Aplikacija za mentoriranje studentskih seminara

Prakljačić, Stjepan

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA

Sveučilišni studij

APLIKACIJA ZA MENTORIRANJE STUDENTSKIH SEMINARA

Završni rad

Stjepan Prakljačić

Osijek, 2021.

SADRŽAJ

1. UVOD 1

1.1. Zadatak završnog rada 1

2. OPIS KORIŠTENJA TEHNOLOGIJA 2

2.1. Struktura Oracle baze podataka 2

2.2. Oracle APEX razvojno okruženje 3

2.3. PL/SQL programski jezik 3

2.4. Postojeća programska rješenja 4

3. MODELIRANJE BAZE PODATAKA 6

3.1. Analiza zahtjeva 7

3.2. Projektiranje baze podataka 7

3.3. Stvaranje modela 8

3.4. Fizičko oblikovanje baze podataka 15

4. STVARANJE APLIKACIJE U ORACLE APEX OKRUŽENJU 21

4.1. Izgled aplikacije 26

5. ZAKLJUČAK 31

LITERATURA 32

SAŽETAK 34

ABSTRACT 35

POPIS KRATICA 36

POPIS SLIKA 37

POPIS TABLICA 38

ŽIVOTOPIS 39

1. UVOD

Aplikacija za mentoriranje studentskih seminara je web aplikacija koja objedinjuje studente, mentore, kolegije i odabrane teme seminara. Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek, trenutno koristi nepraktična i neučinkovita rješenja vezana za pitanje odabira, mentoriranja i ocjenjivanja studentskih seminara. Ta rješenja se često nalaze u obliku tablica i upisnih lista. Veliki nedostatak takvog načina mentoriranja studentskih seminara jest činjenica da se vrlo često događaju greške prilikom upisa i ispisa pojedinih studenata na pojedine, željene teme. Osim toga takav način upisa studenata na željenu temu ne nudi mogućnost uvida u ocjenu niti nudi mogućnost olakšane komunikacije, odnosno komuniciranja s mentorom. Gotovo sva komunikacija se odvija putem službene elektroničke pošte što često rezultira mnogim drugim mogućim problemima. Osim toga unos tema je vrlo nepraktičan kao i brisanje te održavanje istih. Takav način unosa ne nudi olakšanu kontrolu mentoru nad samim temama pa tako niti nad studentima. Cilj završnog rada je izrada aplikacije koja otklanja navedene nedostatke i pruža učinkovita rješenja koje su praktična i korisna. Bit će opisan proces modeliranja te korištenja baze podataka kao i postupak izrade aplikacije u Oracle APEX razvojnom okruženju..

1.1. Zadatak završnog rada

Zadatak završnog rada je izrada aplikacije koja omogućuje mentoru kreiranje kolegija, unos tema seminarskih radova te broj studenata za svaku pojedinu grupu. Aplikacija također omogućuje studentu upis na pojedinu temu te pohranu izrađenog seminara na ocjenjivanje. Aplikacija nudi kontrolu mentoru za svaku pojedinu temu kao i za teme u cjelini, unos ocjene te kontrolu nad studentima.

2. OPIS KORIŠTENJA TEHNOLOGIJA

U ovom radu upotrijebljene su efikasne i pouzdane tehnologije. Primijenjena su znanja prethodno stečena o programskim jezicima kao i korištenje SQL-a (engl. *Structured Query Language*) koji se pokazao kao vrlo stabilan, učinkovit i fleksibilan.

2.1. Struktura Oracle baze podataka

Oracle je dizajniran kako bi bio lako prenosiva baza podataka - dostupna je na svim relevantnim platformama, od Windows-a do UNIX-a. Međutim, fizička arhitektura Oracle-a izgleda drugačije na različitim operacijskim sustavima.

Dva izraza koja prilikom korištenja u Oracleovom kontekstu, često izazivaju pogreške su: „Baza podataka“ i „instanca“. U Oracle terminologiji, definicije ovih pojmova su sljedeće:

- Baza podataka: Zbirka fizičkih datoteka ili diskova operacijskog sustava. Prilikom korištenja ASM-a (engl. *Oracle Automatic Storage Management*) ili RAW particije (dijela fizičkog diska kojem se pristupa na najnižoj mogućoj razini), možda se neće prikazivati kao pojedinačne, zasebne datoteke u operacijskom sustavu, ali definicija ostaje ista.
- Instanca: Skup Oracle pozadinskih procesa ili niti i zajednička memorija koja se dijeli između tih niti ili procesa koje izvodi jedno računalo. Instanca baze podataka može postojati bez ikakve memorije na disku.

Ta se dva pojma ponekad koriste naizmjenično, ali obuhvaćaju vrlo različite koncepte. Odnos između njih je da se baza podataka može otvoriti u mnogim instancama. A instanca može otvoriti samo jednu bazu podataka u bilo kojem trenutku. Zapravo instanca će otvoriti najviše jednu bazu podataka tijekom cijelog svog života[1]!

Oracle baza podataka je kolekcija podataka. Svrha baze podataka je pohranjivanje i preuzimanje povezanih podataka. Poslužitelj baze podataka je ključ za rješavanje problema upravljanja informacijama. Općenito, poslužitelj pouzdano upravlja velikim količinama podataka u višekorisničkom okruženju tako da mnogi korisnici mogu istodobno pristupiti istim podacima. Sve se to postiže pružajući visoke performanse. Baza podataka također sprječava neovlašteni pristup i pruža učinkovita rješenja za oporavak u slučaju kvara[2].

2.2. Oracle APEX razvojno okruženje

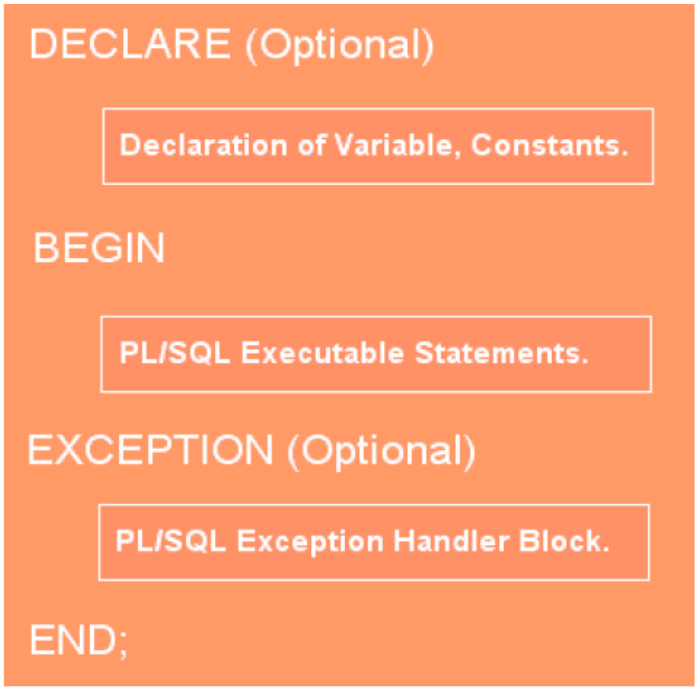
Oracle APEX (engl. *Oracle Application Express*) je nisko-kodna razvojna platforma koja pruža mogućnost izrade jednostavnih, stabilnih te učinkovitih aplikacija koje se mogu implementirati na više različitih načina. Vrlo su sigurne i koriste bazu podataka koja pruža brz pristup podacima, vrhunske performanse i skalabilnost. Oracle APEX koristi jednostavnu troslojnu arhitekturu gdje se zahtjevi iz preglednika, putem web poslužitelja, šalju u bazu podataka. Obrada i manipulacija podacima, izvršavaju se unutar baze podataka. Oracle APEX jedinstven je po tome što nudi fleksibilnost implementacije. Može se implementirati na Oracle oblaku, privatnom oblaku ili na bilo kojem drugom mjestu gdje se Oracle baza podataka pokreće[3].

2.3. PL/SQL programski jezik

PL/SQL, Oracle proceduralno proširenje SQL-a, je jezik visokih performansi za obradu transakcija. PL/SQL kombinira moć upravljanja podacima SQL-a s proceduralnim jezicima. Kao i drugi proceduralni programski jezici, PL/SQL omogućuje deklariranje konstanti i varijabli, kontroliranje tijeka programa, definiranje potprograma i obradu pogreške u izvođenju. Složeni problemi se mogu rastaviti na lako razumljive potprograme, koji se mogu ponovno koristiti u više aplikacija[4]. PL/SQL je blok-strukturirani jezik podijeljen i zapisan u tri logička bloka. Blokovi „BEGIN“ i „END“ su obvezni, a druga dva bloka „DECLARE“ i „EXCEPTION“ neobavezni su blokovi. Blok „END“ se ne smatra blokom, već samo ključnom riječi na kraju PL/SQL programa. Struktura blokova vidljiva je na slici 2.1.

Svaki od ovih blokova ima sljedeće uloge:

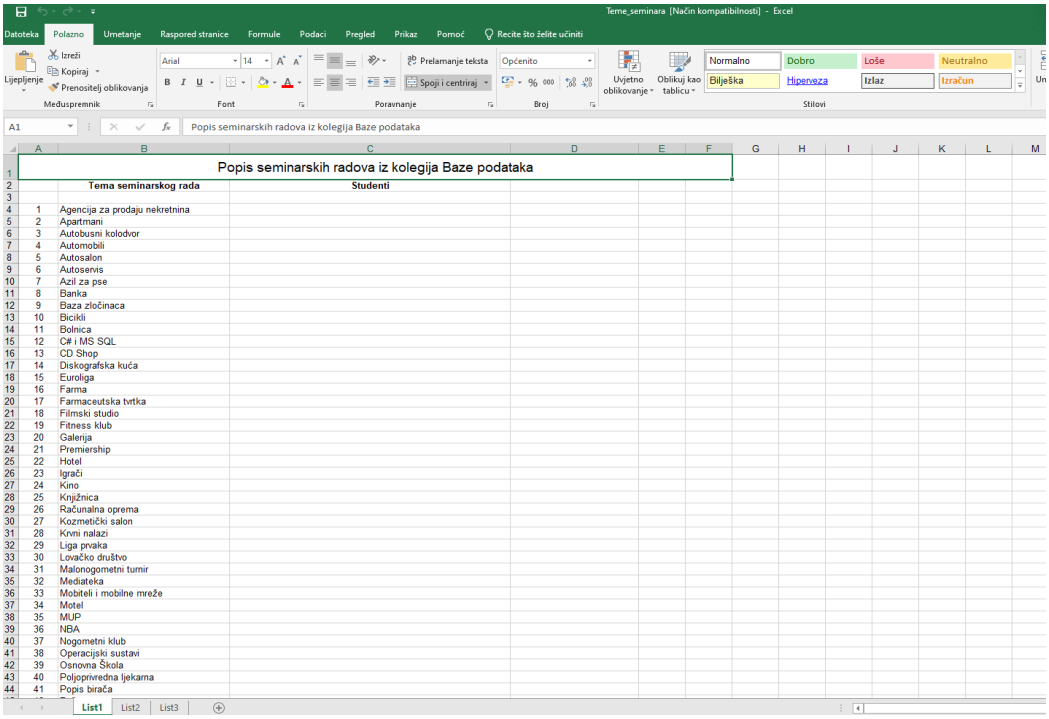
- Declare - varijable i konstante deklariraju se i inicijaliziraju unutar ovog bloka. Obavezna je deklaracija varijabli i konstanti prije nego što ih se referencira u proceduralnom iskazu;
- Begin - blok proceduralnih izjava odgovoran za implementaciju stvarne programske logike;
- Exception – u ovom bloku se mogu prijaviti definirani ili unaprijed definirani uvjeti pogreške. PL/ SQL, vrlo učinkovito nudi način za rješavanje pogrešaka. Pogreške mogu nastati zbog pogrešne sintakse, loše logike ili neprimjenjivanja pravila provjere valjanosti[5].



Slika 2.1 Struktura blokova PL/SQL, izvor: [5]

2.4. Postojeća programska rješenja

Proučavanjem postojećih aplikacija, odnosno programskih rješenja, uočljivo je kako ne postoje aplikacije koje sadrže potrebne funkcionalnosti, ne nude potrebna rješenja za problematiku koju obrađuje ovaj rad. Sustav koji se trenutno koristi za mentoriranja seminara ne zadovoljava potrebe studenata niti mentora koji žele veću preglednost, učinkovitost i funkcionalnost. Trenutni oblik omogućava upis studenata na željenu temu, ali isto tako moguće su pogreške poput: slučajnog ispisa, krivog unosa i slično. Slika 2.2, prikazuje dosadašnja rješenja.

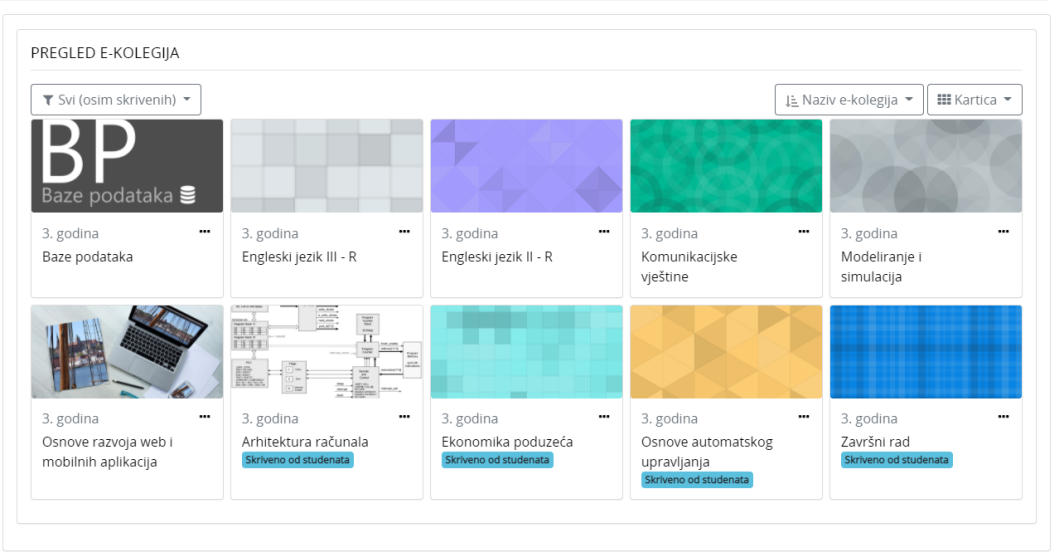


Popis seminarskih radova iz kolegija Baze podataka

	Tema seminarskog rada	Studenti
1	Agencija za prodaju nekretnina	
2	Apartmani	
3	Autobusni kolodvor	
4	Automobili	
5	Autosalon	
6	Autosenis	
7	Azil za pse	
8	Banka	
9	Baza zločinaca	
10	Bicikli	
11	Bolnica	
12	C# i MS SQL	
13	CD Shop	
14	Diskografska kuća	
15	Euroliga	
16	Farma	
17	Farmaceutska tvrtka	
18	Filmski studio	
19	Fitness klub	
20	Galerija	
21	Premiership	
22	Hotel	
23	Igrači	
24	Kino	
25	Knjižnica	
26	Računalna oprema	
27	Kozmetički salon	
28	Krvni nalazi	
29	Liga prvaka	
30	Lovačko društvo	
31	Malonogometni turnir	
32	Mediateka	
33	Mobiteli i mobilne mreže	
34	Motel	
35	MUP	
36	NBA	
37	Nogometni klub	
38	Operacijski sustavi	
39	Osnovna Škola	
40	Poljoprivredna ljekarna	
41	Popis birača	

Slika 2.2 Prikaz odabira teme seminara na kolegiju Baze podataka

Aplikacija sa sličnim funkcionalnostima je Merlin sustav za e-učenje koja nudi pregled kolegija, upisanih studenata i seminarskih tema, ali ne nudi mogućnost upisa na odabrane teme kao niti mogućnost ocjenjivanja te uvida u ocjenu. Slika 2.3, prikazuje odabir kolegija unutar sustava Merlin.



Slika 2.3 Prikaz odabira kolegija unutar sustava Merlin

3. MODELIRANJE BAZE PODATAKA

Proces modeliranja baze podataka predstavlja stvaranje koncepta podatka unutar baze podataka te definiranje i stvaranje veza među pojedinim, određenim objektima temeljenih na poimanju stvarnog svijeta [6].

Proces modeliranja rastavlja se na tri djela:

- Analiza korisničkih zahtjeva, utvrđivanje zahtjeva i postojećih podataka
- Kreiranje logičkih i konceptualnih modela
- Pretvaranje, odnosno prevođenje logičkog modela u oblik pogodan za implementaciju

Svaki podatak unutar baze podataka organizira se prema nekom modelu. Model podataka predstavlja skup pravila koja određuju samo strukturu baze podataka. Model je osnova za projektiranje i implementaciju baze podataka. DBMS (engl. *Database Management System*) je softver dizajniran za upravljanje podacima u bazi podataka, a podržava neke od različitih modela:

- Relacijski model – Predstavlja matematičko poimanje relacije. Podaci i veze prikazani su tablicama.
- Mrežni model – Baza podataka prikazuje se usmjerenim grafom. Čvorovi predstavljaju tip, a lukovi vezu između tipova zapisa.
- Hijerarhijski model – Predstavlja specijalni slučaj mrežnog modela. Baza podataka predstavljena je stablom.
- Objektni model – Baza podataka je predstavljena skupom objekata koji se sastoje od podataka i operacija. Objekti pripadaju određenim klasama, a između pojedinih klasa se uspostavljaju veze[7].

3.1. Analiza zahtjeva

Prilikom registracije, korisnik unosi vlastito ime i prezime, željeni račun elektroničke pošte i zaporku te odabire svoj status („Student“, „Djelatnik“). Nakon autorizacije, korisnika se usmjerava na stranicu za upis, odnosno kreiranje kolegija, ovisno o korisničkoj ulozi. Određeni kolegij može upisati više studenata, a djelatnik može kreirati više, različitih kolegija. Nakon kreiranja kolegija, djelatnik unosi naslove, odnosno teme seminarskih radova kao i broj studenata za svaku pojedinu temu. Nakon odabira kolegija, studenti se upisuju na temu seminarskog rada. Djelatniku je omogućeno uređivanje, kreiranje, brisanje, odnosno ispis pojedinih studenata, tema seminarskih radova, ocjena i slično. Student može odabrati željeni kolegij te se upisati na željenu temu. Po završetku pisanja seminarskog rada, student rad pohranjuje na stranici teme, a potom djelatnik, odnosno mentor, temu pregledava i ocjenjuje.

3.2. Projektiranje baze podataka

Projektiranjem se uz pomoć pravila i određenih specifikacija oblikuje građa baze podataka. Analiza podataka utvrđuje vrstu podataka koju baza podataka treba sadržavati te određuje što se s tim podacima može raditi. Proces projektiranja nudi rješenje i predlaže načine kako grupirati podatke te na koji način ih povezati. Rezultat procesa projektiranja treba biti shema, kreirana s pravilima i zapisana tako da ju DBMS može realizirati.

Projektiranje je podijeljeno na nekoliko koraka:

- Prvi korak je projektiranje na konceptualnoj razini koja kao rezultat kreira konceptualnu shemu koja se sastoji od entiteta te veza među njima.
- Drugi korak je projektiranje na logičkoj razini koja kao rezultat kreira logičku shemu. U slučaju relacijskog modela, shema je sastavljena od tablica. Glavi dio projektiranja na logičkoj razini čini normalizacija u kojoj se uz pomoć pravila popravlja logička struktura relacija.
- Treći korak je projektiranja na fizičkoj razini koja kao rezultat kreira fizičku shemu baze podataka, odnosno daje opis cijele fizičke građe. Fizička shema je zapravo niz SQL naredbi koje realiziraju prethodno kreirane tablice[8].

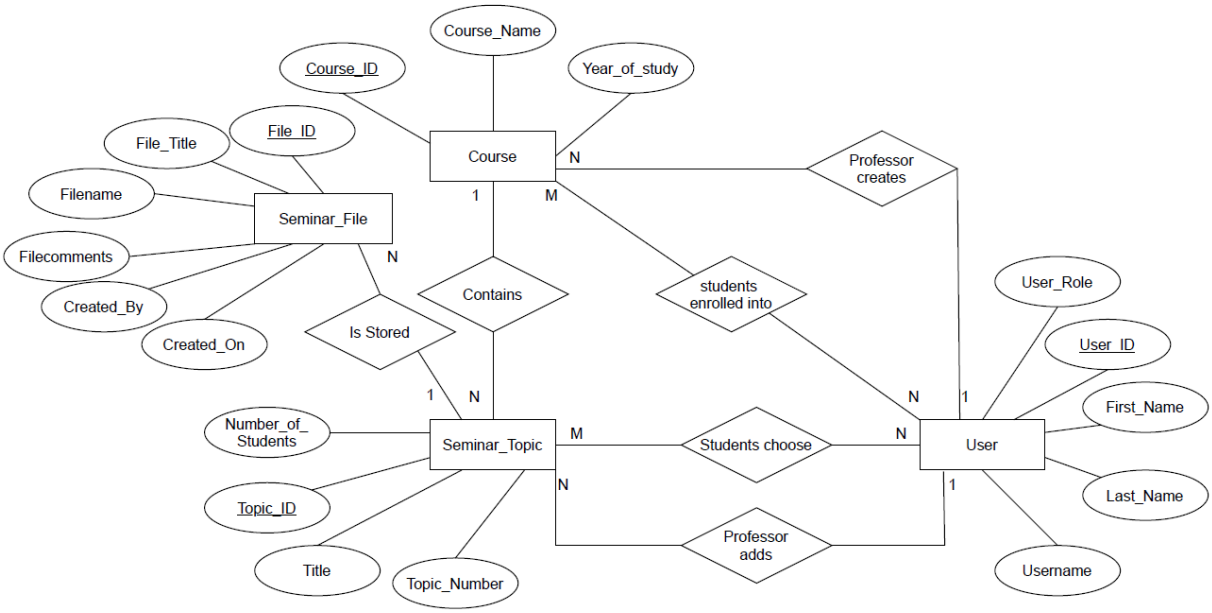
3.3. Stvaranje modela

Postupak stvaranja modela odnosi se na kreiranje entiteta te njihovo međusobno povezivanje binarnim vezama. Entitet predstavlja objekt unutar baze podataka u kojeg se spremaju podaci. Entitet može biti na primjer: student, kolegij, tema seminara. Atributi opisuju, odnosno jednoznačno označavaju svaki pojedini entitet, na primjer: ime i prezime. Svaki entitet prikazan je tablicom ili relacijom. Veza predstavlja odnos među entitetima. Vrste veza su sljedeće:

- Veza 1:1 znači da jednom zapisu unutar entiteta odgovara isključivo jedan zapis u drugom entitetu;
- Veza 1:N znači da zapisu u jednom entitetu odgovara jedan, nula ili više zapisa u drugom entitetu;
- Veza N:M znači da jedan zapis u entitetu može biti u vezi s nula, jednim ili više zapisa u drugom entitetu, vrijedi i obrnuto[9].

Skup relacija čini bazu podataka, a za svaku relaciju vrijedi da ima ime po kojem se razlikuje od ostalih relacija u toj bazi podataka. Jedan stupac u relaciji sadrži vrijednost jednog atributa, a atribut također ima svoje ime po kojem se razlikuje od ostalih[10].

Za prikaz relacije između entiteta u nekom sustavu, koristi se ER (engl. *Entity-Relationship*) dijagram. Primjer ER dijagrama prikazan je na slici 3.1.

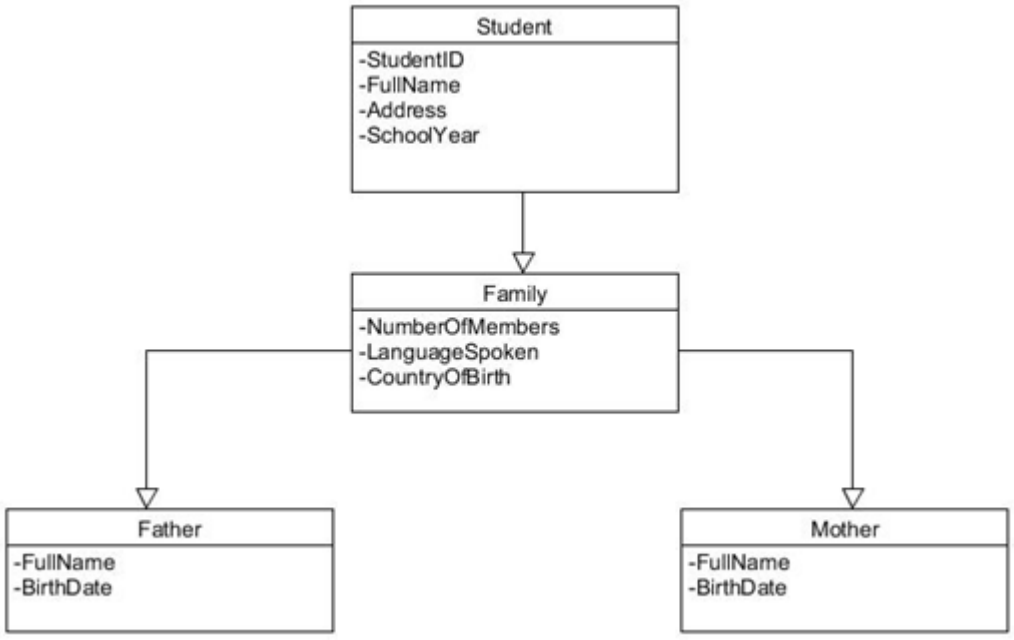


Slika 3.1 Prikaz ER dijagram Aplikacija za mentoriranje studentskih seminara

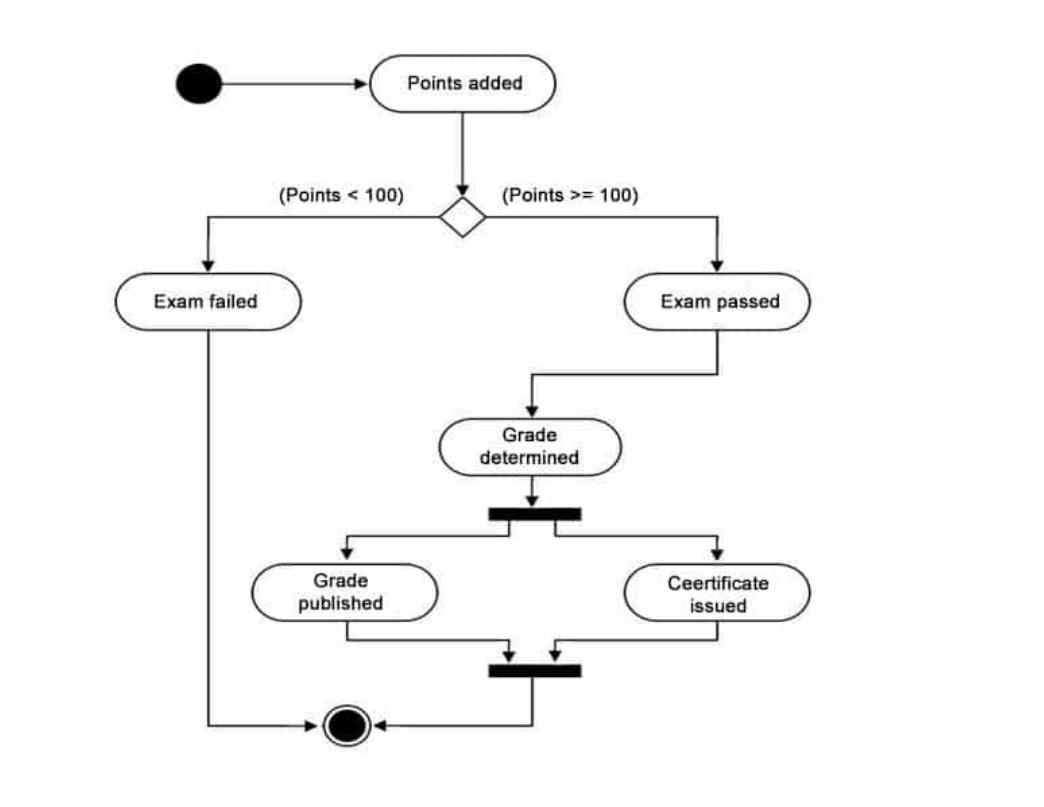
U slučaju kada postoji velik broj entiteta s mnogo atributa te s velikim brojem veza među njima, savjetuje se prelazak u notaciju jedinstvenog jezika za modeliranje UML (engl. *Unified Modelling Language*). UML je grafički jezik koji se koristi kada želimo vizualizirati i konstruirati softverski intenzivni sustava. UML nudi jednostavan i standardan način pisanja sustava. Također uključuje mogućnosti poput poslovnih procesa i funkcija sustava, kao i dijelove programskog jezika, sheme baza podataka i softverske komponente za višekratnu upotrebu.

UML dijagrami predstavljaju dva različita pogleda na model sustava:

- Statički prikaz - Ovaj pogled naglašava statičku strukturu sustava koja koristi objekte, atribute, operacije i odnose. Primjer: Dijagram klase, Dijagram složene strukture.
- Dinamički prikaz - Ovo stajalište naglašava dinamično ponašanje sustava pokazujući suradnju među objektima i promjene unutarnjih stanja objekata. Primjer: Dijagram slijeda, Dijagram aktivnosti[11].



Slika 3.2 Prikaz primjera statičkog ili strukturnog dijagrama u UML notaciji



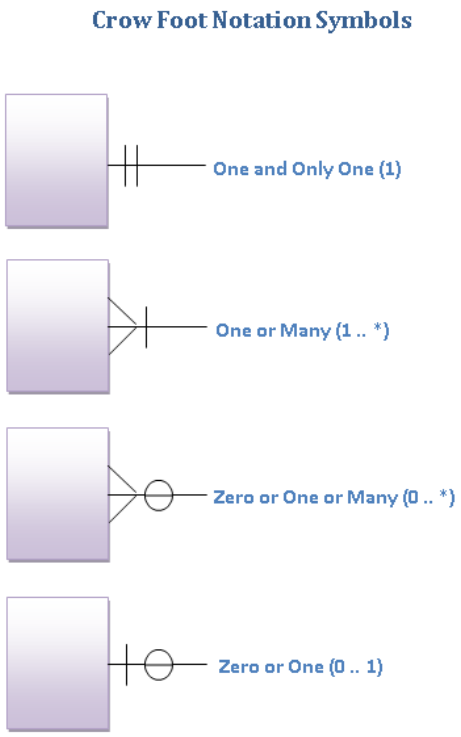
Slika 3.3 Prikaz primjera dinamičkog ili ponašajnog dijagrama u UML notaciji

Vidljivo je na slici 3.2 kako su svi atributi strukturnog dijagrama sadržani zajedno. Veze su predstavljene crtama koje sadrže simbole koji označavaju kardinalnost veze. Ti znakovi poznati su pod nazivom vranino stopalo (engl. *Crow's foot notation*). Takav zapis često se koristi u softverskom inženjerstvu i dizajniranju baza podataka. Začetnik notacije bio je Gordon Everest, koji je ponudio ideju kako vizualno predstaviti različite vrste odnosa koji mogu postojati između objekata u dijagramu odnosa entiteta (ER). Često se koristi kako bi se određeni podaci tablice mogli pozivati na podatke u drugim tablicama unutar baze podataka[12].

Vrste veza vidljive na slici 3.4

- Linija okomita na liniju – označava obavezno članstvo u kojem sudjeluju minimalno jedan i maksimalno jedan entitet
- Vranino stopalo i crtica – označava obavezno članstvo u kojem sudjeluju minimalno jedan i maksimalno više entiteta
- Vranino stopalo i prsten – označava neobavezno članstvo u kojem sudjeluju minimalno nula i maksimalno više entiteta

- Crtica i prsten - označava neobavezno članstvo u kojem sudjeluju minimalno nula i maksimalno jedan entitet[13].



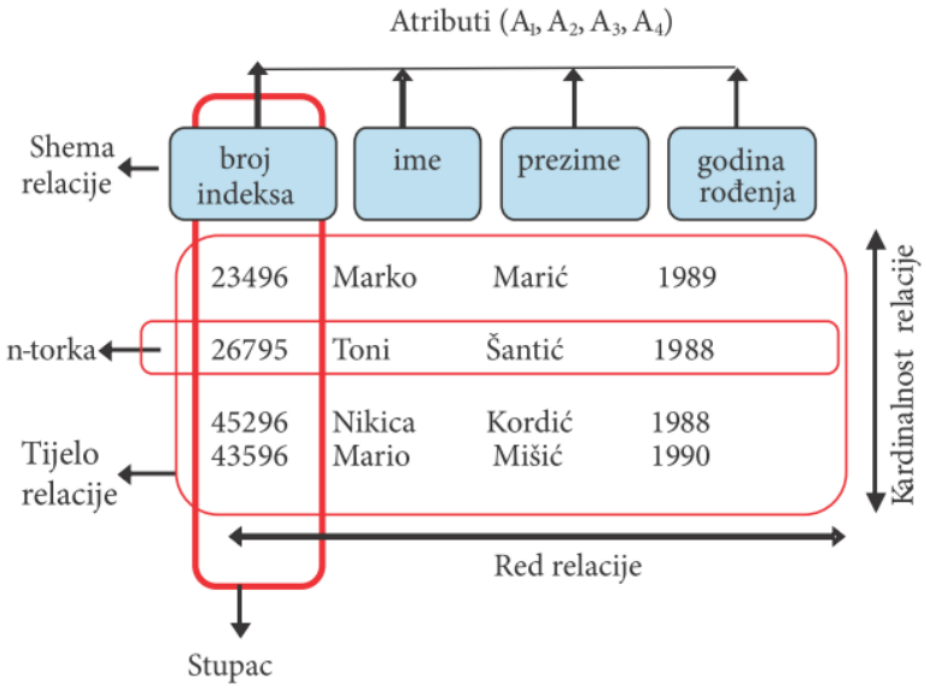
Slika 3.4 Prikaz Crow Foot notacije

Nakon kreiranja ER modela, slijedi pretvaranje u relacijski model. Unutar relacijskog modela, sadržani su podaci interpretirani preko tri kategorije:

- Prva je kategorija strukture,
- Druga je kategorija ograničenja unutar modela,
- Treća je kategorija operatora unutar modela.

Shema relacijskog modela definira strukturu, a elementi strukture su relacije[14].

Relacija ima svoje ime prema kojem se razlikuje od ostalih unutar baze podataka. Stupac unutar relacije sadrži vrijednost atributa stoga se poistovjećuje s atributom, a vrijednost su podaci istoga tipa. Također svaki atribut ima svoje ime prema kojem se razlikuje od ostalih. Domena atributa predstavlja definirani skup dopuštenih vrijednosti, a vrijednost je jednostruka i jednostavna te nije obavezna za unos. Jedan redak unutar relacije predstavlja jedan zapis[9]. Slika 3.5, prikazuje zaglavlje i tijelo relacije.



Slika 3.5 Prikaz zaglavlja i tijela relacije, izvor: [15]

Unutar relacijskog modela postoje ograničenja kao što su:

- Primarni ključ (engl. *primary key*) - kandidat ključ odobren da jedinstveno odredi n-torku unutar relacije.
- Vanjski ili strani ključ (engl. *foreign key*) - atribut ili skup atributa unutar jedne relacije koji odgovara kandidat ključu druge relacije.
- Entitetski integritet - primarni ključ ne može imati *Null* vrijednost.
- Referencijalni integritet - svaka vrijednost vanjskog ključa mora biti uparena [15].

Kada pretvaramo ER model u relacijski važno je pažnju posvetiti pretvorbi veze u odgovarajuće entiteta i strane ključeve. Pravila prilikom transformacije su:

- Veza kardinalnosti 1 : N – Sa strane veze s oznakom jedan, primarni ključ se dodaje kao strani entitetu na strani s oznakom veze više,
- Veza kardinalnosti N : M – Kreira se novi entitet koji sadrži složeni primarni ključ koji je načinjen od primarnih ključeva obaju entiteta.
- Usporedne veze – Svaku vezu zamijenimo stranim ključem na strani s oznakom veze više,

- Povratne veze – Predstavlja vezu entiteta sa samim sobom, a transformira se tako da se dodaje strani ključ koji je jednak primarnom ključu unutar te relacije, ali s drugim imenom[16].

Prevođenjem ER modela sa slike uz primjenu navedenih pravila kreira se relacijska shema baze podataka, odnosno skup relacija prikazanih tablicom 3-1. Rječnik podataka za bazu podataka iz tablice 3-1, prikazan je u tablici 3-2.

Tablica 3-1 Relacijska shema baze podataka Aplikacije za mentoriranje studentskih seminara

APPLICATION_USERS	(User_ID, First_Name, Last_Name, Email, User_Password, User_Role, Is_Verified)
COURSE	(Course_ID, Course_Name, Year_of_Study, Study_Program, Created_By(User_ID))
CHOOSEN_COURSE	(User_ID, Course_ID)
SEMINAR_TOPIC	(Topic_ID, Topic_Number, Title, Short_Description, Number_of_Students, Enrolled, Grade, Course_ID)
CHOOSEN_TOPIC	(Topic_ID, User_ID)
FILE_INFO	(File_ID, File_Title, Documants, Filename, File_Mmetype, File_Charset, File_Comments, Created_By, Created_On, Active, Topic_ID)

Tablica 3-2 Rječnik podataka za bazu podataka Aplikacije za mentoriranje studentskih seminara

Ime atributa	*tip*	*opis*
USER_ID	Cjelobrojni tip	Jedinstvena oznaka korisnika
FIRST_NAME	Niz od 50 znakova	Ime osobe
LAST_NAME	Niz od 50 znakova	Prezime osobe
E-MAIL	Niz od 50 znakova	E-mail osobe
USER_PASSWORD	Niz od 50 znakova	Jedinstvena lozinka korisnika
USER_ROLE	Niz od točno 25 znakova	Uloga korisnika
IS_VERIFIED	Jedan znak	Potvrda o autorizaciji
COURSE_ID	Cjelobrojni tip	Jedinstvena oznaka kolegija
COURSE_NAME	Niz od 200 znakova	Naziv kolegija
YEAR_OF_STUDY	Cjelobrojni tip	Na kojoj godini se kolegij sluša
STUDY_PROGRAM	Niz od 500 znakova	Studijski program
CREATED_BY	Cjelobrojni tip	Jedinstvena oznaka profesora
TOPIC_ID	Cjelobrojni tip	Jedinstvena oznaka teme
TOPIC_NUMBER	Cjelobrojni tip	Broj teme
TITLE	Niz od 200 znakova	Naziv teme
SHORT_DESCRIPTION	Niz od 1000 znakova	Kratki opis teme
NUMBER_OF_STUDENTS	Cjelobrojni tip	Broj studenata po temi
ENROLLED	Cjelobrojni tip	Upisani studenti na temu
GRADE	Cjelobrojni tip	Ocjena
FILE_ID	Cjelobrojni tip	Jedinstvena oznaka seminara
FILE_TITLE	Niz od 300 znakova	Naziv teme seminara
DOCUMENTS	BLOB	Pohranjivanje binarnih podataka kao jedan entitet
FILENAME	Niz od 200 znakova	Naziv datoteke
FILE_MMETYPE	Niz od 40 znakova	Višenamjenska proširenja
FILE_CHARSET	Niz od 40 znakova	Kodiranje znakova dokumenta
FILE_COMMENTS	Niz od 1000 znakova	Komentar
CREATED_BY	Niz od 40 znakova	Oznaka autora
CREATED_ON	Datum	Datum kreiranja dokumenta
ACTIVE	Niz od 20 znakova	Specifikacija vrste podataka za aktivne oznake i aktivne sheme

3.4. Fizičko oblikovanje baze podataka

Stvaranjem modela kreira se relacijska shema uz pomoć koje se fizički oblikuje baza podataka. Za oblikovanje koristimo SQL kako bi kreirali naredbe za definiranje objekata. SQL je razvio IBM, a koristi se kako bi postavili upit, ažurirali ili pak definirali relaciju. Lako ga je koristiti jer koristi rječnik sličan engleskom jeziku. Također moguće ga je koristiti za aritmetičke operacije, sortiranje podataka, definiranje pogleda i slično[7].

Prilikom fizičkog oblikovanja baze podataka, najčešće korištene DDL (engl. *Data definition language*) naredbe su CREATE i ALTER TABLE, koje omogućuju stvaranje te uređivanje entiteta, ali bez promjene sadržaja[17]. Osnovna sintaksa CREATE TABLE naredbe, prikazana je na slici 3.6.

```
CREATE TABLE
[ database_name.[ owner ]. | owner. ] table_name
( { < column_definition >
  | < table_constraint > }
)

< column_definition > ::= { column_name data_type }
[ [ DEFAULT constant_expression ]
[ < column_constraint > ] [ ...n ]

< column_constraint > ::= [ CONSTRAINT constraint_name ]
    { [ NULL | NOT NULL ]
        | [ { PRIMARY KEY | UNIQUE } ]
        | [ [ FOREIGN KEY ]
            REFERENCES ref_table [ ( ref_column ) ]
            [ ON DELETE { CASCADE | NO ACTION } ]
            [ ON UPDATE { CASCADE | NO ACTION } ]
        ]
        | CHECK ( logical_expression )
    }
```

Slika 3.6 Osnovna sintaksa CREATE TABLE naredbe, izvor:[17]

```
1:      CREATE TABLE SEMINAR_TOPIC(
2:      Topic_ID Number Primary key,
3:      Topic_Number Number,
4:      Title Varchar2(200),
5:      Short_Description Varchar2(1000)
6:      Number_of_Students Number,
7:      Enrolled Number,
8:      Grade Number,
9:      Course_ID Number REFERENCES COURSE(Course_ID)
10:     );
```

Slika 3.7 Prikaz korištene SQL naredba za kreiranje tablice SEMINAR_TOPIC

Slika 3.8, prikazuje osnovnu sintaksu ALTER TABLE naredbe.

```
ALTER TABLE table
{ [ ALTER COLUMN column_name
    { new_data_type [ ( precision [ , scale ] ) ]
    [ NULL | NOT NULL ] } ]
  | ADD { [ < column_definition >
          | < table_constraint > ] } [ ,...n ]
  | DROP { [ CONSTRAINT ] constraint_name
           | COLUMN column } [ ,...n ]
```

Slika 3.8 Osnovna sintaksa ALTER TABLE naredbe, izvor:[17]

Osim naredbi CREATE i ALTER TABLE, postoje DML naredbe za manipulaciju podacima:

- Naredba SELECT – koristi se kod prikaza sadržaja pojedine tablice.
- Naredba INSERT – koristi se kod dodavanja novog sadržaj u već postojeću tablicu.
- Naredba DELETE – koristi se kod uklanjanja, odnosno brisanja sadržaj iz tablice.
- Naredba UPDATE – koristi se prilikom unosa promjena.
- Naredba COMMIT WORK – koristi se prilikom fiksiranja promjena.
- Naredba ROLLBACK WORK – koristi se prilikom poništavanja svih promjena.
- Naredba SAVEPOINT – koristi se prilikom djelomičnog poništavanja promjene.
- Naredba COMMIT WORK – koristi se kod potvrđivanja promjena[14].

Slika 3.9, prikazuje osnovnu sintaksu SELECT naredbe, a na slici 3.10, prikazan je SELECT upit na tablicu CHOOSEN_COURSE koji studentu pruža jednostavan uvid na upisane kolegije. Student potom odabire željeni kolegiji i vrši odabir željene teme.

Osnovni oblik SELECT naredbe je:
SELECT [ALL/DISTINCT] atribut [, atribut...] **FROM** relacija [, relacija...]
[**WHERE** uvjet]
[**GROUP BY** atribut [, atribut]]
[**HAVING** uvjet]
[**ORDER BY** specifikacija uređaja [ASC/DESC]]

Slika 3.9 Osnovna sintaksa SELECT naredbe, izvor:[14]

```
1:      SELECT Course.Course_ID,
2:      Course.Course_Name,
3:      Course.Study_program,
4:      Course.Year_of_Study,
5:      Course.Created_By
6:      FROM CHOOSEN_COURSE, COURSE
7:      WHERE Course.Course_ID = Choosen_Course.Course_ID AND
        Choosen_Course.User_ID IN ( SELECT USER_ID FROM APPLICATION_USERS
        WHERE upper(Email) = v('APP_USER'));
```

Slika 3.10 Prikaz korištene SELECT naredbe

Sa slike 3.10, vidljivo je korištenje WHERE naredbe, odnosno uvijeta koji je zadovoljen za sve kolegije koje je upisao student. Unutar WHERE dijela, prilikom provjere koristimo ugniježđeni SELECT, pri čemu je potrebno voditi računa na sljedeće:

- Ako je ugniježđeni SELECT argument operatora koji očekuje jednu vrijednost (npr. operator =) onda se rezultat tog SELECT-a mora sastojati od samo jednog retka i samo jednog stupca.
- Ako je ugniježđeni SELECT argument operatora koji očekuje listu vrijednosti (npr. operator IN) onda se rezultat tog SELECT-a mora sastojati od samo jednog stupca [18].

Ugniježđeni SELECT u ovom primjeru će vratiti USER_ID korisnika za kojega se provjerava prethodno napisan uvjet.

Aplikacija za mentorianje studentskih seminara vodi brigu o svojim korisnicima koje prilikom registracije pohranjuje u tablicu APPLICATION_USERS. Izgled tablice prikazan je slikom 3.11.

```
1:      CREATE TABLE  "APPLICATION_USERS" (
2:      "USER_ID" NUMBER,
3:      FIRST_NAME" VARCHAR2(50) NOT NULL ENABLE,
4:      "LAST_NAME" VARCHAR2(50) NOT NULL ENABLE,
5:      "EMAIL" VARCHAR2(50) NOT NULL ENABLE,
6:      "USER_PASSWORD" VARCHAR2(50) NOT NULL ENABLE,
7:      "USER_ROLL" VARCHAR2(25) NOT NULL ENABLE,
8:      "IS_VERIFIED" VARCHAR2(1),
9:      PRIMARY KEY ("USER_ID")
        USING INDEX  ENABLE,
10:        UNIQUE ("EMAIL")
        USING INDEX  ENABLE,
11:        UNIQUE ("USER_PASSWORD")
12:     USING INDEX  ENABLE
```

Slika 3.11 Prikaz tablice APPLICATION_USERS

Prilikom prijave u aplikaciju, provjerava se status korisnika uz pomoć korisnički definirane funkcije. Takva funkcija ili procedura nameće istu poslovnu logiku svim klijentima. Takav pristup pruža sigurnost da se prilikom obrade podataka koriste isti postupci te sprječava klijente da od tih postupaka odstupaju. Ovim pristupom olakšavaju se sve moguće izmjene ili nadogradnje te se postiže stabilnosti i konzistentnost u radu.

Korištenjem SQL servera, omogućeno je pisanje vlastitih, korisnički definiranih funkcija i procedura s tabličnim vrijednostima (engl. *table-valued functions*). Prilikom pisanja takvih funkcija koristimo CLR (engl. *Microsoft .NET Framework Common Language Runtime*). Funkcija će vratiti jednu vrijednost, a povratna vrijednost skalarne funkcije definira se naredbom RETURN. Prilikom izvršavanja naredbe RETURN, treba koristiti komponente imena sheme i naziv funkcije. Argumenti se predaju unutar zagrada koje slijede nakon imena funkcije. Skalarna funkcija može sadržavati i podrazumijevane parametre, ali oni nisu obavezni. Ukoliko želimo koristiti podrazumijevane vrijednosti parametara potrebno je iskoristiti ključnu riječ DEFAULT[19].

Primjer korištenja funkcije prikazan je slikom 3.12.

```
        FUNCTION USER_AUTHENTICATION(
        p_username varchar2,
1:      p_password varchar2) RETURN BOOLEAN
2:      AS
3:      v_ps_check varchar2(1);
4:      BEGIN
5:      SELECT 'x' into v_ps_check
6:      FROM APPLICATION_USERS
7:      WHERE upper(EMAIL) = upper(p_username)
8:      AND upper(USER_PASSWORD) = upper(p_password)
9:      AND IS_VERIFIED = 'v';
        apex_util.set_authentication_result(0);
10:          RETURN TRUE;
11:     exception WHEN no_data_found THEN
        apex_util.set_authentication_result(4);
12:          RETURN FALSE;
13:     END;
```

Slika 3.12 Prikaz funkcije verifikacije korisnika

Funkcija USER_AUTHENTICATION kao ulazne parametre prima korisničko ime i lozinku, a kao povratnu vrijednost vraća BOOLEAN tip podatka. Funkcija provjerava podudaraju li se uneseni podaci s onima koji su prilikom registracije pohranjeni unutar tablice. U slučaju da se uneseni podaci podudaraju s pohranjenima, korisnik je potvrđen. Potvrđeni korisnik tada može koristiti aplikaciju. Procedura za umetanje podataka u tablicu korisnika sa slike 3.11, prikazana je na slici 3.13.

```
1:      DECLARE
2:      UserID NUMBER;
3:      BEGIN
4:      SELECT NVL(MAX(UER_ID),0)+1 INTO UserID FROM APPLICATION_USERS;
5:      INSERT INTO APPLICATION_USERS
6:      (USER_ID,FIRST_NAME,LAST_NAME,EMAIL,USER_PASSWORD,USER_ROLL,IS_VERIFIED)
        VALUES(UserID, FIRST_NAME, LAST_NAME, EMAIL,USER_PASSWORD, USER_ROLL,
7:      'v');
8:      END;
```

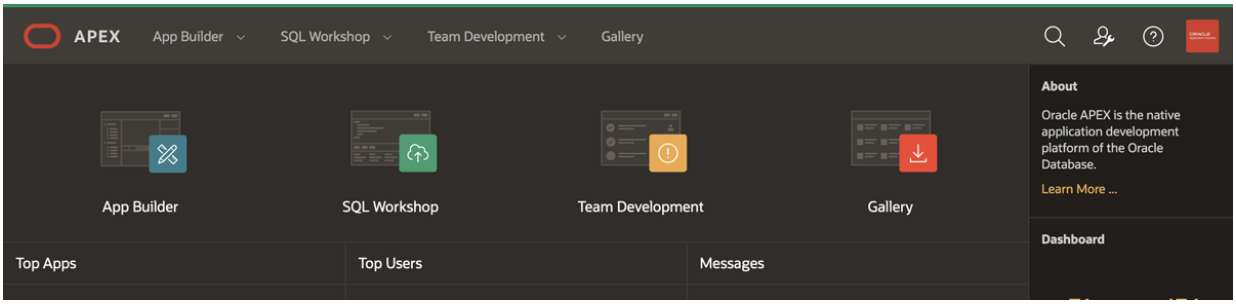
Slika 3.13 Procedura za umetanje podataka u tablicu APPLICATION_USERS

4. STVARANJE APLIKACIJE U ORACLE APEX OKRUŽENJU

Oracle APEX smanjuje složenost višestranih aplikacija i nudi mogućnost kreiranja web aplikacije za rješavanje složenih upita i problema bez pretjeranog poznavanja web tehnologija. Prilikom odabira odnosno prijave u razvojno okruženje, ponuđene su tri opcije:

- Usluga razvoja aplikacija - Oracle APEX pruža unaprijed konfigurirano, potpuno upravljano i sigurno okruženje u Oracle Cloudu za izgradnju i implementaciju aplikacija.
- Oracle APEX nudi besplatni radni prostor na https://apex.oracle.com/en/. Potrebno je nekoliko trenutaka za obavljanje prijave te pripremu prostora za početak izrade prve aplikacije.
- Oracle APEX je dostupan za preuzimanje i instaliranje u bilo kojem izdanju Oracle baze podataka.

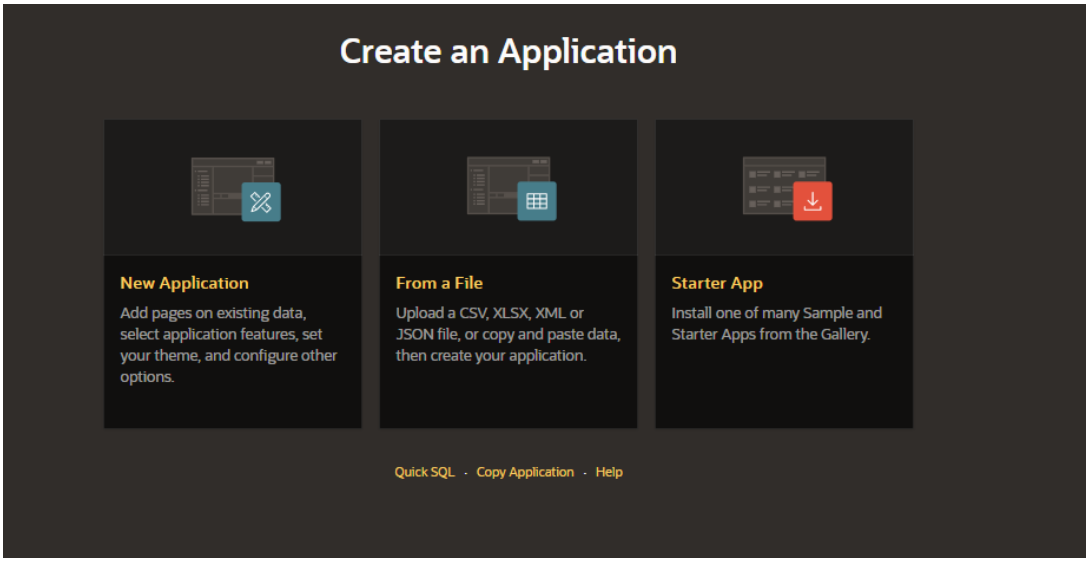
Po završetu, korisnika se preusmjerava na početnu stranicu vidljivu na slici 4.1 s četiri glavna alata.



Slika 4.1 Prikaz naslovne stranice Oracle APEX programerskog sučelja

- App Builder - alat za razvoj aplikacije.
- SQL Workshop - jednostavno okruženje koje nudi mogućnost razvoja baze i izvođenja naredbi.
- Team Development - alat koji služi za praćenje napretka.
- Gallery - kolekcija unaprijed ugrađenih aplikacija.

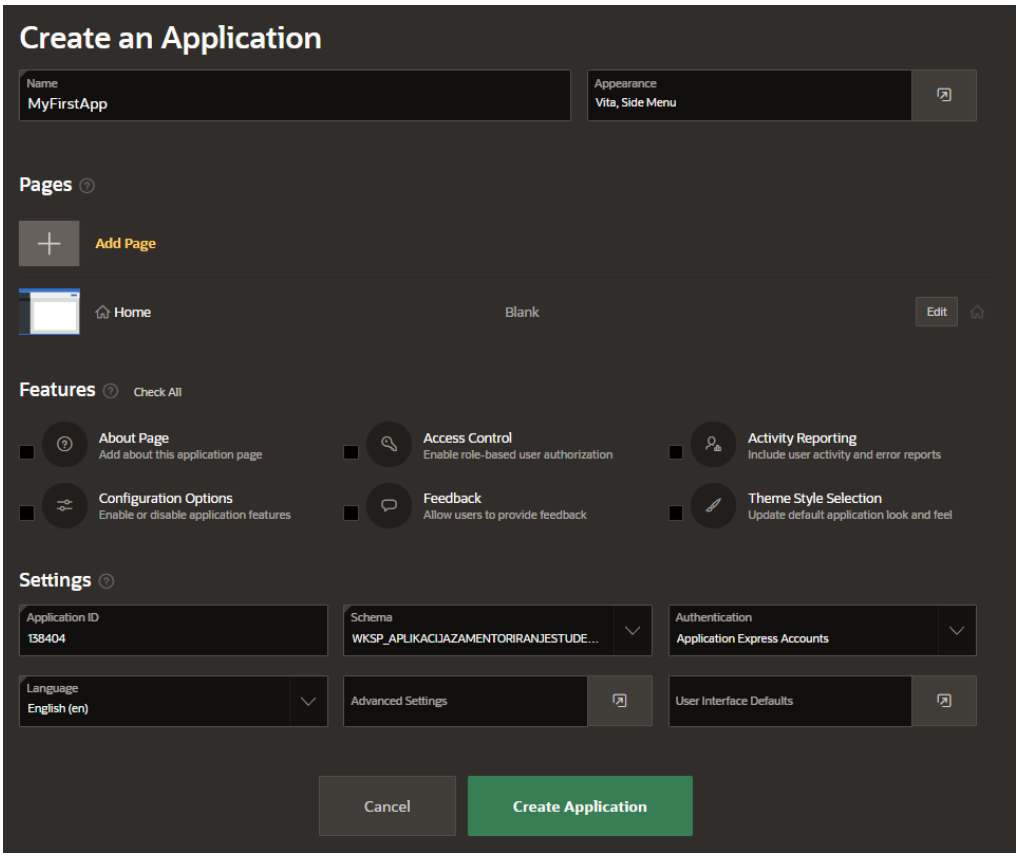
Pokretanjem App Builder alata, ponuđena je mogućnost kreiranja novih aplikacije i uređivanja odnosno nadogradnje već postojećih. Razvoj aplikacije se započinje tako da se pokrene čarobnjak koji služi za izradu aplikacije, a prikazan je na slikama 4.2 i 4.3.



Slika 4.2 Prikaz izrade aplikacije u Oracle APEX razvojnom okruženju

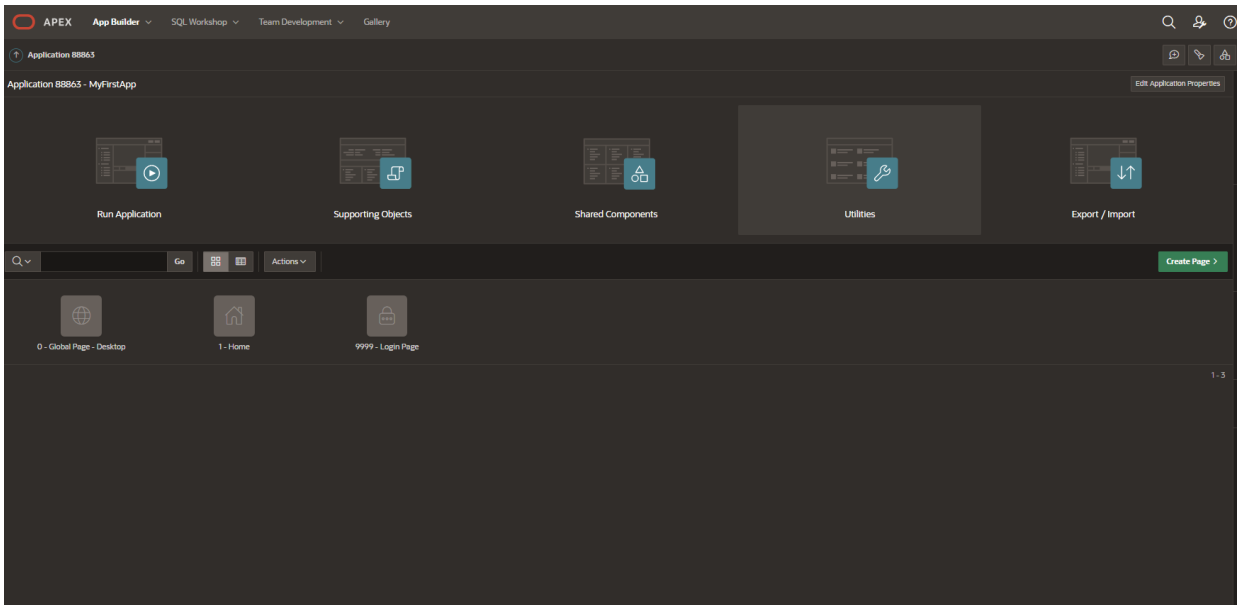
Prilikom kreiranja aplikacije, ponuđene su tri opcije:

- New Application – nudi mogućnost kreiranja aplikaciju bez ikakvih dodatnih podataka, ali nudi mogućnost dodatnog postavljanja teme i konfiguracije.
- From a File – nudi mogućnost kreiranja aplikaciju na osnovu uvezenih podataka.
- Starter App – nudi mogućnost instaliranja neke od prethodno pohranjenih aplikacija iz galerije.



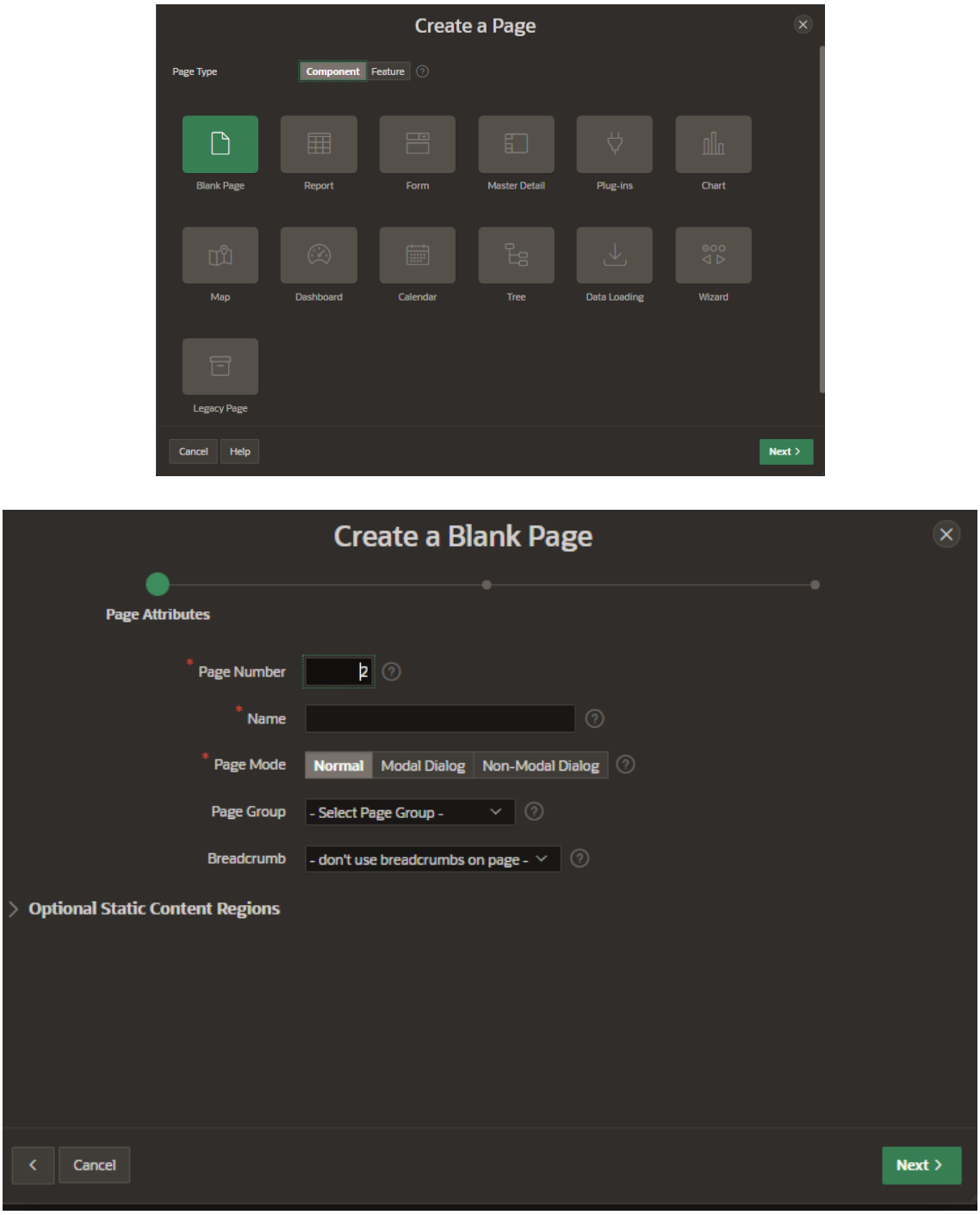
Slika 4.3 Čarobnjak za izradu aplikacija

Unutar čarobnjaka mogu se odabrati mnoge korisne značajke koje su unaprijed ugrađene te se nudi mogućnost dodavanja novih stranica. Kreiranjem aplikacije, unutar App Builder alata, prikazuje se popis kreiranih stranica te se nude razni alati za upravljanje, izmjenu, opcije za uvoz podatak, pokretanje aplikacije i slično. Pregled novonastale aplikacije prikazan je na slici 4.4.



Slika 4.4 Pregled novonastale aplikacije

Pritiskom na „Create Page“, pokreće se čarobnjak za izradu nove stranice, unutar aplikacije. Čarobnjak za izradu stranice prikazan je na slici 4.5.



Slika 4.5 Čarobnjak za izradu stranice

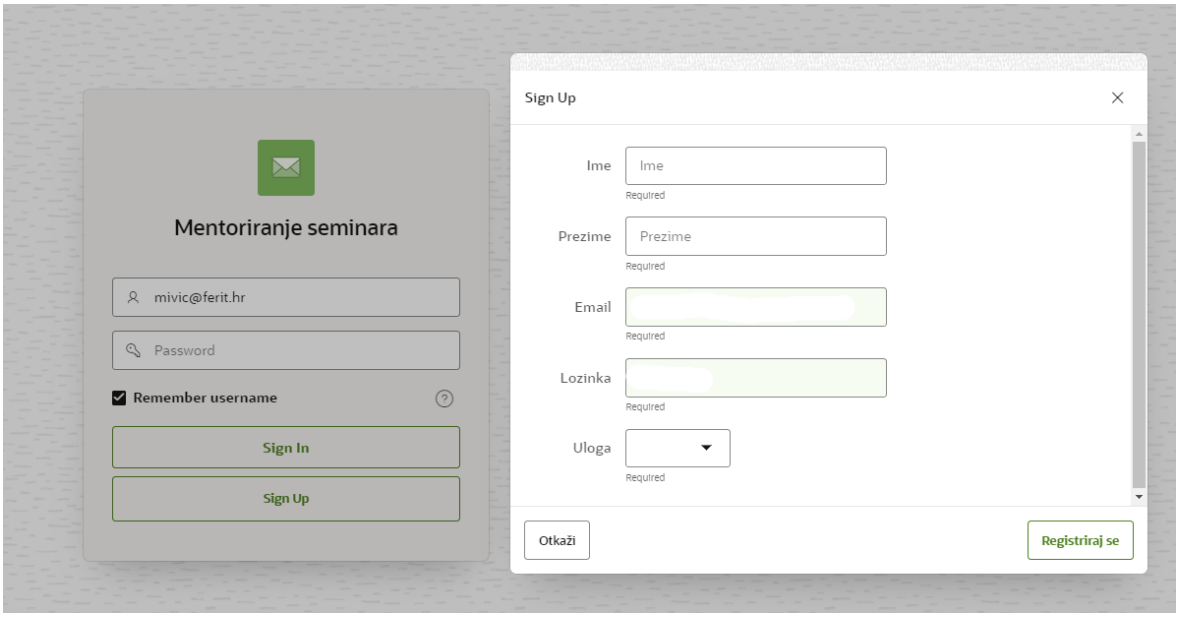
Nakon što je stranica kreirana, pojavljuje se u pregledniku koji nudi mogućnosti izmjene. Moguće je promijeniti raspored stranice, dodati nove elemente ili ih ukloniti te dodatno konfigurirati novonastalu stranicu.

4.1. Izgled aplikacije

Aplikacija se sastoji od jedanaest stranica od kojih je pet stranica implementirano kao modalni dijalog. Stranice poput registracijskog obrasca, upisa kolegija i slično, su izvedene kao poseban oblik skočnih prozora. Kada korisnik posjeti web aplikaciju, pruža mu se mogućnost registriranja i prijave kako bi mogao koristiti aplikaciju. Na slici 4.6, prikazana je početna stranica, Aplikacije za mentoriranje studentskih seminara, prije prijave, odnosno registracije korisnika.

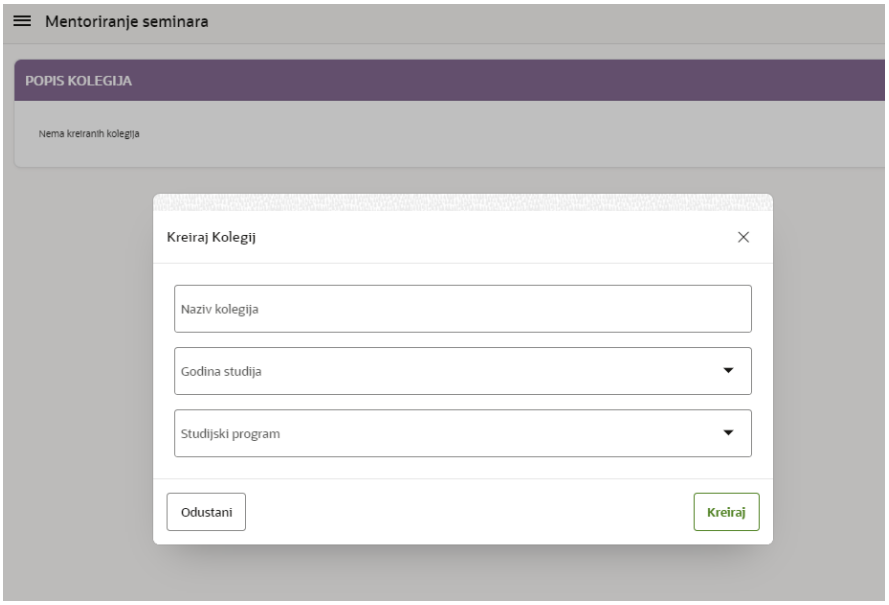
Slika 4.6 Prikaz početne stranice

Stranica prijave i obrazac registracije prikazani su na slici 4.7.



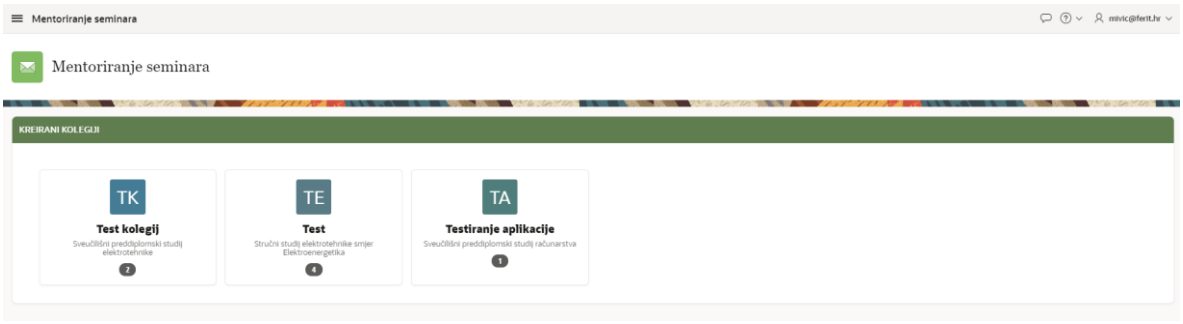
Slika 4.7 Stranica prijave i obrazac registracije

Nakon uspješno odrađene autorizacije, korisnika se preusmjerava na početnu stranicu i nudi mu se mogućnost na stranici Kolegij, kreirati ili upisati određeni kolegij, ovisno o ulozi korisnika. Slika 4.8, prikazuje stranicu Kolegij i obrazac kreiranja kolegija.



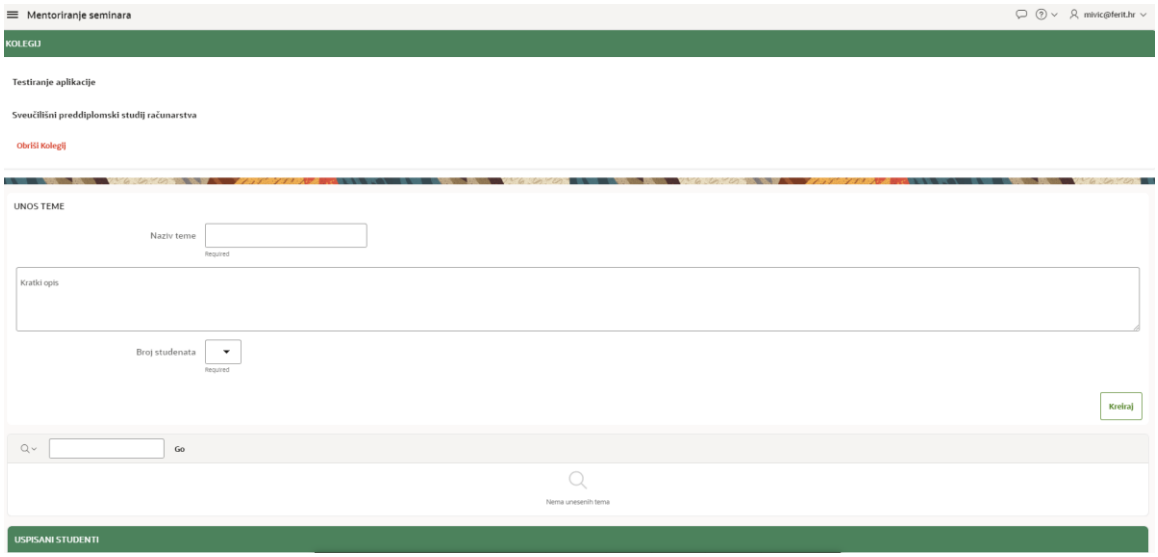
Slika 4.8 Stranica Kolegij i obrazac kreiranja kolegija

Nakon kreiranog odnosno upisanog kolegija, djelatniku će biti prikazani svi kolegiji koje je kreirao, a student će moći vidjeti i pristupiti kolegijima koje je prethodno upisao. Početna stranica s upisanim, odnosno kreiranim kolegijima prikazana je na slici 4.9.



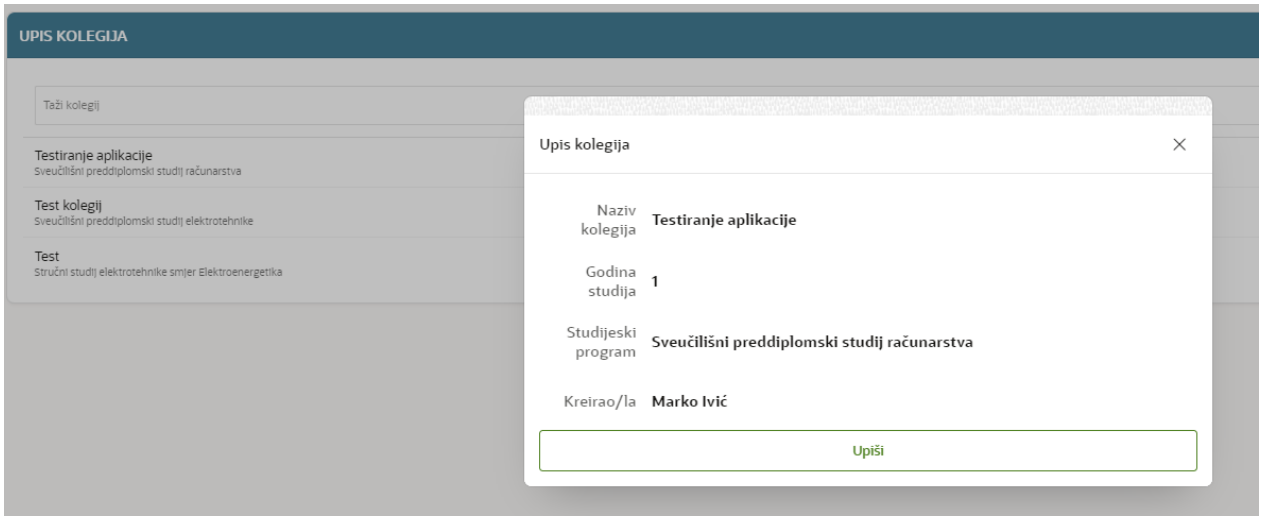
Slika 4.10 Prikaz početne stranice s kreiranim kolegijima

Prilikom odabira željenog kolegija, korisnika se preusmjerava na stranicu Ekolegij na kojoj su prikazani podaci za svaki kolegij pojedinačno. Stranica Ekolegij prikazana je na slici 4.10 .

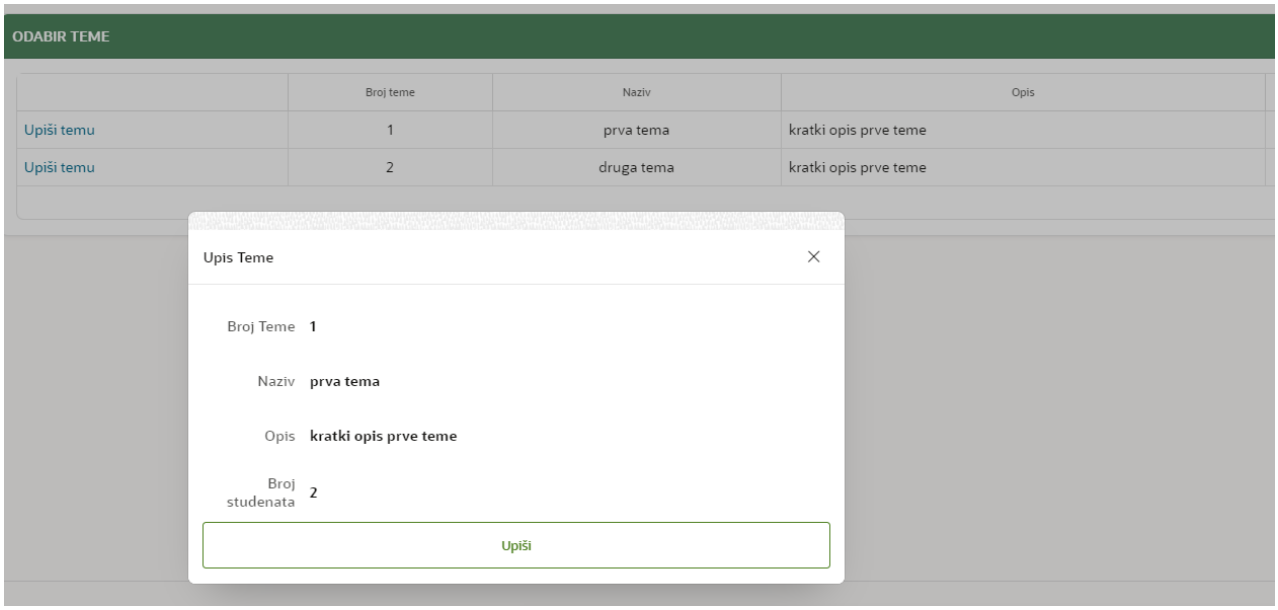


Slika 4.9 Prikaz stranice Ekolegij

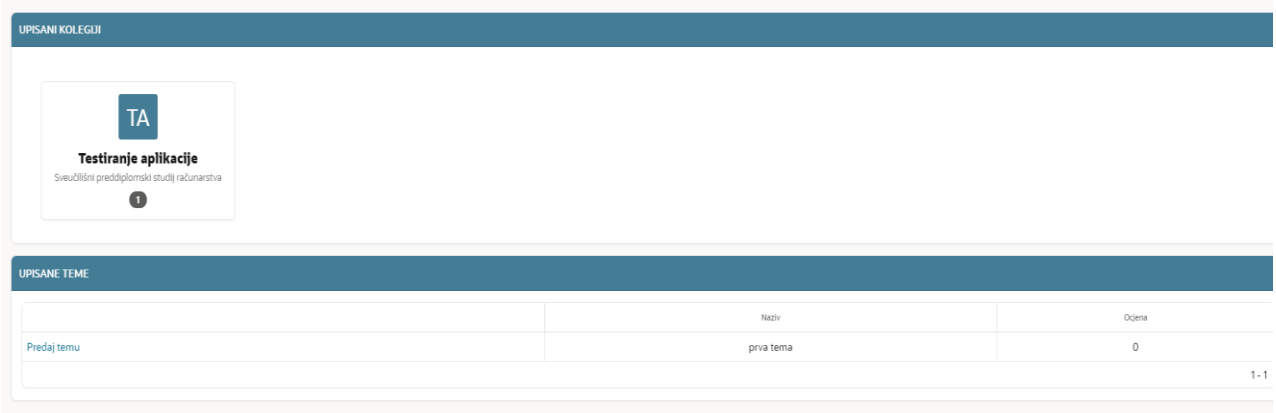
Stranica Ekolegij djelatniku nudi mogućnost kreiranja teme, brisanje pojedinačnih tema, ocjenjivanje te pregled upisanih studenata i njihov ispis na kraju akademske godine. Također omogućena je opcija brisanja cijelog kolegija. Student na stranici Ekolegij vrši upis na željenu temu koja mu je potom prikazana na početnoj stranici. Prikaz upisa na kolegij te odabir teme prikazan je na slikama 4.11 i 4.12 .



Slika 4.11 Prikaz upisa studenata na kolegij

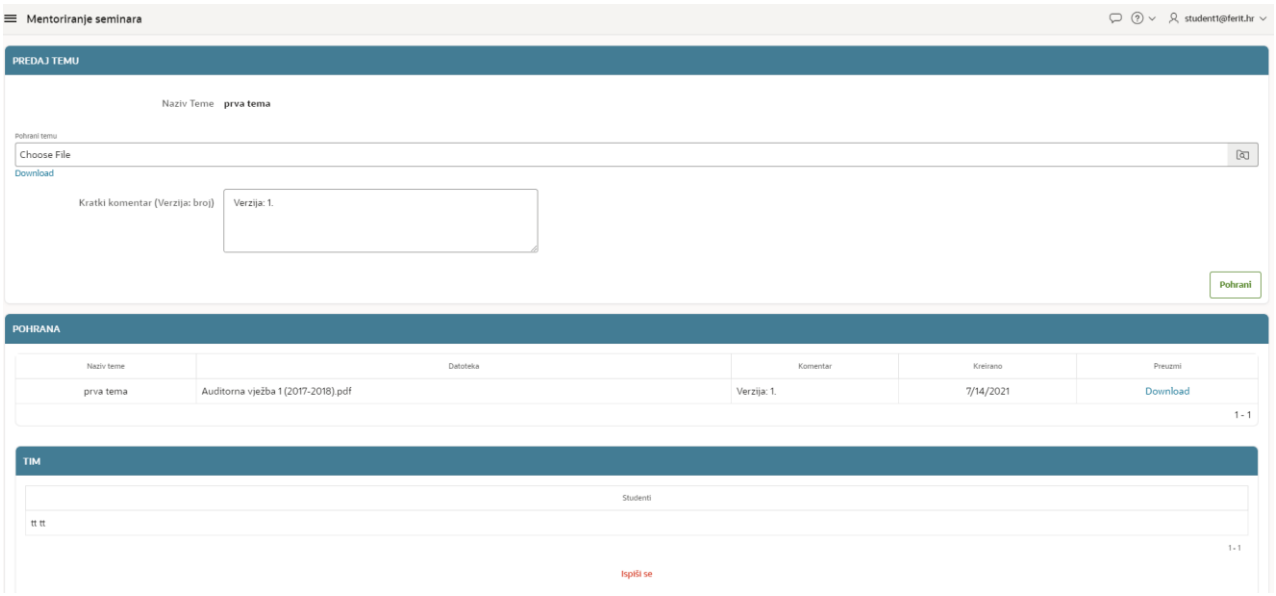


Slika 4.12 Prikaz odabira teme seminarskog rada



Slika 4.13 Prikaz početne stranice s odabranim temama seminarskih radova

Pritiskom na „Predaj temu“, studenta se preusmjerava na stranicu Etema na kojoj se nalaze pojedinosti vezane za temu i vrši predaja napisanog seminarskog rada, kojeg potom djelatnik na istoj stranici, pregledava i ocjenjuje. Studentu je omogućeno predavanje više verzija rada. Također pritiskom na „Ispiši se“, omogućen je ispis s teme. Djelatniku je omogućeno preuzimanje rada ili čitanje putem web servisa te brisanje pojedinih studenata s određene teme.



Slika 4.14 Prikaz stranice Etema

5. ZAKLJUČAK

Zadatak završnog rada bio je izraditi web aplikaciju koja će ponuditi novu i bolju funkcionalnost prilikom odabira i mentoriranja seminarskih radova studenata Fakulteta elektrotehnike, računarstva i informacijskih tehnologija u Osijeku. Cilj je bio pojednostaviti i smanjiti komunikaciju službenim mailom na način da aplikacija nudi mogućnosti i funkcionalnosti koje dosadašnja rješenja nisu nudila. Aplikacije je izrađena koristeći Oracle Application Express (APEX), nisko-kodnu razvojnu platforma koja omogućuje izgradnju skalabilnih, sigurnih poslovnih aplikacija koje se mogu implementirati bilo gdje uz potporu PL/SQL programskog jezika. Najveći dio rada odnosio se na modeliranje odnosno izradu novog modela aplikacije, prilagođenog potrebama zahtjeva korisnika. Na samom početku rada spomenute su korištene tehnologije te je napravljena usporedba s trenutnim sustavima za mentoriranje seminarskih radova. Prilikom modeliranja kratko su objašnjeni koncepti i ideja vezana za dijagrame. Također, objašnjena je sintaksa i pravila dizajniranja kao i pravila transformacije te fizičko oblikovanje baze podataka odnosno pisanje SQL naredbi. U kratkim i najbitnijim crtama, objašnjena je sintaksa i korištenje PL/SQL programskog koji se koristio u procesu fizičkog modeliranja odnosno pisanja koda za umetanja podataka, dohvaćanja, obradu i slično. Na kraju je objašnjen postupak kreiranja jednostavne aplikacije u Oracle APEX razvojnom okviru te je demonstriran konačan izgled kreirane aplikacije kao i njezino korištenje.

Aplikacija za mentoriranje studentskih seminara se može još dorađivati. Moguće je omogućiti, uz minimalne preinake odabir i mentoriranje završnih i diplomskih radova. Također moguće je u potpunosti izbjeći korištenje službenog maila na način da s mentoru omogući izravna komunikacije s studentima preko aplikacije u obliku postavljanja komentara, stupnjeva odrađenosti rada i slično.

LITERATURA

[1] T. Kyte, *Expert Oracle Database architecture: Oracle Database 9i, 10g, and 11g programming techniques and solutions*, 2. ed. Berkeley, Calif.: Apress, 2010. [Na internetu]. Dostupno na: https://javidhasanov.files.wordpress.com/2012/01/expert-oracle-database-architecture.pdf (pristupljeno srp. 08, 2021).

[2] M. Cyran, P. Lane, i J. Polk, „Oracle Database Concepts“, [Na internetu]. Dostupno na: https://docs.oracle.com/cd/B19306_01/server.102/b14220.pdf (pristupljeno srp. 08, 2021).

[3] „Home“, *Oracle APEX*. [Na internetu]. Dostupno na: https://apex.oracle.com/en/ (pristupljeno srp. 08, 2021).

[4] L. Morin i L. Jayapalan, E. Belden, P. Huey, S. Moore, K. Rich, „Database PL/SQL Language Reference“, *Oracle Its Affil. 21c*, [Na internetu]. Dostupno na: https://docs.oracle.com/en/database/oracle/oracle-database/21/lnpls/database-pl-sql-language-reference.pdf (pristupljeno srp. 08, 2021).

[5] F. Almeida, *Developing Effective PL/SQL Reference Guide for Undergraduate Students*. 2015. [Na internetu]. Dostupno na: https://www.researchgate.net/profile/Fernando-Almeida-10/publication/319852598_Developing_Effective_PLSQL_Reference_Guide_for_Undergraduate_Students/links/59be42830f7e9b48a2985999/Developing-Effective-PL-SQL-Reference-Guide-for-Undergraduate-Students.pdf (pristupljeno srp. 08, 2021).

[6] I. Lukić, „Baza podataka - Uvodno predavanje“. [Na internetu]. Dostupno na: https://moodle.srce.hr/20202021/pluginfile.php/4468276/mod_resource/content/6/BP%2001%20Uvodno%20predavanje.pdf (pristupljeno srp. 08, 2021).

[7] Robert Manger, „Baze podataka“, 2003. [Na internetu]. Dostupno na: http://jadran.izor.hr/~dadic/EKO/baze-podataka.pdf (pristupljeno srp. 08, 2021).

[8] Robert Manger, „Osnove projektiranja baza podataka“. [Na internetu]. Dostupno na: https://www.srce.unizg.hr/files/srce/docs/edu/osnovni-tecajevi/d310_polaznik.pdf (pristupljeno srp. 08, 2021).

[9] Silvije Davila i Tina Kaštelan, „Modeliranje podataka“, Zagreb 2010. [Na internetu]. Dostupno na: https://www.algebra.hr/cjelozivotno-obrazovanje/wp-content/uploads/sites/3/2017/12/Administrator-baza-podataka.pdf (pristupljeno srp. 08, 2021).

[10] I. Vazler i M. Đumic, „Osnove baza podataka - Osnovni pojmovi“, *Data Base*, [Na internetu]. Dostupno na: https://www.mathos.unios.hr/~ivazler/obp/obp-1-uvod.pdf

[11] B. Padmanabhan, „Unified Modeling Language (UML) Overview“, *Unified Model. Lang.*, [Na internetu]. Dostupno na: https://people.eecs.ku.edu/~saiedian/Teaching/Fa16/810/Readings/UML-diagrams.pdf (pristupljeno srp. 08, 2021).

[12] „How To Make a Crow’s Foot ER Diagram | ERD Symbols and Meanings | Design Element: Crows Foot for Entity Relationship Diagram - ERD | Crow Feet Notation Pdf Download“, *https://www.conceptdraw.com*. [Na internetu]. Dostupno na: https://www.conceptdraw.com/examples/crow-feet-notation-pdf-download (pristupljeno srp. 08, 2021).

[13] Patrycja Dybka, „Crow’s Foot Notation in Vertabelo“, *Vertabelo Data Modeler*, ožu. 24, 2016. [Na internetu]. Dostupno na: https://vertabelo.com/blog/crow-s-foot-notation-in-vertabelo/ (pristupljeno srp. 08, 2021).

[14] Mile Pavlić, „Oblikovanje baza podataka“, velj. 2011. [Na internetu]. Dostupno na: https://ris.hr/wp-content/uploads/Oblikovanje-Baza-podataka.pdf (pristupljeno srp. 08, 2021).

[15] I. Lukić, „Baza podataka - Relacijski model podataka“. [Na internetu]. Dostupno na: https://moodle.srce.hr/2020-2021/pluginfile.php/4468277/mod_resource/content/5/BP%2002%20Relacijski%20model.pdf (pristupljeno srp. 08, 2021).
[16] I. Lukić, „Baza podataka - Modeliranje podataka“. [Na internetu]. Dostupno na: https://moodle.srce.hr/2020-2021/pluginfile.php/4468278/mod_resource/content/4/BP%2003%20Modeliranje%20podataka.pdf (pristupljeno srp. 08, 2021).
[17] I. Lukić, „Baza podataka - Procedure, funkcije i okidači.pdf“. [Na internetu]. Dostupno na: https://moodle.srce.hr/20202021/pluginfile.php/4468282/mod_resource/content/3/BP%2007%20Procedure%2C%20funkcije%20i%20okida%C4%8Di.pdf (pristupljeno srp. 09, 2021).
[18] „Baze podataka SQL - složeni upiti i funkcije“, [Na internetu]. Dostupno na: https://web.math.pmf.unizg.hr/~goranc/SQL_vj3.pdf (pristupljeno srp. 09, 2021).
[19] Željko Kovačević, *Modeliranje, Implementacija i Administracija baze podataka*. Zagreb: Tehničko veleučilište u Zagrebu Vrbik 8, Zagreb, 2018. [Na internetu]. Dostupno na: https://www.researchgate.net/publication/335727854_Modeliranje_implementacija_i_administracija_baza_podataka (pristupljeno srp. 09, 2021).

SAŽETAK

Cilj ovoga rada jest izraditi aplikaciju koja pruža nove funkcionalnosti i uklanja postojeće probleme prilikom odabira, mentoriranja i ocjenjivanja seminarskih radova. Aplikacija je izrađena u Oracle APEX razvojnom okviru koristeći PL/SQL programski jezik. Prije kreiranja fizičkog modela odnosno pisanja koda, bilo je potrebno kreirati ER i relacijskog modela baze podataka. Osim modeliranja same baze, objašnjen je i proces kreiranja aplikacije u APEX razvojnom okruženju. Na kraju je demonstriran konačan izgled aplikacije koja omogućuje kreiranje kolegija, unos tema seminarskih radova od strane mentora, prijavu na željeni kolegij, odabir odnosno upis željene teme, njezinu pohranu i ocjenjivanje.

Ključne riječi: APEX, baza podataka, mentoriranja seminara, Oracle, PL/SQL

ABSTRACT

Student seminar mentoring application

The aim of this paper is to create an application that provides new functionalities and eliminates existing problems in the selection, mentoring and evaluation of seminar papers. The application was created in the Oracle APEX development framework using the PL/SQL programming language. Before creating a physical model that is writing code, it was necessary to create an ER and a relational database model. In addition to modeling the same databases, the process of creating applications in the APEX development environment is also explained. At the end, the final layout of the application was demonstrated, which enables the creation of a course, entry of seminar paper topics by the mentor, registration for the desired course, selection of the desired topic, topic storage and evaluation.

Key words: APEX, data base, mentoring seminars, Oracle, PL/SQL

POPIS KRATICA

ASM	Oracle Automatic Storage Management
Oracle APEX	Application Express
SQL	Structured Query Language
PL/SQL	Procedural Language for SQL
DBMS	Database Management System
UML	Unified Modelling Language
ER	Entity-Relationship
DDL	Data definition language
IBM	International Business Machines Corporation
DML	Data Manipulation Language
CLR	Common Language Runtime

POPIS SLIKA

Slika 2.1 Struktura blokova PL/SQL, izvor: [5] **4**
Slika 2.2 Prikaz odabira teme seminara na kolegiju Baze podataka **5**
Slika 2.3 Prikaz odabira kolegija unutar sustava Merlin **5**
Slika 3.1 Prikaz ER dijagram Aplikacija za mentoriranje studentskih seminara **8**
Slika 3.2 Prikaz primjera statičkog ili strukturnog dijagrama u UML notaciji **9**
Slika 3.3 Prikaz primjera dinamičkog ili ponašajnog dijagrama u UML notaciji **10**
Slika 3.4 Prikaz Crow Foot notacije **11**
Slika 3.5 Prikaz zaglavlja i tijela relacije, izvor: [15] **12**
Slika 3.6 Osnovna sintaksa CREATE TABLE naredbe, izvor:[17] **15**
Slika 3.7 Prikaz korištene SQL naredba za kreiranje tablice SEMINAR_TOPIC **16**
Slika 3.8 Osnovna sintaksa ALTER TABLE naredbe, izvor:[17] **16**
Slika 3.9 Osnovna sintaksa SELECT naredbe, izvor:[14] **17**
Slika 3.10 Prikaz korištene SELECT naredbe **18**
Slika 3.11 Prikaz tablice APPLICATION_USERS **19**
Slika 3.12 Prikaz funkcije verifikacije korisnika **20**
Slika 3.13 Procedura za umetanje podataka u tablicu APPLICATION_USERS **21**
Slika 4.1 Prikaz naslovne stranice Oracle APEX programerskog sučelja **21**
Slika 4.2 Prikaz izrade aplikacije u Oracle APEX razvojnom okruženju **22**
Slika 4.3 Čarobnjak za izradu aplikacija **23**
Slika 4.4 Pregled novonastale aplikacije **24**
Slika 4.5 Čarobnjak za izradu stranice **25**
Slika 4.6 Prikaz početne stranice **26**
Slika 4.7 Stranica prijave i obrazac registracije **27**
Slika 4.8 Stranica Kolegij i obrazac kreiranja kolegija **27**
Slika 4.9 Prikaz početne stranice s kreiranim kolegijima **28**
Slika 4.10 Prikaz stranice Ekolegij **28**
Slika 4.11 Prikaz upisa studenata na kolegij **29**
Slika 4.12 Prikaz odabira teme seminarskog rada **29**
Slika 4.13 Prikaz početne stranice s odabranim temama seminarskih radova **30**
Slika 4.14 Prikaz stranice Etema **30**

POPIS TABLICA

Tablica 3-1 Relacijska shema baze podataka Aplikacije za mentoriranje studentskih seminara.....13

Tablica 3-2 Rječnik podataka za bazu podataka Aplikacije za mentoriranje studentskih seminara 14

ŽIVOTOPIS

Stjepan Prakljačić je rođen 6. 7. 1999. godine u Osijeku. 2001. godine s roditeljima seli u mjesto Viškovce gdje pohađa Osnovu školu Luke Botića. 2013. godine upisuje se u prvi razred Srednje strukovne škole Antuna Horvata u Đakovu, smjer tehničar za mehatroniku. Tijekom srednjoškolskog obrazovanja aktivno sudjeluje u školskim projektima te osvaja nagradu Hrvatske udruge poslodavaca „Poduzetnici budućnosti!“, za projekt i poduzetničku ideju: „Solarni, prijenosni hladnjak“. 2017. godine uručeno mu je priznanje kao najboljem učeniku u tehničkim zanimanjima, u četverogodišnjem trajanju. Po završetku srednjoškolskog obrazovanja, upisuje se na preddiplomski sveučilišni studij Računarstvo na Fakultetu elektrotehnike, računarstva i informacijskih tehnologija Osijek. Poseban interes su mu Automobilsko računarstvo te razvoj baze podataka.

_____

Potpis autora