojim podacima iste, bazirat ću se na upravljanje profesorima.

DODATI SLIKU SA ISPRAVNIM IKONAMA

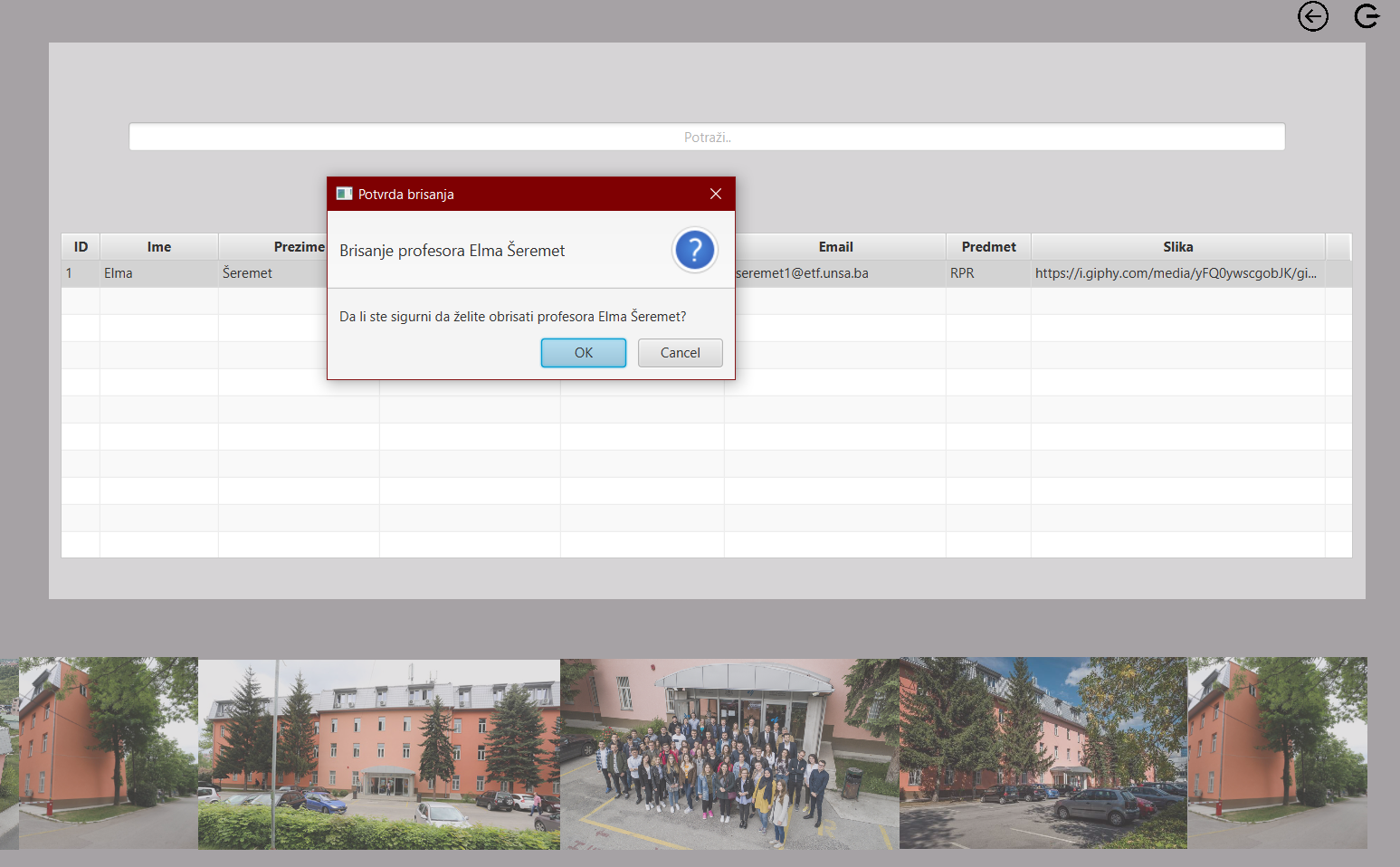


Dakle, kao što se može vidjeti na slici iznad, admin ima sljedeće mogućnosti:

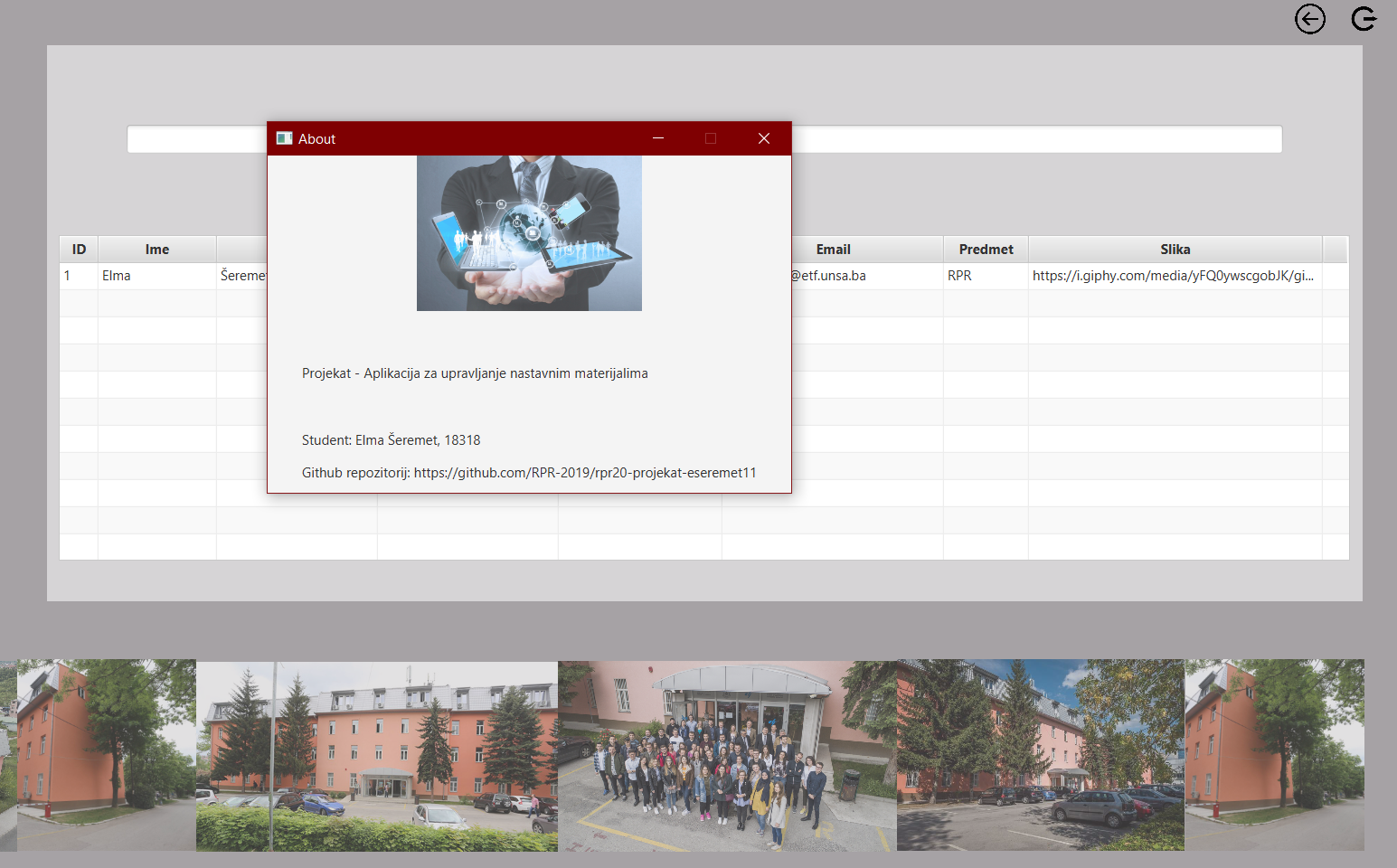
* Pregled svih profesora u bazi podataka
* Dodavanje novog profesora, pri čemu nije potrebno unositi podatke za email (budući da se email odredi automatski – [username+@etf.unsa.ba](mailto:username+@etf.unsa.ba)), lozinku (admin ima mogućnost samo generisanja slučajne lozinke, koja sadrži 8 znakova sa uključenim velikim, malim slovom, brojem i slučajnim znakom), te sliku (nju dodaje svaki korisnik posebno).



* Izmjena već postojoćeg profesora (slično kao dodavanje)
* Brisanje profesora iz baze



* Izvještaj svih profesora sa prikazanim podacima o ID-u, imenu, prezimenu i predmetu
* Pretraživanje profesora po određenim kolonama ukoliko je potrebno brzo pronaći podatke o određenom profesoru, a u bazi postoji dosta korisnika
* Promjena jezika (aplikacija je dostupna na engleskom i bosanskom jeziku)
* Prikaz osnovnih informacija o aplikaciji



* Navigacija
* Odjava

Kao što sam već navela, iste funkcionalnosti su moguće i nad studentima i predmetima.

Nakon kreiranja računa i predmeta, aplikacija je posmatrana iz dvije perspektive: perspektive profesora i studenta.

Naime, prilikom pokušaja pristupa aplikaciji, urađena je validacija unesenih podataka, gdje se provjera da li je polje prazno, te obavještenja o neispravnoj lozinci ukoliko korisnik sa napisanim username-om zaista postoji u bazi.

* 1. Profesor

Budući da je tema projekta vezana za upravljanje nastavnim materijalom, fokus je stavljen na profesora. Nakon uspješne prijave, postoji niz sljedećih funkcionalnosti:

* Prikaz svih predmeta sa lijeve strane, pri čemu su predmeti na kojim trenutno prijavljeni profesor predaje označeni sa zvjezdicom (npr. umjesto RPR, RPR\*)
* Dodavanje novog materijala (samo na predmet na kojem predaje)

Što se tiče ove funkcionalnosti, ona je podijeljena na dodavanje .pdf i .docx materijala sa računara, te pisanje .txt materijala u samoj aplikaciji. Ukoliko se prijavljeni profesor odluči sam napisati materijal, također ima opciju spremanja materijala na računar i dodavanje nekog .txt materijala sa računara (postoji opcija About gdje je također to objašnjeno). Također, bitno je spomenuti, da za svaki materijal može se odlučiti da li će on biti privatan ili javan (postoji i opcija CUSTOM, koja je ostavljena da ima smisao kao da je materijal privatan, budući da nisam uspjela smisliti na koji način bi bila najbolje iskorištena – prva ideja je bila da to mogu vidjeti samo profesori, ali smatram da to nije od neke koristi).

* Dodavanje kviza (samo na predmet na kojem predaje)

Profesor ima mogućnost dodavanja kviza prema unaprijed određenom formatu.

Napomena: Svaki dodani kviz ima nastavak „-QUIZ“ pri pregledu materijala, te ova funkcionalnost je dodatno uvedena da bih iskoristila čitanje datoteka, budući da profesor unese i tačan odgovor, a kod studenta su prikazana samo pitanja i ponuđeni odgovori bez tačnog odgovora.

* Prikaz materijala po predmetima

Napomena: Mora biti odabran bilo koji predmet!

* Prikaz samo kvizova
* Prikaz sortiranih materijala po nazivu
* Prikaz određenog već dodanog materijala kroz aplikaciju
* Brisanje materijala
* Prikaz materijala stavljenih na private
* Postavljanje skrivenih materijala javnima
* Prikaz profila sa osnovnih podacima
* Promjena slike koristeći Giphy
* Promjena lozinke (budući da je generisana lozinka teška za pamćenje, uvedena je mogućnost promjene, pri čemu je potrebno da nova lozinka ima barem 8 karaktera, od čega mora sadržavati barem jedno veliko, malo slovo, te broj).
* Promjena jezika
* Prikaz osnovnih informacija o aplikaciji
* Odjava
  1. Student

Student je korisnik koji ima funkcionalnosti:

* Pregled materijala po predmetima
* Prikaz samo kvizova (bez prikazanog tačnog odgovora)
* Prikaz sortiranih materijala po nazivu
* Pregled pojedinačnog materijala
* Prikaz profila sa osnovnim podacima
* Promjena slike koristeći Giphy
* Promjena lozinke (budući da je generisana lozinka teška za pamćenje, uvedena je mogućnost promjene, pri čemu je potrebno da nova lozinka ima barem 8 karaktera, od čega mora sadržavati barem jedno veliko, malo slovo, te broj).
* Promjena jezika
* Prikaz osnovnih informacija o aplikaciji
* Odjava

1. Realizacija

2.1. Java paketi

Paket koji je napravljen za projekat je ba.unsa.etf.rpr.project. Prilikom kreiranja paketa korišteno je pravilo da nazivi paketa imaju hijerarhijsku strukturu. Što se tiče uključivanje klasa iz drugih imenskih paketa, rađena je optimizacija.

* 1. Izuzeci

Pored korištenja bibliotečnih klasa izvedenih iz klase Exception, napravljen je i vlastiti izuzetak. Prilikom svakog izuzetka za koji je moguća situacija da se desi, vođeno je računa da on bude obrađen.

* 1. Pravila imenovanja i pravila formatiranja

Korišten je CamelCase. Imena promjenjivih i metoda su uvijek započeta malim slovom, dok su ostala početna slova riječi započeta velikim. Imena klasa uvijek započinju velikim slovom. Za svaku metodu i varijablu korištena su jasna imena zavisno na šta se odnosi. Korišten je K&R stil formatiranja. Također, svi nazivi u kodu su pisani na engleskom jeziku.

* 1. Objektno orjentisano programiranje

Kroz čitavu aplikaciju vođeno je računa da svi atributi budu privatni, te da im se može pristupiti samo pomoću kreiranih gettera i settera. Također, korišten je JavaBeans standard, pri čemu se svaki getter zove getAtribut, a svaki setter setAtribut. Budući da i profesor i student imaju većinu atributa zajedničkih korišteno je nasljeđivanje iz klase User. Ovo će pokazati svakako veliki značaj ukoliko dođe do potrebe da se dodaju još neke slične vrste korisnika (asistent, demonstrator i slično). Također, vođeno je računa i kod konstruktora naslijeđenih klasa, gdje su rađeni pozivi roditeljske klase, a zatim ostale potrebne naredbe. Is i has relacije (is kod nasljeđivanja), has sam često koristila, budući da dosta klasa sadrži referencu na neku drugu klasu (naprimjer, Profesor sadrži predmet na koji predaje). Što se tiče polimorfizma, on je iskorišten pri radu sa pregledom profila.

* 1. Metode klase Object

Pozivane su metode klase Object (equals), te također metoda toString je preklopljena u svim odgovarajućim klasama. Za ostale, nije se ukazala potreba.

* 1. Enum

U programu je korišten i pobrojani tip, pomoću kojeg je omogućena vidljivost materijala. Ovo je urađeno pomoću pobrojanog tipa, budući da bi se vremenom mogla javiti potreba da naprimjer neke materijale mogu vidjeti samo demonstratori, drugi profesori ili određeni korisnici koji imaju neku privilegiju. Budući da je enum klasa, koristila sam također CamelCase za imenovanje, a budući da su vrijednosti konstante, njihova imena su pisana velikim slovima (PUBLIC, PRIVATE, CUSTOM).

* 1. Unit testovi u Javi

Korištena je Junit 5 verzija. Za svaku klasu je dodana testna klasa, pri čemu čemu je vođeno računa da se testna klasa naziva ImeKlaseTest. Kreirano je 5 testnih klasa, te što se tiče kreiranja, praćen je način objašnjen na predavanjima, odnosno testne klase su kreirane automatski, pri čemu je pažnja obraćena na što veću pokrivenost. Kreirano je 33 Unit testa od čega svi prolaze. Što se tiče Junit anotacija, korištene su @BeforeEach i @Test. Za ostale nije bilo potrebe.

* 1. TestFX

Dodana je biblioteka org.testfx:testfx-junit5:4.0.15-alpha.

* 1. Kolekcije

Što se tiče kolekcija, zbog samog projekta, koji nije pretjerano složen, uglavnom je korišten ArrayList. Također, pored njega najviše su korišteni Set, TreeSet, SortedList i FilteredList.

* 1. Funkcionalno programiranje

U programu su korištene lambda funkcije gdje je ukazana potreba za njima (najviše pri kreiranju listenera). Iskorišten je i gotovi funkcionalni interfejs Predicate<T>, prilikom prikaza materijala koji su dodani kao kviz u aplikaciju. Također, u programu su iskorišteni i streamovi gdje god je za njima ukazana prilika, tipa za prikaz materijala po određenim predmetima, prikaz materijala za određenog profesora, prikaz materijala koji su skriveni, javni i slično..

* 1. Objektno orjentisana analiza i dizajn

Nakon obavljenog intervjua, dobila sam obavezne korisničke zahtjeve koji su mi bolje ukazali korisnikova očekivanja od aplikacije. Iako je bilo dosta izmjena, prilično sam dobro na početku procijenila koje će mi sve klase biti potrebne, te koje atribute trebaju otprilike sadržavati. Što se tiče klasa, te njihovih atributa i metoda, koristila sam se nomenklaturom objašnjenom na predavanjima (sve klase imaju naziv u prvom licu jednine, metode su imperativne tvrdnje, dok svaki naziv atributa jasno objašnjava atribut na koji se odnosi). Dijagram klasa sam ostavila za kraj, te iskoristila okruženje i njegovo automatsko generisanje, tako da nisam morala voditi računa o ispravnom pisanju, anotacijama i slično, budući da je to program sve uradio umjesto mene. Dijagram klasa sam dobila u UML design tool window-u. Budući da je dijagram klasa ogroman,

Također, što se tiče OO dizajna, pokušavala sam iskoristiti Don't Repeat Yourself gdje god sam uspjela, naprimjer nasljeđivanjem Professora i Studenta iz klase User. Također, napravljen je interfejs da bi se izbjeglo ponavljanje koda budući da je metoda generatePassword prilično duga. Keep It Simple, Stupid je svakako praćen, budući da sam sve pokušavala uraditi na što jednostavniji način i sa što manje klasa. Također, kroz izradu vođeno je računa o SOLID principima.

* 1. Grafički korisnički interfejs

Korišteni Layouts-i : Hbox, FlowPane, BorderPane, Vbox, GridPane..

Korisnički interfejsi koji imaju određene funkcionalnosti, a ne samo prelaz ili izbor određenih opcija su responsive. Za potrebe izgleda, korišten je i CSS za ikone, validacije, prelazak mišom, određeni klik i slično. Što se tiče glavnih prozora, korišten je BorderPane. Organizacija unutar njega je napravljena korištenjem ostalih gore navedenih panela. Nepotrebne prozore sam se trudila uvijek zatvoriti, međutim budući da nekada nema smisla da se neki prozori u pozadini zatvore, to je jedini razlog zašto su glavni prozori ostali uglavnom tu. Što se tiče Gestalt principa: Skrenuta je pažnja da uglavnom dugmad koja upravljaju određenim segmentom se nalaze jako blizu i izgledaju kao grupa, naprimjer (upravljanje profesorima, studentima, predmetima, materijalima – sve moguće funkcionalnosti nalaze se u jednoj grupi). Kroz čitavu aplikaciju skrenuta je pažnja na boje, veličine, ikone i slično, da sve izgleda grupisano po funkcionalnostima, povezano, a u suštini kao jedna cjelina. Također, kod dosta interfejsa korištena je figura/pozadina. Također, Alert dijalozi su često korišteni, čime je održana konzistentnost. Korištena je navigacija tastaturom gdje god se za to ukazala prilika, naprimjer enter umjesto default dugmadi, kretanje kroz polja korišenjem tipke Tab, također korištene su i određene kratice. Korišten je meni, budući da zbog same aplikacije nisam imala pretjeran broj mogućnosti koje sam mogla staviti tamo, korišten je na više mjesta (naprimjer kreiranje dokumenta – tu je jako koristan). Pored kratica, što se tiče značenja stavki i ikonica, korištena je konzistentnost što je više moguće (budući da nisam uspjela pronaći jednu temu koja sadrži sve ikone, koristeći se IconFlat, pokušala sam pronaći sve ikone da izgledaju kao dio jedne cjeline). Također pored menija, jako dosta je korišten i Toolbar (za logout, back – zbog ljepšeg izgleda su stavljeni tu, kod kreiranja dokumenta i slično). Validacije su prisutne kod prijave na aplikaciju, promjene šifre, dodavanja u bazu podataka i slično. Što se tiče pomoći korisniku, korišteni su about dijalozi koji pružaju osnovne informacije o aplikaciji, zatim prilikom kreiranja vlastitog materijala, korištena je statusna traka, te također korišten je Tooltip (prilikom prijave budući da zbog (meni) ljepšeg izgleda, polja za unos su prilično „stopljena“ sa pozadinom, zatim za objašnjenje većine funkcionalnosti kroz aplikaciju).

* 1. Organizacija koda kroz resurse

U programu postoji folder označen za resurse u kojem se nalaze poddirektoriji za slike, izvještaje, css, fxml i jezik. Također, aplikacija sadrži i folder materials zbog lakšeg snalaženja u kodu.

* 1. Rad sa datotekama

Zbog same prirode aplikacije, rad sa datotekama je jedna od glavnih funkcionalnosti. Postoji i pisanje i čitanje iz datoteka, a posebna funkcionalnost koja je uvedena zbog čitanja iz datoteke, jeste prikaz kviza. Naime, kada profesor otvori da vidi sadržaj kviza ima mogućnost da pročita i tačan odgovor, dok student vidi samo pitanja i ponuđene odgovore.

* 1. Mrežno programiranje

Mrežno programiranje sam odlučila iskoristiti kod izbora slike prilikom promjene profila od strane profesora ili studenta koja se također čuva u bazi. Budući da sam kod zadaće imala grešku, odlučila sam da bi bila dobra ideja da ovo pokušam uraditi na ispravan način, te zbog toga sam ponovo koristila Giphy.

* 1. Rad sa bazom podataka

Korištena je SQLite baza podataka, te kreirane su četiri tabele. Što se tiče baze, bitno je da spomenem MaterialManagementDAO klasu u kojoj se vrši uspostavljanje konekcije sa bazom, u kojoj su definisani svi pripremljeni upiti, te razne metode koje obrađuju podatke iz baze i nude ih aplikaciji na korištenje. Time je postignuto odvajanje baze podataka od ostatka programa, što je omogućilo veći prostor za nadogradnju baze bez pretjeranih komplikacija za ostatak programa.

U nastavku je prikazan ER dijagram:

* 1. Višenitno, malo vidjet pa dodat
  2. Izvještavanje

Za kreiranje izvještaja korišten je Jaspersoft Studio, te uključena je biblioteka net.sf.jasperreports:jasperreports:6.14.0. Izvještaji u aplikaciji su iskorišteni kod samog administratora koji pomoću njih ima mogućnost pregleda svih profesora, studenata i predmeta koji se nalaze u bazi.

* 1. Internacionalizacija

Aplikacija je dostupna na engleskom i bosanskom jeziku. Promjena jezika je moguća u meni-ju na glavnom prozoru aplikacije.

* 1. Alati za automatsku – dat mu i jar još

Koristeći okruženje kreirana je JAR datoteka.