UNIVERZITET U SARAJEVU

ELEKTROTEHNIČKI FAKULTET, SARAJEVO

Seminarski rad iz predmeta:

Razvoj programskih rješenja

Tema:

Aplikacija za evidenciju ocjena studenata na fakultetu (e-Index)

Profesor:

Doc. dr Vedran Ljubović, dipl.ing.el.

Student:

Faris Poljčić (18120)

Sarajevo, februar 2019. godine

Sadržaj

1. Uvod 1
2. Klase 2
2.1 JavaBeans klase 2
2.2 Controller klase 2
2.3 Singleton klase 8
2.4 Wrapper klase 9
2.5 Dodatne klase 9
3. Resursi 11
4. Baza 12
5. Zaključak 14

1. Uvod

Cilj ovog projekta bila je samostalna izrada aplikacije koja rješava konkretan korisnički problem. U ovom slučaju to je bio problem evidencije ocjena studenata na fakultetu.

Aplikacija je pisana u programskom jeziku Java (verzija 10) uz pomoć IntelliJ IDEA razvojnog okruženja. Korištena baza podataka je Oracle dodijeljena od strane fakulteta. Za kreiranje entiteta baze korišten je Oracle SQL Developer. Pored datih alata korišten je Jaspersoft Studio za kreiranje izvještaja, Adobe Photoshop za uređivanje slika i Sublime Text za pregled pojedinih datoteka.

Osnovne mogućnosti ove aplikacije su vođenje osnovnih podataka svakog studenta i profesora, evidencija trenutnog stanja bodova studenta i upisanih ocjena, te mogućnost promjene bodova i upisa ocjene. Pored ovih funkcionalnosti dodato je mnoštvo dodatnih mogućnosti opisanih u poglavljima koje slijede. Izvorni kod aplikacije je prilagođen engleskom jeziku, a sama aplikacija je na bosanskom jeziku.

Program se sastoji od 32 klase, te 14 različitih formi koje pružaju sve potrebne osnovne funkcionalnosti nekom fakultetu. Aplikacija razlikuje 3 vrste korisnika:

- Administrator
- Profesor
- Student

Baza podataka se sastoji od 10 različitih entiteta, međusobno povezanih u jednu cjelinu koristeći 1:1, te 1:N veze. Entiteti korišteni za izradu ovog projekta su:

- Login
- Osoba
- Administrator
- Profesor
- Student
- Kurs
- Semestar
- Ocjena
- Predmet
- Plan i program

U prilogu se nalazi dijagram šeme baze podataka i dijagram klasa.

2. Klase

Sve klase ovog programa možemo podijeliti na JavaBeans (podatkovne) klase, Controller klase, Singleton klase, Wrapper klase i dodatne klase.

2.1 JavaBeans klase

JavaBeans (zrna kafe) klase su čisto podatkovne klase koje nemaju ništa osim atributa i settera/gettera. Ove klase predstavljaju entitete baze podataka. Svaka od njih sadrži konstruktor bez parametara, konstruktor sa svim atributima klase i settere/gettere za svaki atribut. Atributi su privatni, dok su metode javne. Klase, zajedno sa njihovim atributima su:

- *Login* (id, username, password, dateCreated, userType, lastLoginDate)
- *Person* (id, firstName, lastName, jmbg, address, email, login (referenca na *Login* klasu))
- *Administrator* – nasljeđuje klasu *Person*
- *Professor* (title) – nasljeđuje klasu *Person*
- *Student* (birthDate, semester (referenca na klasu *Semester*), course (referenca na klasu *Course*), pauseDate) – nasljeđuje klasu *Person*
- *Semester* (id, no, cycleNo, ects)
- *Coruse* (id, name)
- *Subject* (id, name, code, ects, professor (referenca na klasu *Professor*), reqSubject (referenca na klasu *Subject*))
- *Grade* (id, student (referenca na klasu *Student*), subject (referenca na klasu *Subject*), points, score, gradeDate, professor (referenca na klasu *Professor*))
- *Curriculum* (id, course (referenca na klasu *Course*), semester (referenca na klasu *Semester*), subject (referenca na klasu *Subject*), requiredSubject)

Sve ove klase preklapaju *equals* metodu na način da dva objekta smatramo jednakim ako imaju jednak id. U pojedinim klasama je također preklopljena metoda *toString* zbog potrebe ispisa klasa u kontrolama kao što su *ChoiceBox*. Metoda je preklopljena tako da vraća atribut naziv pojedine klase. Npr. Za klasu *Subject* ova metoda vraća naziv predmeta.

2.2 Controller klase

Controller klase predstavljaju logiku koja se nalazi u pozadini svake forme i predviđena je da sadrži handlere za događaje. Program se sastoji od 14 controller klasa, jedna klasa za svaku formu. U nastavku slijedi pregled svake controller klase:

- *AboutController*

Ovo je najjednostavnija controller klasa i ona se nalazi u pozadini *about.fxml* forme. Ovo je forma koja prikazuje osnovne podatke o autoru programa. Pri inicijalizaciji samo se postavlja slika *ImageView* kontroli. Ima samo jedno dugme kojim se zatvara.

- *AddCourseController*

Ova klasa omogućava dodavanje i ažuriranje postojećih kurseva. Vezana je za *addCourse.fxml* formu. Konstruktor prima jedan parametar i to je referenca na klasu *Course*. U slučaju da je ova

referenca *null* to označava da se vrši dodavanje novog kursa, u suprotnom se ažurira postojeći. Pri inicijalizaciji se dodaje slušač na polje naziva kursa u svrhu provjere validnosti naziva. Naziv kursa je validan ako je njegova dužina između 3 i 60 karaktera. U slučaju da je referenca na kurs različita od *null*, polje naziva se postavlja na odgovarajući naziv postojećeg kursa. U slučaju greške pri pristupu bazi prikazuje se *Alert*, tipa *error*. Atribut *okClicked* služi kao flag koji nam govori da li je pritisnuto Ok dugme i da li je pristup bazi prošao bez problema. Vizualni indikator koji obavještava korisnika o ispravnosti polja se vrši preko procedure *addColor*. Ova procedura prima objekat tipa *TextField* i bool vrijednost. Prvi parametar predstavlja polje nad kojim se vrši validacija, a drugi parametar je vrijednost koja označava da li je polje validno ili ne. U slučaju da je polje prazno, koristeći css se dodaje atribut *blankField*, dok se ostali uklanjaju. Ako nije prazno provjerava se drugi parametar. U slučaju da je on true, kroz css se dodaje atribut *validField*, a uklanjaju ostali. Ako je drugi parametar postavljen na false, dodaje se atribut *invalidField* i uklanjaju ostali.

- *AddCurriculumController*

Ova klasa omogućava izmjenu plana i programa fakulteta. Svaka stavka *Curriculum* tabele definiše za koji kurs i u kojem semestru se pojedini predmet nalazi. Također definiše da li je predmet obavezan za dati semestar. Forma koja je vezana za ovu klasu je *addCurriculum.fxml*. Konstruktor prima referencu na klasu *Curriculum*, te *ObservableList* predmeta, semestara i kurseva. Pri inicijalizaciji se popunjavaju odgovarajuće *ChoiceBox* kontrole sa podacima predmeta, semestara i kurseva. Ovo popunjavanje se vrši u zasebnoj niti radi povećavanja efikasnosti. Na ovoj formi nema validacije jer se unos svih podataka vrši kroz *ChoiceBox* kontrole i *CheckBox*. Na osnovu reference na klasu *Curriculum* (prvog parametra konstruktora) se razlikuje dodavanje nove stavke i ažuriranje postojeće. Atribut *okClicked* ima istu ulogu kao i u klasi *AddCourseController*.

- *AddSubjectController*

AddSubjectController omogućava dodavanje i izmjenu postojećih predmeta. Forma za koju je vezana je *addSubject.fxml*. Konstruktor prima referencu na klasu *Subject, ObservableList* profesora i predmeta. Ovi parametri služe za razlikovanje dodavanja i ažuriranja predmeta, te za popunjavanje odgovarajućih *ChoiceBox* kontrola. Inicijalizacija počinje popunjavanjem *ChoiceBox* konotrola sa profesorima i predmetima u zasebnoj niti. *ChoiceBox* predmeta služi za definisanje uslovnog predmeta. Za ovu potrebu u datu listu predmeta je dodat i jedan imaginarni predmet, sa postavljenim nazivom „Bez uslovnog predmeta“, sa očiglednim značenjem. Definisana su i dva slušača na polje naziva predmeta i njegove šifre u svrhu validacije. Naziv predmeta je validan ako je njegova dužina između 3 i 50 karaktera. Šifra je validna ako nije kraća od 2 karaktera, niti duža od 19 karaktera. Nakon što korisnik klikne na Ok dugme provjerava se i da li je naziv predmeta jednak nazivu uslovnog predmeta. Tj. Da li se predmet i njegov uslovni poklapaju. Atribut *okClicked* ima isto značenje kao i kod ostalih klasa. Forma *addSubject.fxml* sadrži i kontrolu *Spinner* za potrebe definisanja broja ECTS kredita. Vrijednosti *Spinnera* se također definišu prilikom inicijalizacije. Omogućeno je i korištenje strelica za promjenu vrijednosti *Spinnera*, koristeći proceduru *spinnerStep* koja se aktivira nakon pritiskanja tastera na tastaturi. U zavisnosti od koda pritisnutog tastera vrši se inkrementiranje ili dekrementiranje vrijednosti kontrole.

- *AddPersonController*

Ova klasa je pozadina *addPerson.fxml* forme. Omogućava dodavanje, te ažuriranje postojećih korisnika aplikacije. Data forma se sastoji od tri dijela, a to su: osnovne informacije, login informacije i dodatne informacije. Osnovne informacije predstavljaju ime, prezime, jmbg, adresu i email. Login informacije čine korisničko ime, šifra i tip korisnika. Dodatne informacije su datum rođenja, semestar i smjer za studenta, te titula za profesora. Svi tipovi korisnika imaju samo mogućnost promjene informacija koje su vezane za njih. Tako, studenti nemaju pristup polju titule, dok profesori nemaju pristup poljima datuma rođenja, semestra i smjera. Sama klasa prima četiri parametra: referencu na klasu *Person*, tekst koji predstavlja tip korisnika, *ObservableList* semestara i smjerova. Drugi parametar, tj. Tip korisnika je neophodan jer referenca na klasu *Person* može biti *null* (što označava dodavanje novog korisnika), te se ne može odrediti tip koristeći *instanceof*. Prilikom inicijalizacije se prvo onemogućava odgovarajući dio forme u zavisnosti od tipa korisnika. Nakon toga se postavljaju slušači i *ChoiceBox* kontrole se pune podacima. Zbog velikog broja polja validacija se ne vrši prilikom kucanja već nakon pritiska na Ok dugme. Dva prisutna slušača služe za automatsko postavljanje korisničkog imena koristeći prvo slovo imena u kombinaciji sa prezimenom prilikom dodavanja korisnika. Dato korisničko ime je samo prijedlog i može se promijeniti. U slučaju da je reference na klasu *Person* različita od *null*, sva polja forme se postavljaju na osnovu podataka osobe. Prilikom ažuriranja se također onemogućava polje korisničkog imena, te polja semestra i kursa u slučaju studenta. Atribut *okClicked* i vizualna indikacija validnosti polja je ista kao i do sada. Validacija polja se vrši kroz četiri funkcije. Prva funkcija provjerava validnost osnovnih informacija. Provjerava se dužina imena, prezimena i adrese. Zatim postojanje numeričkih karaktera u imenu i prezimenu, validnost jmbg i maila. Također se provjerava postojanje duplog jmbg u bazi. Druga funkcija provjerava login informacije. Provjerava se dužina korisničkog imena, šifre, te jedinstvenost korisničkog imena u bazi. Treća funkcija provjerava informacije studenta, tj. Validnost njegovog datuma rođenja. Provjerava se da li je postavljen na datum u budućnosti i da li se poklapa sa jmbg. Četvrta funkcija provjerava profesorove informacije, tj. Validnost titule. Dodavanje i ažuriranje se vrši pozivom odgovarajućih funkcija iz *BazaDAO* klase. U slučaju da se dodaje novi student njemu se automatski dodjeljuju predmeti prvog semestra njegovog smjera. Prilikom dodavanja nove osobe potrebno je vršiti naknadnu izmjenu polja šifre. Šifra se prilikom spašavanja u bazu kodira koristeći MD5 hash. Međutim, dato kodiranje se vrši kombinovanjem korisničkog imena, šifre i id-a korisnika. Iz ovog razloga se prvo u bazu dodaje korisnik sa ostalim login informacijama. Nakon toga se pročita njegov id iz baze i na osnovu njega, korisničkog imena i šifre generiše kodirana šifra koja se upisuje u bazu.

- *AdministratorController*

Ovaj controller upravlja *administrator.fxml* formom. Ovo je forma koja se prikazuje nakon što se administrator prijavi. Data forma sadrži šest tabela: predmeti, profesori, studenti, smjerovi, plan i program i administratori. Svaka od ovih tabela se nalazi na zasebnom *Tab*-u. Za svaku je omogućeno dodavanje, ažuriranje i brisanje stavki. *Tab*-ovi sa tabelama predmeta, profesora, studenata i smjerova također sadrže i različite statističke podatke o odabranim stavkama. Forma

također sadrži i meni sa četiri različita dijela. *File* meni omogućava odjavljivanje, zatvaranje programa, te prelazak na sljedeću akademsku godinu. *Edit* meni omogućava isto što i dugmad na odgovarajućem *Tab*-u. *View* meni omogućava sakrivanje/prikazivanje tabela. *Help* meni omogućava prikaz *about.fxml* forme. *AdministratorController* klasa je jedna od najobilnijih klasa ovog programa. Konstruktor prima jedan parametar koji predstavlja referencu na *Login* klasu i sadrži podatke o login informacijama prijavljenog administratora. Inicijalizacija ove klase počinje postavljenjem slušača na polja *view* meni-a. Na početku svi *Tab*-ovi su onemogućeni i tabele su prazne. Zatim se učitavaju neophodni podaci (predmeti, semestri, smjerovi i profesori). Učitavanje ovih podataka omogućava efikasno popunjavanje *ChoiceBox* kontrola pojedinih formi, te ih je zbog toga neophodno učitati. Nakon toga se čita datoteka *config.dat*, koja čuva posljednje spašenu konfiguraciju *view* meni-a i na osnovu nje se omogućavaju odgovarajući *Tab*-ovi i prikazuju tabele. Data forma sadrži i tri *ChoiceBox* kontrole koje se popunjavaju podacima o ciklusima, semestrima i smjerovima fakulteta. U slučaju bilo kakve greške pri inicijalizaciji administrator se automatski odjavljuje i forma se zatvara. Zadnji dio inicijalizacije vrši postavljanje slike za dugme osvježavanja i postavljanje slušača. Slušači omogućavaju ažuriranje statističkih podataka nakon selektovanja stavke neke tabele. Svako ažuriranje se vrši u zasebnoj niti da ne bi opteretilo glavni program. Za svaki predmet se prikazuje ukupan broj studenata, prosječna ocjena, broj studenata koji su položili predmet i koji nisu, te prolaznost u procentima. Za svakog profesora se također prikazuju isti podaci. Za studente se prikazuje njihov prosjek, broj (ne)položenih predmeta. Prosječna ocjena i broj studenata na svakom smjeru se također evidentira. Na *Tab*-u studenata omogućeno je i filtriranje prikazanih studenata po smjeru, ciklusu i semestru koristeći *ChoiceBox* kontrole. Ažuriranje tabela nakon dodavanja, ažuriranja ili brisanja stavki se vrši samo ako korisnik pritisne Ok dugme i ako se data akcija izvrši nad bazom. Ovo se provjerava koristeći *okClicked* atribut. Dato osvježavanje se vrši u zasebnoj niti. Odjavljivanje sa forme podrazumijeva raskidanje veze sa bazom, zatvaranje forme i prikazivanje *login.fxml* forme. Prelazak na sljedeću akademsku godinu podrazumijeva postavljanje bodova iz neupisanih predmeta na 10 svakom studentu.

- *StudentController*

Ova klasa je pozadina *student.fxml* forme. Ovo je forma koja se prikazuje nakon prijavljivanja studenta. Forma sadrži dvije tabele: trenutno aktivne predmete i upisane predmete. Student ima mogućnost ažuriranja svojih informacija, mogućnost printanja spiska upisanih ocjena, spašavanja u vidu pdf, docx ili xlsx datoteke, te može zalediti godinu. Sve date opcije su omogućene i kroz odgovarajući meni. *View* meni također omogućava sakrivanje tabele sa položenim predmetima. Sakrivanje datih podataka skraćuje vrijeme potrebno da se student prijavi. Ažuriranje informacija se vrši koristeći *addPerson.fxml* formu. Konstruktor ove klase prima jedan parametar koji predstavlja referencu na *Login* klasu koja sadrži login informacije studenta. U inicijalizaciji se prvo dohvaća odgovarajući student na osnovu login reference. Zatim se postavlja slušač na *View* meni, te učitavaju svi neophodni podaci. Iz *config.dat* binarne datoteke se učitava spašena postavka *view* meni-a, tj. Da li se prikazuju položeni predmeti. Nakon toga se učitava statistika, te postavlja status. Status je „Aktivan“ ako student nije zaledio godinu, „Zaleđen“ ako je zaledio godinu i „Neaktivan“ ako je prošla godina dana od kako je student zaledio svoje stanje. U slučaju da student nema status „Aktivan“ onemogućeno mu je

ažuriranje svojih informacija. Prilikom inicijalizacije se također provjerava da li je tabela aktivnih predmeta prazna. U slučaju da jeste i student ima status „Aktivan“ prikazuje se *advanceStudent.fxml* forma. Tj. Studentu je omogućen prelazak na sljedeći semestar. Printanje izvještaja položenih ocjena se sastoji u kreiranju nove instance klase *Report* i pozivanje odgovarajuće metode. Spašavanje datog izvještaja se vrši koristeći *FileChooser*, te nakon odabira tipa (pdf, docx, xlsx) i lokacije kreira se nova instanca klase *Report* i poziva metoda *saveStudentReport*.

- *ProfessorController*

Ovo je controller *professor.fxml* forme. Analogno *student.fxml* formi, ovo je forma koja se prikazuje nakon prijavljivanja profesora. Forma sadrži dvije tabele: spisak predmeta koje profesor predaje i spisak studenata na datim predmetima. Profesor ima mogućnost ažuriranja svojih informacija, te mogućnost upisa bodova i ocjene studentima. Date opcije su moguće i kroz meni. Ažuriranje informacija se vrši koristeći *addPerson.fxml* formu, dok promjena bodova i upis ocjene preko *gradeStudent.fxml* forme. Konstruktor ove klase prima referencu na *Login* klasu koja sadrži login informacije profesora. Prilikom inicijalizacije se prvo preuzima odgovarajući profesor koristeći login informacije. Zatim se tabele pune podacima, postavlja slušač i dohvaćaju osnovne informacije o datom profesoru. Slušač je postavljen na tabelu studenata, te omogućava praćenje selektovanog studenta. Nakon selektovanja studenta, profesor može izmijeniti njegov broj bodova i eventualno upisati ocjenu. Ovo se vrši pozivom *gradeStudent.fxml* forme i *GradeStudentController* klase. U slučaju da se izvrši izmjena, u zasebnoj niti se ažurira tabela studenata.

- *AdvanceStudentController*

Ova klasa se poziva ako je student zadovoljio uslove za prelazak na sljedeći semestar. Forma za koju je vezana je *advanceStudent.fxml*. Ova forma obavještava studenta da je stekao uslov za prelazak na sljedeći semestar i omogućava mu izbor izbornih predmeta. Koristi dvije *ListView* kontrole da prikaže izborne i konačne predmete. Student koristi dva tastera da prebaci izborni predmet u konačne ili da, nakon dodavanja, vrati predmet iz liste konačnih u izborne. Konstruktor prima referencu na studenta i listu njegovih položenih predmeta. Ova lista je neophodna zbog provjere uslovnih predmeta. Inicijalizacija prvo popunjava dvije liste sa izbornim i obaveznim predmetima. Nakon toga se postavljaju dva slušača koji služe da zapamte selektovani izborni/konačni predmet. Na kraju se postavlja tekst glavne labele i postavljaju ikone za dva tastera. U slučaju problema sa konektovanjem na bazu, student se odjavljuje i forma zatvara. Nakon što student izabere izborne predmete može pritisnuti na Ok dugme pri čemu se poziva *okClick* procedura. U njoj se prvo provjerava ukupan broj ects bodova. Naime, svaki semestar ima minimalni broj ects bodova koji je potrebno imati za upis, taj broj se upoređuje sa zbirom ects bodova konačnih predmeta. Zatim se provjerava da li je student položio sve uslovne predmete za odabrane predmete. U slučaju da jeste traži se potvrda za odabrane konačne predmete i ako student potvrdi, baza se ažurira i student prelazi na sljedeći semestar. Student se upisuje na odabrane predmete i za svaki predmet se broj bodova postavlja na 10.

- *GradeStudentController*

Ovaj controller omogućava ažuriranje bodova studenta i upis ocjene. Klasa je vezana za *gradeStudent.fxml* formu. Forma sadrži dvije *Spinner* kontrole, te dvije *Radio Button* kontrole. Dvije *Spinner* kontrole predstavljaju broj bodova i ocjenu. *Radio Button* kontrole označavaju da li se vrši upis ocjene ili samo ažuriranje bodova. Konstruktor klase ima tri parametra: referencu na klasu *Student*, *Grade* i *Professor*. Prilikom inicijalizacije se prvo kreira nova *ToggleGroup* koja osigurava da će samo jedan *Radio Button* biti aktivan u svakom trenutku. Zatim se dodaje slušač na *Spinner* bodova. Ovaj slušač automatski određuje ocjenu na osnovu datih bodova, onemogućava selektovanje *Radio Button*-a za upis ocjene ako je broj bodova manji od 55, te daje vizualni indikator o validnosti broja bodova. Broj bodova se ne smije smanjiti od trenutnog broja i ne smije biti veći od 110. Koristeći *text formatter* omogućeno je unos isključivo decimalnih vrijednosti za broj bodova. Na kraju inicijalizacije se postavljaju *Spinner* kontrole. Atribut *okClicked* i vizualni indikator validnosti polja su implementirani kao i kod ostalih klasa. Prilikom upisa ocjene traži se potvrda i u slučaju greške sa bazom ne vrši se nikakva izmjena.

- *LoginController*

Prva forma koja se prikazuje nakon pokretanja programa je *login.fxml*. Ova klasa se veže za tu formu. Login forma omogućava unos korisničkog imena, šifre, te izbor tipa korisnika nakon čega se korisnik prijavljuje. Tip korisnika se odabira selektovanjem odgovarajućeg *Radio Button*-a. Data forma također sadrži i jedno skriveno dugme čije će značenje biti uskoro objašnjeno. Konstruktor ove klase nema parametara, u njemu se samo inicijalizuje baza, *ToggleGroup* i tri *property*-a. Username i password *property* su tipa *SimpleStringProperty* i oni se vežu sa odgovarajućim poljima forme. *ErrorCount property* broji neuspjele pokušaje prijavljivanja i tipa je *SimpleIntegerProperty*. *ToggleGroup* služi da omogući da samo jedan *RadioButton* bude aktivan u datom trenutku. Prilikom inicijalizacije se svaki *RadioButton* povezuje sa datom *ToggleGroup*-om, dodaje se slušač na *errorCount property* i vrši se dvosmjerno povezivanje username i password *property*-a. Slušač na *errorCount* otkriva skriveno dugme u slučaju da je njegova vrijednost veća ili jednaka dva. Tj. U slučaju da korisnik dva puta unese ispravno korisničko ime i odabere ispravan tip korisnika, ali pogriješi šifru. Nakon što korisnik unese korisničko ime, šifru i odabere tip korisnika može pritisnuti na dugme za prijavu. Pored ovog načina može i koristiti enter taster. U oba slučaja poziva se procedura *loginClick*. Ona prvo provjerava da li je korisnik unio korisničko ime i šifru. Zatim pristupa bazi i provjerava tabelu logina, u slučaju da pronađe korisnika sa istim korisničkim imenom i istog tipa kao selektovani *RadioButton* preuzima njegove informacije. Zatim kodira unesenu šifru i upoređuje je sa onom u bazi. U slučaju da se kodirana šifra ne poklapa izbacuje grešku. U suprotnom se ažurira zadnji datum prijave datog logina, trenutna forma se zatvara i prikazuje nova u ovisnosti od selektovanog *RadioButton*-a. U ovisnosti od tipa korisnika se specificira forma i njena controller klasa. Za studenta i administratora se također definišu i odgovarajuće procedure koje se pozivaju nakon njihovog zatvaranja. Nakon zatvaranja *admin.fxml* forme poziva se *writeAdminView* procedura, a nakon zatvaranja *student.fxml* forme poziva se *writeStudentView* procedura. Ove procedure spašavaju postavku odabranih tabela koje korisnik selektuje da se prikazuju prilikom otvaranja forme (kroz *view* meni). Skriveno dugme koje se prikazuje nakon što *errorCount* dosegne vrijednost dva služi da pomogne korisniku u slučaju

da je zaboravio šifru. Ako korisnik pritisne ovo dugme otvara se nova forma *passRecovery.fxml* koju korisnik može koristiti da pokuša resetovati šifru. Ova klasa također sadrži i tri statičke metode *formPasswordId*, *getEncodedPassword* i *encodePassword*. Prva kombinuje id i šifru i vraća string koji se, zajedno sa korisničkim imenom, koristi za formiranje kodirane šifre. Druga vraća kodiranu šifru koja se upisuje u bazu podataka. Formira se na osnovu tri parametra funkcije: korisničkog imena, šifre i id-a. Zadnja statička metoda *encodePassword* prima string, kodira ga koristeći MD5 heširanje i vraća rezultat.

- *PassRecoveryController*, *PassCodeController* i *PassNewController*

Ove tri klase služe za omogućavanje resetovanja šifre u slučaju da je korisnik zaboravi. Pozadina su *passRecovery.fxml*, *passCode.fxml* i *passNew.fxml* formi, respektivno. Nakon što korisnik pritisne dugme za resetovanje šifre prikazuje se *passRecovery.fxml* forma. Ova forma zahtijeva od korisnika unos imena, prezimena, email-a i odabir tipa korisnika. Nakon što korisnik unese potrebne podatke i odabere tip, može pritisnuti na Ok dugme, pri čemu se poziva *recoverClick* metoda. Ova metoda se nalazi u *PassRecoveryController* klasi. Ona prvo provjerava ispravnost unesenih podataka, te pristupa bazi i provjerava da li postoji korisnik sa datim podacima. U slučaju da postoji, generiše se nasumični kod od 6 cifara koji se zatim šalje na uneseni mail. Mail se šalje sa gmail-a eindexfp@gmail.com koristeći klasu *Email*. Ovo se vrši u pozadini, koristeći zasebnu nit, što omogućava dalji nastavak programa bez zadržavanja. Program dalje prikazuje *passCode.fxml* formu iza koje se nalazi *PassCodeController* klasa. Ova klasa sadrži konstruktor sa jednim parametrom koji predstavlja originalno poslani kod. Data forma obavještava korisnika o poslatom mail-u, te sastoji se od jednog polja u koje korisnik treba unijeti poslani kontrolni kod. Korisnik ima pravo tri puta da pogriješi kod nakon čega se ova i *passRecovery.fxml* forma zatvaraju. U slučaju da korisnik unese kontrolni kod koji se poklapa sa poslatim kodom, zatvara se *passCode.fxml* forma i otvara *passNew.fxml* forma. Preko ove forme korisnik može unijeti novu šifru koja se zatim kodira i ažurira u bazi. Pozadina ove forme je *PassNewController* klasa. Konstruktor ove klase prima jedan parametar i on predstavlja id logina koji se spasi u *PassRecoveryController* klasi.

2.3 Singleton klase

Singleton klase su klase koje mogu imati samo jednu instancu, a konstruktor je zabranjen. Prilikom rada sa bazom podataka ovakav tip klase je pogodan jer izbjegava da imamo više različitih pogleda na podatke koji mogu dovesti do konflikta. Klasa koja se koristi isključivo za pristup bazi podataka u ovom programu je *BazaDAO*. Ona je realizovana kao singleton na način da kao privatni statički atribut ima referencu na *BazaDAO*, konstruktor joj je privatan, i sadrži javne metode za dohvaćanje i uklanjanje date reference. Dohvaćanje reference se vrši pozivom metode *getInstance* koja provjerava da li je atribut instance jednak *null*, u slučaju da jeste inicijalizuje se, te se data instanca vraća. Uklanjanje instance se vrši pozivom metode removeInstance koja postavlja atribut instance na *null*. Inicijalizacija ove klase se sastoji u konektovanju na bazu i pripremanja 64 upita za izvršavanje. Preko datih upita je omogućeno dodavanje, brisanje, ažuriranje i sve ostale potrebne funkcionalnosti nad bazom. Sve metode koriste intuitivne nazive, tj. Lahko je zaključiti njihovu funkciju na osnovu naziva. Ova klasa ne obrađuje izuzetke, njihova obrada je zadatak pozivaoca. Metode koje vraćaju JavaBeans

klasu, npr. Neki objekat na osnovu njegovog id-a, vraćaju *null* ako dati objekat ne postoji u bazi.

2.4 Wrapper klase

Wrapper klasa je klasa koja kombinuje više objekata u jednu cjelinu. Potreba za ovakvim klasama se ukazala nakon popunjavanja tabela *student.fxml* i *professor.fxml* formi. Tabele u ovim formama zahtijevaju podatke iz više entiteta baze. Iz ovog razloga neophodne su bile klase koje objedinjuju više entiteta baze u jednu. Ovaj program sadrži tri wrapper klase: *SubjectWrapper, SubjectGradeWrapper* i *StudentGradeSubjectWrapper*.

SubjectWrapper klasa proširuje klasu *Subject* sa tri nova atributa: *avgGrade, noStudentsGraded, noStudentsNotGraded*. Ova klasa se koristi za popunjavanje tabele predmeta na formi *professor.fxml*. Dodatna tri atributa opisuju prosječnu ocjenu, broj studenata koji su položili predmet i broj onih koji nisu. Također, ova klasa sadrži dva gettera koja vraćaju izvedene logičke atribute: *noStudents* i *percentPassed*. Prvi predstavlja broj studenata na predmetu, računa se kao zbir onih koji su položili i onih koji nisu. Drugi se također računa na osnovu datih podataka i vraća procenat studenata koji su položili predmet, tj. Računa prolaznost.

SubjectGradeWrapper klasa se koristi za popunjavanje aktivnih i položenih predmeta *student.fxml* forme. Ona objedinjuje atribute klase *Subject* i *Grade*. Pomoću ove klase je, pored informacija o samom predmetu, omogućen i prikaz podataka o ocjeni i bodovima za svaki predmet.

StudentGradeSubjectWrapper klasa objedinjuje atribute *Student* i *Grade* klase, te čuva i atribut o imenu predmeta. Ova klasa se koristi za tabelu studenata u *professor.fxml* formi. Pomoću nje je moguće objediniti informacije o studentu (ime, prezime, jmbg, adresa, email, datum rođenja, semestar, smjer, zaledio), ocjeni (bodovi) i predmetu (naziv) u jednu klasu.

2.5 Dodatne klase

U ovu grupu klasa možemo svrstat preostale klase:

- *Main*

Ovo je klasa koja služi za pokretanje programa, ona samo sadrži startni kod koji otvara *login.fxml* formu.

- *Email*

Ova klasa je namijenjena za slanje mail-a koristeći java kod. Njeni privatni atributi su *userName* (**korisnicko_ime**@gmail.com), *password* (šifra za dati korisnički račun), *recipients* (lista mail adresa na koje se šalje mail), *subject* (naslov mail-a), *body* (tekst mail-a). Sadrži konstruktor bez parametara i sa svim parametrima, kao i settere za date atribute. Čitanje je zabranjeno. Slanje se vrši pozivom procedure *send*. U svrhu ljepšeg formatiranja poruke, poruke se formatiraju kao html dokumenti.

- *Report*

Report klasa služi da omogući prikaz i spašavanje izvještaja o ocjenama. Sadrži dvije javne metode: *showStudentReport* i *saveStudentReport*. Prva vrši prikaz izvještaja koristeći *JRViewer*. Koristeći dati preglednik moguće je isprintati izvještaj. Druga metoda spašava dati izvještaj na računar u pdf, docx ili xlsx formi. *JasperReport* omogućava korištenje parametara u izvještaju čije se vrijednosti naknadno postavljaju. Ovaj izvještaj koristi tri parametra: *reportsDirPath* (putanja izvještaja, koristi se za potrebu određivanja putanja slika), *name* (ime i prezime studenta) i *organizationName* (ime organizacije, u ovom slučaju „Elektrotehnički fakultet, Sarajevo“).

- *WrongEmailException*

Ova klasa nasljeđuje *Exception* klasu i predstavlja izuzetak koji se baca iz *Email* klase u slučaju pogrešnih parametara prilikom slanja mail-a.

3. Resursi

Radi veće preglednosti i lakše organizacije svi resursi koje ovaj program koristi organizovani su pod *resources* direktorijom. Ovaj direktorij sadrži dvije datoteke i četiri dodatna direktorija.

Prva datoteka je *ETF-ORA.sql* i predstavlja izvršnu SQL skriptu koju je potrebno izvršiti za kreiranje baze. Dati kod je moguće urediti po vlastitim potrebama, a u postavljenoj verziji se sastoji iz četiri koraka. U prvom koraku se prvo brišu sve prisutne tabele u datoj šemi. Drugi korak podrazumijeva kreiranje svih tabela potrebnih za rad programa. Treći korak kreira sekvence koje služe za automatsko inkrementiranje id-eva pojedinih tabela. Zadnji korak popunjava tabele sa relevantnim podacima koji su u ovom slučaju korišteni za testiranje. U realnoj izvedbi aplikacije neophodni bi bili drugi i treći korak, te barem jedan administrator za prijavu. Također neophodno bi bilo definisati semestre fakulteta. Koristeći *administrator.fxml* formu moguće je dalje definisati studente, profesore, plan i program i predmete.

Druga datoteka je *config.dat*. Ovo je binarna datoteka koja čuva informacije o željenim tabelama za prikaz pri pokretanju *administrator.fxml* i *student.fxml* formi. Prvi bajt je vezan za *student.fxml* formu, dok su ostali za *administrator.fxml* formu.

Četiri dodatna direktorija su:

- css

Ovaj direktorij sadrži sve .css datoteke. U ovom slučaju samo jednu, a to je *design.css*. Ova datoteka sadrži sav kod za uređivanje vizualnog izgleda aplikacije.

- fxml

Fxml direktorij sadrži svih 14 formi ovog programa. Naziv fxml datoteka je izabran na način da asocira na njihovu ulogu.

- img

Sve slike koje se koriste u programu (osim onih u izvještaju) su smještene u ovaj direktorij. Ukupno se sastoji od 17 ikona (preuzetih sa web stranice https://icons8.com) i slike koja se koristi na *about.fxml* formi.

- reports

U ovom direktoriju se nalazi pripremljen izvještaj koji se koristi od strane studenata. Pored izvještaja tu su i dvije slike: natpis i ikona ETF-a.

4. Baza

Baza podataka koja se nalazi u pozadini ovog programa je kreirana uz pomoć Oracle SQL Developer-a. Prilikom izrade baze obraćena je pažnja na memorijski utrošak tipova podataka kao i na ograničenja (Primary key, Unique, Not null). Sve tabele, zajedno sa atributima, tipovima i ograničenjima navodim ispod:

- *Login*
  - id integer primary key
  - username varchar(50) not null unique
  - password varchar(100) not null
  - date_created date
  - user_type varchar(30)
  - last_login_date date
- *Person*
  - id integer primary key
  - first_name varchar(50) not null
  - last_name varchar(55) not null
  - jmbg varchar(30) not null unique
  - address varchar(60)
  - email varchar(100)
  - login_id integer references *login* (id)
- *Semester*
  - id integer primary key
  - no integer not null
  - cycle_no integer not null
  - ects integer not null
- *Course*
  - id integer primary key
  - name varchar(60) not null
- *Student*
  - id integer primary key references *person* (id) on delete cascade
  - birth_date date not null
  - semester_id integer references *semester* (id)
  - course_id integer references *course* (id)
  - pause_date date
- *Professor*
  - id integer primary key references *person* (id) on delete cascade
  - title varchar(60) not null
- *Administrator*
  - id integer primary key references *person* (id) on delete cascade

- *Subject*
    - id integer primary key
    - name varchar(50) not null
    - code varchar(20) not null
    - ects integer not null
    - professor_id integer references *professor* (id)
    - req_subject_id integer references *subject* (id)
- *Grade*
    - id integer primary key
    - student_id integer references *student* (id)
    - subject_id integer references *subject* (id)
    - points float not null
    - score integer
    - grade_date date
    - professor_id integer references *professor* (id)
- *Curriculum*
    - id integer primary key
    - course_id integer references *course* (id)
    - semester_id integer references *semester* (id)
    - subject_id integer references *subject* (id)
    - required_subject char(2) not null

5. Zaključak

Ovo je bio jako izazovan projekat čija samostalna izrada je trajala nekih 20 dana. Radeći projekat sam shvatio suštinu ove rečenice: „Softver nije proizvod i on se ne proizvodi. Softver se gradi, to je proces“ i shvatio potrebu za timskim radom prilikom razvoja softvera.

Ovaj program se može dalje nadograđivati raznim funkcionalnostima i opcijama. Mislim da trenutno sadrži sve osnovne potrebne funkcionalnosti za vođenje ocjena studenata na fakultetu. Pored osnovnih sadrži i neke opcije koje čisto pokazuju potencijale ovakvog programa. Kao neke dodatne funkcionalnosti mogu spomenuti serijalizaciju u xml datoteke, prevođenje aplikacije na druge jezike, kreiranje Fat JAR datoteke, evidencija pojedinih ispita, vođenje više informacija o svakom entitetu, više statističkih podataka itd.

Ukupno ovaj program sadrži 4891 linija java koda (5556 računajući prazne redove), css datoteka 110 linija (139 računajući prazne redove), a sql skripta 243 linije.