Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.9
DOI: https://doi.org/10.24867/07BE05Ilic

APLIKACIJA ZA UPRAVLJANJE RAZMENOM STUDENATA PUTEM ERASMUS+ PROGRAMA

STUDENT EXCHANGE APPLICATION THROUGH ERASMUS+ PROGRAMME

Jelena Ilić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – SOFTVERSKO INŽENJERSTVO I INFORMACIONE TEHNOLOGIJE

Kratak sadržaj – *U radu je objašnjen proces prijavljivanja studenata za program mobilnosti. Dat je model sistema kao i objašnjenje dela njegove implementacije. Prikazan je korisnički interfejs sistema.*

Ključne reči: *Erasmus+ program, razmena studenata, RESTful veb servisi, pozadinsko izvršavanje zadataka*

Abstract – *This paper explains the process of student application for a mobility program. The implementation and model of the system are given. User interface of the system is shown.*

Keywords: *Erasmus+ programme, student exchange, RESTful web services, background task execution*

1. UVOD

Erasmus+ [1] program je program Evropske unije koji podržava obrazovanje, obuku i mobilnost mladih širom Evrope. Povećava mogućnosti zapošljavanja kao i modernizacije obrazovnih sistema. Ovim putem se otvara mogućnost putovanja u edukativne svrhe za učenike, studente, nastavno osoblje i omladinske radnike.

Erasmus+ program nudi finansijsku podršku organizacijama i institucijama koje rade u oblasti obrazovanja. Program traje 7 godina (od 2014. do 2020. godine) i njegov budžet od 14,7 milijardi evra pruža mogućnosti za preko 4 miliona Evropljana da studiraju, obučavaju se i stiču nova iskustva u inostranstvu.

U radu će biti posmatrana procedura podnošenja zahteva studenta na Fakultetu tehničkih nauka u Novom Sadu za studiranje u inostranstvu. Cilj rada jeste automatizacija celog procesa kako bi se studentu olakšalo prijavljivanje na Erasmus+ program, tako i organizaciji fakulteta odobravanje i uvid u sve zahteve studenata koji žele da dobiju priliku za studiranje u inostranstvu. Takođe u radu će biti opisane korišćene tehnologije za implementaciju rešenja ali i objašnjena sama implementacija određenih delova sistema.

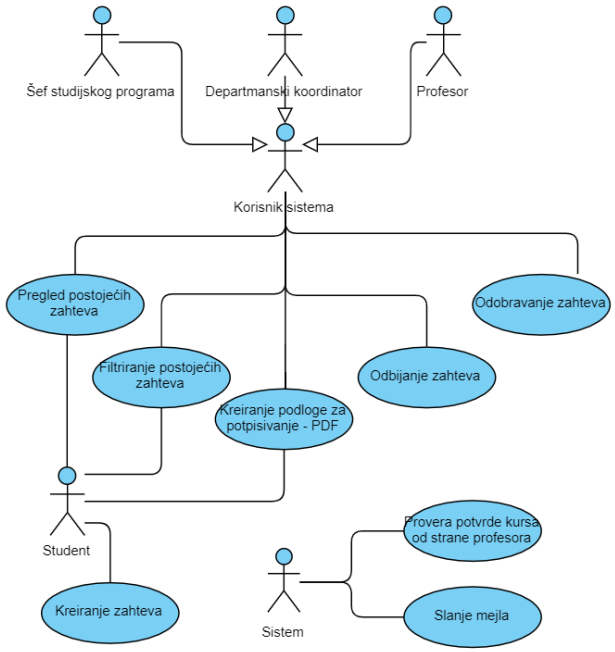
2. SPECIFIKACIJA ZAHTEVA SISTEMA

Ova aplikacija treba da podrži funkcionalnosti vezane za čitav proces koji student treba da prođe prilikom konkurisanja na program razmene.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio prof. dr Goran Savić.

2.1. Slučajevi korišćenja

Pomoću *dijagrama slučajeva korišćenja* moguće je prikazati skup akcija (slučajeva korišćenja) koje korisnici (akteri) sistema mogu da izvrše (slika 2.1).



Slika 2.1 *Dijagram slučajeva korišćenja*

2.2. Uloga sistema u okviru aplikacije

S obzirom da sistem ima posebnu ulogu u aplikaciji, funkcionalnosti koje može da obavi su slanje e-mail poruka i izvršavanje određenih pozadinskih zadataka.

2.2.1 E-mail podrška

Po odobravanju predmeta od strane koordinatora sistem automatski šalje e-mail poruke svim profesorima čije je predmete student naveo u svojoj prijavi.

2.2.1 Izvršavanje zadataka u pozadini

Automatska saglasnost profesora nakon određenog vremenskog perioda obezbeđena je uz pomoć pozadinskih zadataka. To znači da se vrši provera potrvrde profesora da se njegov predmet zameni na stranom fakultetu.

3. TEHNOLOGIJE

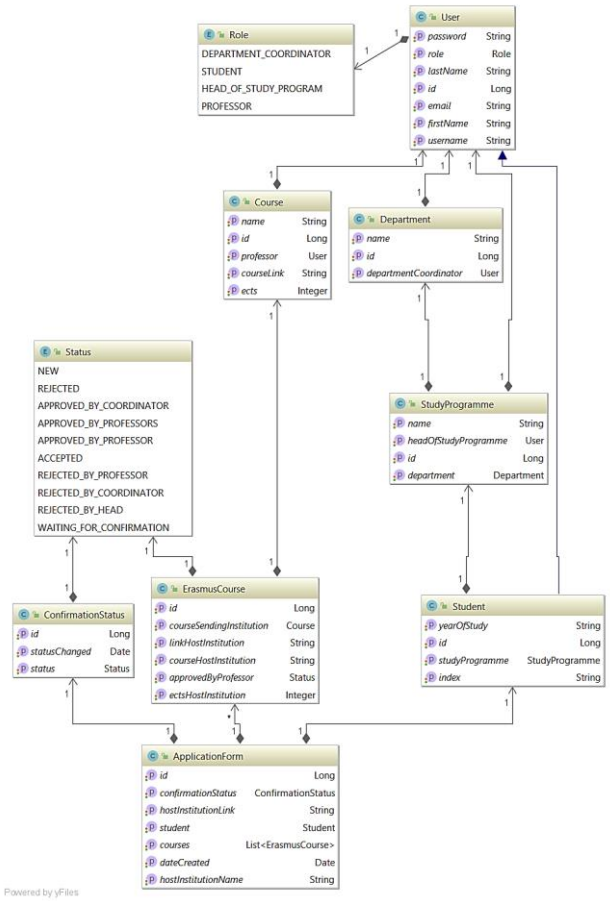
Aplikacija je napisana u programskom jeziku *Java* [2]. Za implementaciju serverske strane aplikacije korišćen je *Spring Boot*, koji olakšava korišćenje *Spring* [3] radnog okvira, a izgrađena je pomoću alata *Apache Maven* [4]. Na klijentskoj strani korišćen je *Angular* [5]. Skladištenje podataka omogućeno je putem *MySQL* [6] baze podataka.

4. MODEL SISTEMA

Kada se govori o modelu čitavog sistema potrebno je sagledati sve aspekte. Počinje se od modela podataka. S druge strane aplikacija prolazi kroz različite procedure i potrebno je obratiti pažnju na model poslovnog procesa.

4.1 Model podataka

Model podataka koji se preslikava na prethodno pomenutu bazu podataka prikazan je na slici 4.1 u okviru dijagrama klasa. U priloženom dijagramu prikazana su kako svojstva svakog entiteta tako i relacije među njima.

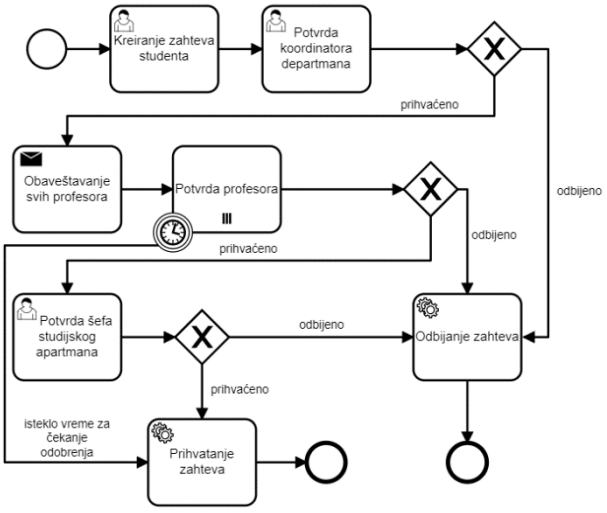


Slika 4.1 *Dijagram klasa osnovnog modela*

Kreiranje novog zahteva studenta opisano je klasom *ApplicationForm*. S obzirom da student prilikom prijavljivanja na program razmene treba da odabere predmete, ovaj entitet ima vezu *1:N* sa entitetom *ErasmusCourse*. U okviru njega definisani su svi potrebni podaci koje student treba da popuni za jedan predmet. Entitet *Course* opisuje postojeće predmete na Fakultetu tehničkih nauka sa podacima o broju bodova, profesoru i linku gde je opis predmeta dostupan. Uloge korisnika sistema modelovane su enumeracijom *Role* sa vrednostima *Student, Professor, Department-Coordinator, Head-Of-Study-Program.*

4.2 Model poslovnog procesa

Model poslovnog procesa predstavlja bitan deo sistema u kome se smenjuju unapred predviđeni događaji, odnosno aktivnosti. Ovakav model prikazuje redosled događaja tokom obavljanja funkcionalnosti aplikacije, odnosno sekvencijalne i konkurente korake u jednom poslovnom procesu. (slika 4.1).



Slika 4.1 *Model poslovnog procesa*

5. IMPLEMENTACIJA

Kada se govori o funkcionisanju veb aplikacija potrebno je pomenuti osnovne koncepte koji omogućavaju komunikaciju. Drugi deo se odnosi na implementaciju određenih delova aplikacije.

5.1 RESTful web servisi

Serverska strana aplikacije implementirana je pomoću Spring radnog okvira, a zasniva se na REST (Representational State Transfer) protokolu [7]. REST definiše skup principa za razvoj veb usluga. Implementacija RESTful web servisa predstavljena je sa 4 osnovna principa: eksplicitno korišćenje HTTP metoda, stateless komunikacija, identifikacija resursa pomoću URI mehanizma i transfer podataka putem XML ili JSON objekata.

5.2 Implementacija serverskog dela aplikacija

Pre svega, potrebno je opisati tačku sistema u kojoj se vrši komunikacija sa bazom podataka. Pored toga, svaka aplikacija zahteva određenu obradu podataka. Implementacija funkcionalnosti predstavlja poslovnu logiku aplikacije. Zadatak serverskog dela aplikacije je i da omogući dostupnost javnog API-ja preko koga se omogućava komunikacija klijenata sa serverom.

5.3 Implementacija klijentskog dela aplikacija

Klijentska aplikacija se sastoji od komponenti TypeScript klasa predstavlja aplikativnu logiku. Ima pridružen *template* fajl (html) koji definiše izgled komponente. Klasa komunicira sa prikazom kroz javni API odnosno preko definisanih atributa i metoda. Servis klijentske aplikaciju predstavlja tačku gde se vrši komunikacija sa serverskim delom aplikacije.

5.4 Autentifikacija pomoću JWT tokena

JWT (JSON Web Token) [9] predstavlja način autentifikacije korisnika u sistemu. Princip funkcionisanja ovakve autentifikacije je da svaki korisnik po uspešnoj prijavi na sistem dobija jedinstveno izgenerisan token od strane servera. Taj token korisnik dalje koristi u radu sa aplikacijom, tako što ga prosleđuje u svakom sledećem zahtevu. Time je korisniku omogućen pristup rutama i resursima. U okviru klijentske aplikacije omogućeno je presretanje zahteva i dodavanje tokena u svaki zahtev ka serveru. Na serverskoj strani se vrši provera postojanja tokena pri svakom zahtevu kao i njegove validnosti.

5.5 Servis za slanje e-mail poruka

Slanje e-mail poruka često predstavlja blokirajuću tačku u sistemu. S obzirom da student može da odabere proizvoljan broj predmeta, dolazi se do problema slanja velikog broja poruka u istom trenutku. Čekanje na završetak ove operacije nije dobro rešenje. Iz tog razloga korišćeni su asinhroni zadaci.

Anotacija *@Async* iznad metode za slanje poruka označava da će se metoda izvršavati u posebnoj niti. Asinhrono izvršavanje operacija potrebno je omogućiti i pomoću anotacije *@EnableAsync* u main klasi aplikacije.

```
@Async("threadPoolTaskExecutor")
public void sendConfirmationEmail(String to, String
subject, String text) {
 try {
  SimpleMailMessage message = new
SimpleMailMessage();
   message.setTo(to);
  message.setSubject(subject);
  message.setText(text);
  emailSender.send(message);
 } catch (MailException exception) {
  throw new
BadRequestException(exception.getMessage());
 }
}
```

Listing 5.3 *Implementacija slanja e-mail poruka*

5.6 Čuvanje istorije statusa zahteva studenata

Istorija promene statusa predstavlja kolekciju svake promene statusa zahteva. Za svaku promenu čuva se status u koji je promenjen kao i trenutno vreme promene.

5.7 Implementacija pozadinskih zadataka

Trenutak potvrde koordinatora beleži se u bazi podataka u okviru istorije promene statusa. Potrebno je obezbediti mehanizam provere koliko je vremena prošlo od tog trenutka i obezbediti automatsku potvrdu od strane profesora nakon isteka određenog vremena.

Zakazivanje zadataka implementirano je uz pomoć *@Scheduled* anotacije (listing 5.4).

```
@Scheduled(cron = "0 * * ? * *")
@Transactional
public void checkProfessorsApproval(){
 List<ApplicationForm> applicationForms =
applicationFormRepository.
findByConfirmationStatusStatus(Status.
APPROVED_BY_COORDINATOR);

 for (ApplicationForm applicationForm : a) {
 long diff = new Date().getTime() -
applicationForm.getConfirmationStatus()
           .getStatusChanged().getTime();
 int diffDays = (int) (diff / DAY_IN_MS);
 if (diffDays >= WAITING_PERIOD) {
  ConfirmationStatus confirmationStatus =
confirmationStatusRepository
.getOne(a.getConfirmationStatus().getId());
confirmatonStatus.setStatusChanged(newDate())
confirmationStatus.setStatus(
Status.APPROVED_BY_PROFESSORS);
a.setConfirmationStatus(confirmationStatus);
confirmationStatusRepository.save(confirmationStatus
);
 }
}
```

Listing 5.4 *Implementacija pozadinskog zadatka*

Takođe, da bi se omogućilo izvršavanje zadataka potrebno je dodati anotaciju *@EnableScheduling* u okviru main klase aplikacije. Ovom konfiguracijom, pri pokretanju aplikacije startuju se sve metode koje su ovako anotirane.

Za definiciju vremenskog perioda kada će se vršiti provera korišćen je *cron* izraz. Format ovakvih izraza je: *sekunde, minuti, sati, dan u mesecu, mesec, dan u nedelji, godina*. U primeru je definisano da se zadatak, odnosno provera izvršava svakog dana u ponoć.

5.8 Generisanje podloge za potpisivanje

Podloga za potpisivanje predstavlja dokument koji služi za priznavanje perioda mobilnosti studenta. Svi odobreni predmeti se nalaze u ovom dokumentu, a sam dokument mora biti potpisan od strane koordinatora departmana i šefa studijskog programa. Svi korisnici sistema mogu da preuzmu ovu podlogu u PDF formatu.

S obzirom da se radi sa promenljivim skupom podataka potreban je mehanizam za dinamičko generisanje dokumenta u trenutku zahteva korisnika. Na serverskoj strani korišćena je *itext* biblioteka. Preuzimanje dokumenta na klijentskoj strani obezbeđeno pomoću biblioteke *files-saver* koja omogućava preuzimanje dokumenata u okviru veb aplikacija.

6. PRIKAZ SISTEMA

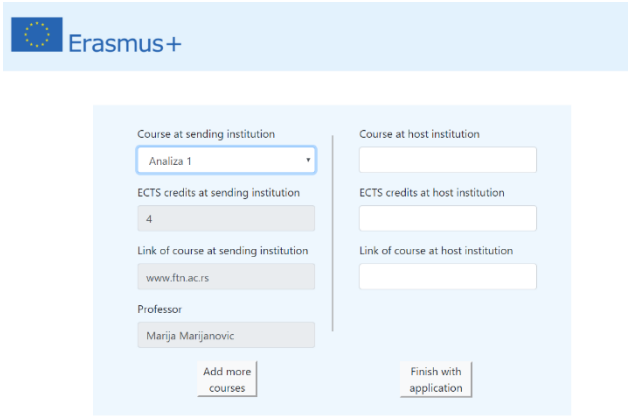
Pokretanje klijentske aplikacije vrši se iz browsera na računaru korisnika sistema.

6.1 Prijava na sistem i pregled funkcionalnosti

Rad sa aplikacijom korisnik započinje prijavljivanjem na sistem unošenjem korisničkog imena i lozinke. Na osnovu uloge korisnika otvara se početna stranica sa glavnim prikazom funkcionalnosti koje korisnik može da obavi.

6.2 Formiranje novog zahteva studenta

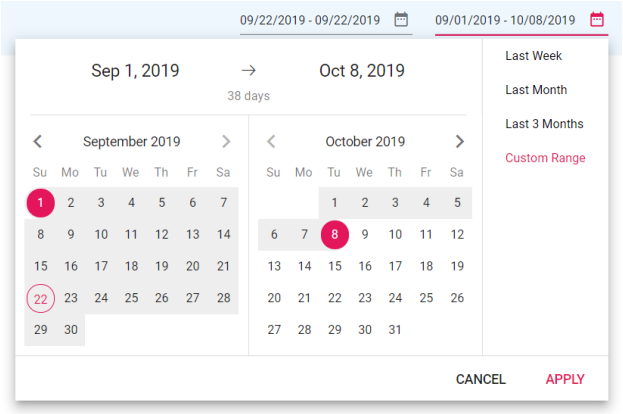
Osnovni podaci o studentu se sami popunjavaju, a zatim se unose podaci o stranoj instituciji na koju student konkuriše. Student može da dodaje proizvoljan broj kurseva koje želi da sluša tokom razmene (slika 6.1).



Slika 6.1 *Dodavanje kurseva koje student želi da sluša*

6.3 Pregled i filtriranje postojećih zahteva

Pregled istorije zahteva dostupan je svim korisnicima. Funkcionalnost sortiranja omogućena je po svim poljima u rastućem i opadajućem redosledu. Filtriranje podataka je po atributima *application created*, *status changed* i *status*. Filtriranje po datumu omogućeno je odabirom vremenskog intervala (slika 6.2).



Slika 6.2 *Prikaz odabira intervala za filtriranje po datumu*

6.4 Generisanje podloge za potpisivanje

Poslednji korak u procesu konkurisanja studenta na studije u inostranstvu jeste potpisivanje podloge za akademsko priznavanje perioda mobilnosti (slika 6.3).

Student: Jelena Ilic (index: SW42-2014)
Study programme: Softversko inzenjerstvo
Year of study: 4
Host university: University of Jaen

ERASMUS COURSES:

	Course at sending institution	ECTS credits to be recognised by the Sending Institution	Course at host institution	ECTS credits at the Host Institution
1	Masinsko ucenje	6	Machine learning	4
2	Analiza 1	4	Math	4

Head of study programme:

(Djordje Popovic)

Coordinator:

(Milan Petrovic)

Slika 6.3 *Izgled podloge za potpisivanje*

7. ZAKLJUČAK

Kako bi se olakšao osnovni proces konkurisanja na program mobilnosti u radu je opisana procedura podnošenja zahteva na Fakultetu tehničkih nauka u Novom Sadu, odnosno podrška za uvođenje automatizacije procesa podnošenja zahteva za studije u inostranstvu.

Opisan je način funkcionisanja Erasmus+ programa. Zatim je bilo reči o zahtevima sistema iz ugla korisnika. Navedene su korišćene tehnologije a kasnije i implementacija određenih delova sistema. U radu je dat opis modela podataka sistema. Takođe opisana je i podrška za vođenje procedure zahteva studenata putem modela poslovnog procesa. U poslednjem poglavlju, čitalac stiče sliku o celokupnom izgledu aplikacije iz ugla posmatranja korisnika.

Kako trenutna implementacija sistema podržava postupak kreiranja zahteva studenata samo na Fakultetu tehničkih nauka, dalje unapređenje aplikacije bilo bi proširenje sistema na nivo univerziteta. Proširivanjem sistema na veći broj fakulteta znatno bi povećao broj korisnika koji pristupaju sistemu. Ovo proširenje pre svega bi dovelo do znatnog povećanja broja poslatih e-mail poruka. Iz ovog razloga, trebalo bi razmisliti o korišćenju eksternih servisa koji pružaju API za razmenu e-pošte.

8. LITERATURA

[1] Ptak A, “Business -university cooperation in Europe” The educational program for 2014-2020 - key actions providing business cooperation with higher education institutions, Czestochowa University of Technology, 2012

[2] Ken Arnold, James Gosling, David Holmes, “The Java programming language”, Addison Wesley Professional, 17. avgust 2005.

[3] Pivotal „Spring“, Dostupno na: https://spring.io/, pristupljeno: 15. septembar 2019.

[4] The Apache Software Foundation “Apache Maven”, dostupno na: https://maven.apache.org, pristupljeno: 16. septembar 2019.

[5] „Angular”, dostupno na: https://angular.io/docs, pristupljeno: 16. septembar 2019.

[6] TechTarget “MySQL”, dostupno na: https://searchoracle.techtarget.com/definition/MySQL, pristupljeno: 16. septembar 2019.

[7] Alex Rodriguez, “RESTful Web services: The basics”, IBM, 6. novembar 2008.

[8] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee, “Hypertext Transfer Protocol -- HTTP/1.1” - Introduction, jun 1999.

[9] M. B. Jones , B. Campbell , Ping Identity , C. Mortimore, “JSON Web Token (JWT) Profile for OAuth 2.0 Client Authentication and Authorization Grants, 29. mart 2013.

Kratka biografija:

Jelena Ilić rođena je 21. decembra 1995. godine u Novom Sadu, gde je 2010. godine završila osnovnu školu „Prva vojvođanska brigada“. Iste godine upisuje srednju ekonomsku školu „Svetozar Miletić“ u Novom Sadu, smer Ekonomski tehničar, koju završava 2014. godine. Iz osnovne i srednje škole izlazi kao nosilac Vukove diplome. Fakultet tehničkih nauka upisuje 2014. godine, smer Softversko inženjerstvo i informacione tehnologije. Polaže sve ispite predviđene planom i programom, nakon čega pristupa izradi diplomskog rada na temu „Generisanje alarma u sistemu za upravljanje logovima“. Zvanje diplomirani inženjer elektrotehnike i računarstva stiče 04.09.2018. godine sa prosečnom ocenom 9.61. Iste godine upisuje master akademske studije na smeru Softversko inženjerstvo i informacione tehnologije, odsek Elektronsko poslovanje. Polaže sve ispite master studija sa prosečnom ocenom 9.86.