PANEVROPSKI UNIVERZITET APEIRON

FAKULTET INFORMACIONIH TEHNOLOGIJA

Redovne studije

Smjer: "Programiranje i softversko inžinjerstvo"

Predmet:

Objektno orjentisano programiranje

(sa primjenom na programskom jeziku Java)

IZDRADA DESKTOP APLIKACIJE "DESKTOP APLIKACIJA ZA KNJIGOVODSTVO U UGOSTITELJSKIM OBJEKTIMA" U JAVA PROGRAMSKOM JEZIKU

Seminarski rad

Predmetni nastavnik:

Doc. dr Saša Salapura

Student:

Jovan Aulić

Indeks br. 143-18/RITP-S

Banja Luka, decembar 2020.

SADRŽAJ

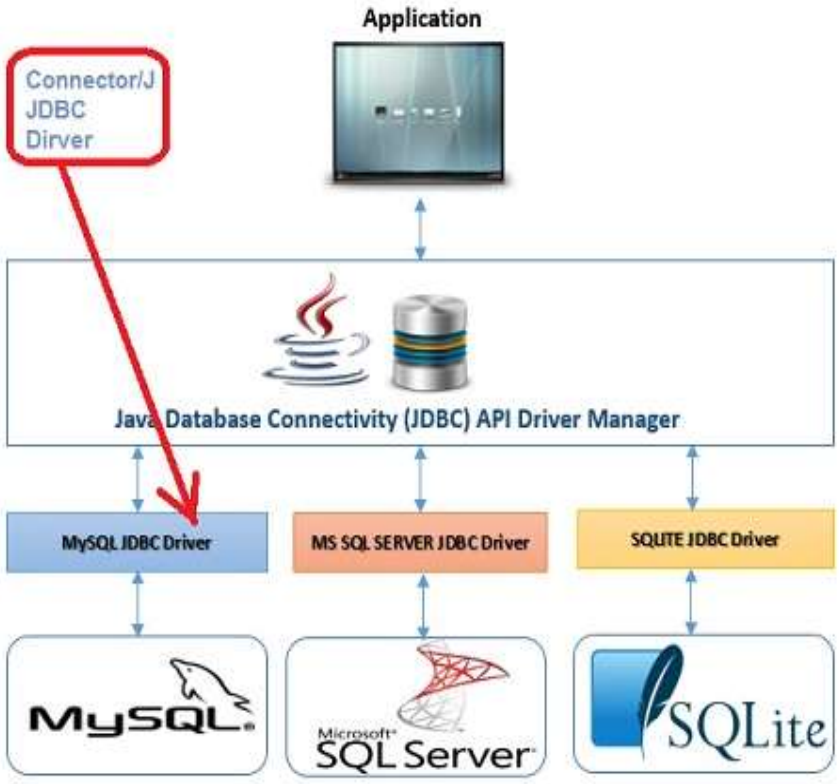
1. UVOD

Java je objektno orjentisani programski jezik koji ima široku namjenu. Jedna od osnovnih pogodnosti , jeste što programi pisani u Javi imaju mogućnost izvođenja na svim operativnim sistemima kod kojih postoji Java Virtual Machine. Neki statistički podaci procjenjuju da broj korisnika koji korisi navedeni programski jezik iznosi od sedam do deset miliona. Prilikom izrade aplikacija za forme grafičkog korisničkog interfejsa koristi se Swing koji predstavlja GUI(Graphical User Interface) kolekciju datoteka koje objedinjuju skup grafičkih kontrolnih elemenata ili eng. widgets za Java programe.

Aplikacija "Desktop aplikacija za knjigovodstvo ugostiteljskih objekata" zamišljena je kao informacioni sistem koji posjeduje mogućnost pohrane i rada sa podacima, a koji se oslanja na osnovnu administraciju potrebnu za vođenje određenih ugostiteljskih objekata. Seminarski rad će pokazati samu strukturu aplikacije kao i pojedine mogućnosti koje sadrži, a to su uvođenje artikala, kontrola troškova i prihoda, kao i dodatne mogućnosti kao što su evidencija smještaja, broja radnika, rad sa resursima i transferima.

2. KORIŠTENE TEHNOLOGIJE

Aplikacija koju obrađuje moj seminarski rad stvarana je u razvojnom okruženju Apache NertBeans IDE 12.1 uz Java SE Development Kit 15.0.1. Programski jezik koji je korišten je Java kao Ii Swing grafičke komponente. Radi mogućnosti veze sa MySQL (8.0.22) uveden je JDBC upravljački program ili drajver Connector/J 8.0.22.



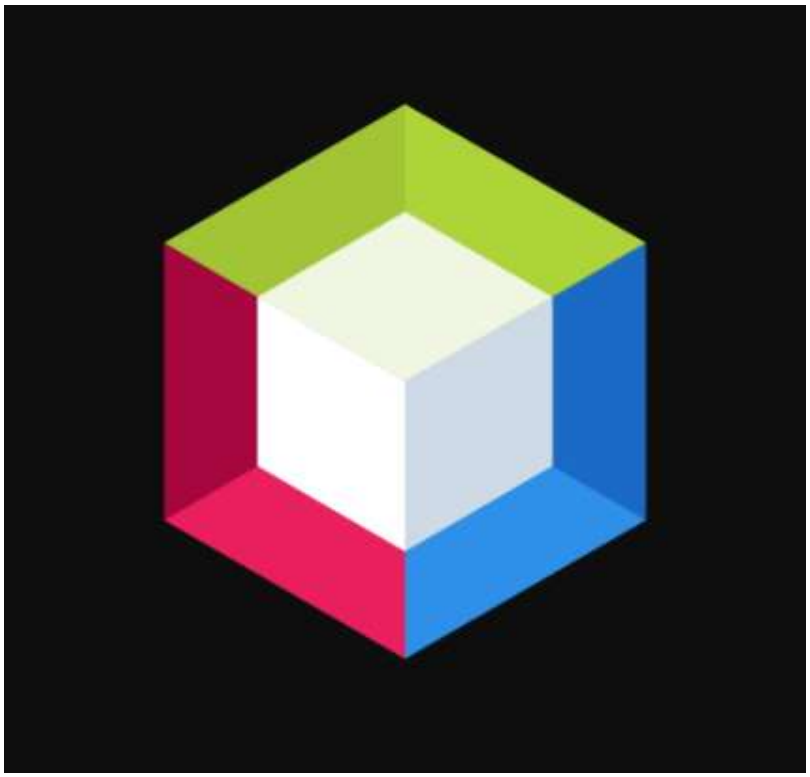
1.0. Connector/J

2.1. APACHE NETBEANS RAZVOJNO OKRUŽENJE

Kao što sam već napomenuo razvojno okruženje korišteno za Java desktop aplikaciju je Apache NetBeans verzija 12.1. On predstavlja integrisano razvojno okruženje IDE, odnosno softverska aplikacija koja pruža veliki broj pogodnosti za razvoj softvera. Primarno je namijenjen programskom jeziku Java, ali se može koristiti i prilikom rada sa drugim jezicima, kao što su Fortran, Python, Rubi i mnogi drugi. Jedna od pogodnosti se ogleda kao mogućnost rada na više platformi.

Neke od glavnih karakteristika NetBeans razvojnog okruženja:

- Razvoj internet aplikacija
- Razvoj desktop aplikacija
- Podrška dinamičkim jezicima(PHP,Rubi)
- Modularna organizacija
- Podrška za upravljanje konfiguracijom
- Podrška za upravljanje metrikom

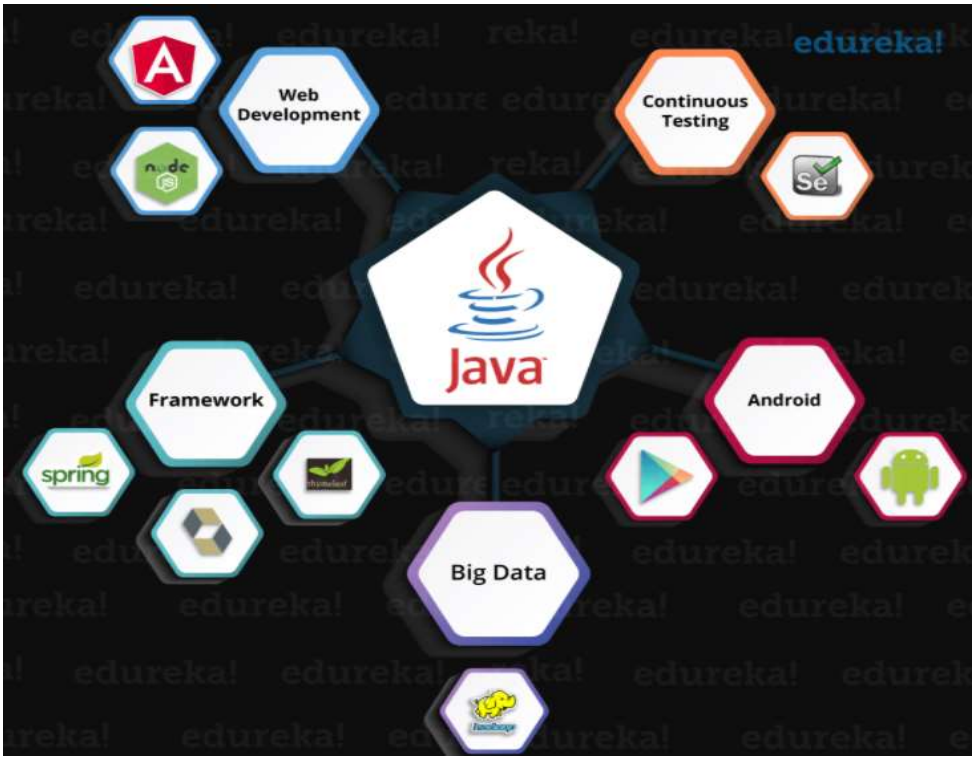


2.0 NetBeans logo

2.2. PROGRAMSKI JEZIK JAVA

Programski jezik Java razvila je Sun Mycrosystems u drugoj polovini devedesetih godina prošlog vijeka. Koncepti Jave zasnovani su na programskom jeziku Oberon. Međutim, razlika je u izbacivanju koncepta modula i uvođenjem objektno orjentisanog programiranja., onosno pojmova klasa i objekata. U odnosu na do tada većinu programskih jezika Java je podržana na svim platformama za koje postoji Java Virtual Machine.

Prvobitno nosio je naziv Oak, ali je zbog autorskih prava dobio je naziv Java po radašnjem slengu za kafu. Odnosno po velikom izvozniku kafe, ostrvu Java. Veoma je popularan i može se pronaći ugrađenim od velikih enterprise aplikacija do smalih "pametnih" IoT uređaja. Posjeduje veliki asortiman alata, biblioteka i koda. Implemetacija se može posmatrati i u razvoju Android uređaja. Popularan je i kod razvoja igrica, pa je jedna od veoma popularnih online igara Minecraft upravo razvijen korištenjem Jave.

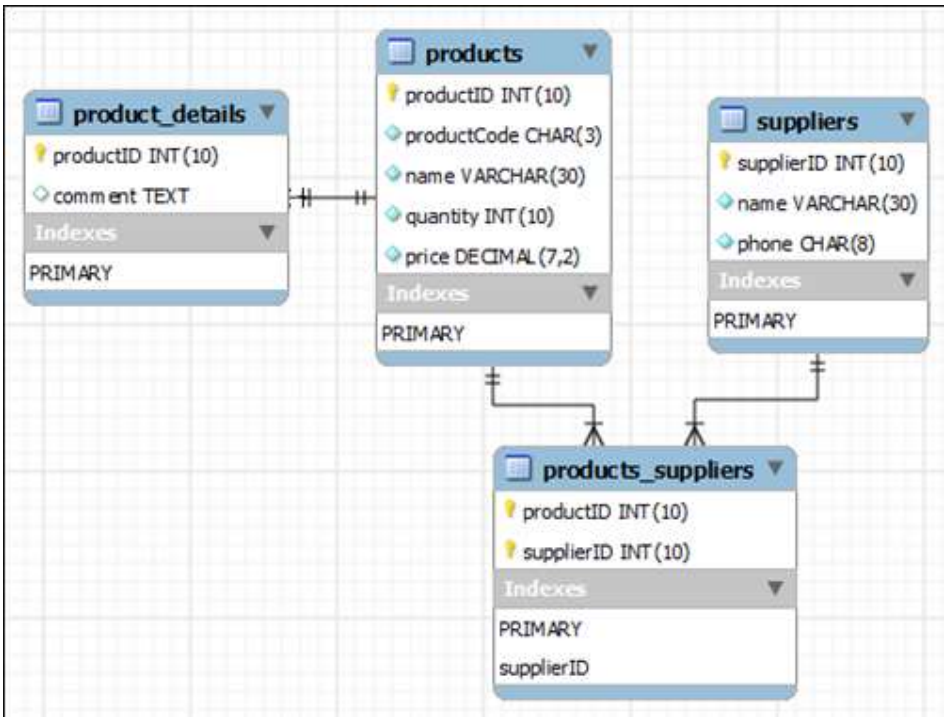


2.1. Java

2.3. MYSQL BAZA PODATAKA

MySQL je besplatan sistem slobodnog koda za upravljanje i rad sa bazama podataka. Princip rada je zasnovan na serveru, a kome mogu pristupiti više korisnika. Prvi put je objavljen u maju mjesecu 1995. godine od strane MySQL LAB, a 2008. ga otkupljuje koorporacija Sun Microsystems. Poznata internet baza podataka je i Wikipedia koja MediaWiki softver razvijen korištenjem PHP programskog jezika i MySql. Biblioteke za pristup ovom sistemu postoje za veliku većinu programskih jezika, a sam MySql je zvanično pisan u jezicima C I C++.

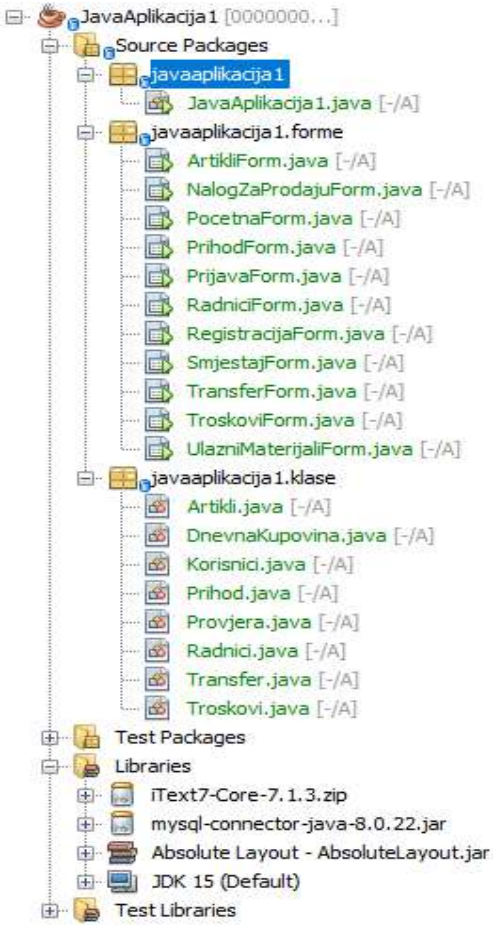
MySql baze su relacionog tipa koji je ujedno i najlakši za organizaciju velikog broja podataka, shodno tome firme ih koriste kako bi lakše izvršile organizaciju i klasifikaciju podataka.



2.2 MySql relacioni model

3. STRUKTURA DESKTOP APLIKACIJE

Strukturalno aplikacija se sastoji od tri paketa. Prvi paket posjeduje main klasu za pokretanje cjelokupne aplikacije. Drugi paket koji nosi naziv "javaaplikacija1.forme" je sačinjen od jedanaest JFrame Form GUI (Graphical User Interface) klasa. Kao što slika 3. prikazuje, postoje klase za autentikaciju(PrijavaForm i RegistracijaForm), zatim PocetnaForm pomoću koje se vrši pristup ostalim klasama, a koje imaju određenje funkcije kao što su evidencija broja radnika(RadniciForm), evidencija o artiklima(ArtikliForm), evidencija o prihodina i troškovima (PrihodForm i TroskoviForm), evidencija o korisnicima usluge smještaja (SmjestajForm), evidencija o ulaznim materijalima i sirovinama(UlazniMaterijaliForm), dodadne opcije evidencija o transkakcijama novca(TransferForm) i klasa funkcije za izdavanje računa(NalogZaProdajuForm).



3. Arhitektura aplikacije

Drugi paket pod nazivom "javaaplikacija1.klase" sadrži osam klasa, od kojih klasa Provjera vrši provjeru unosa podataka JTextField (Polje za unos teksta), ostale klase deklarišu varijable koje uzimaju vrijednost iz baze podataka.

Slike su uvedene iz foldera Ikone koji se nalazi na desktopu.

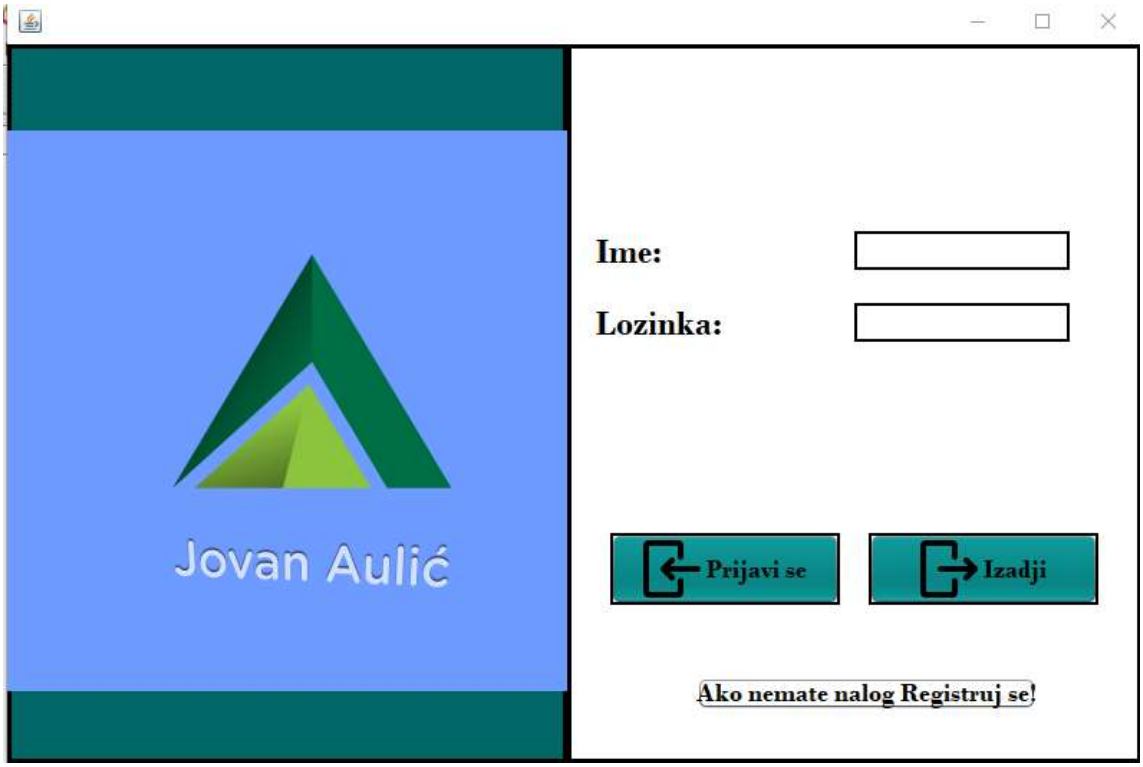
4.0 IZGLED APLIKACIJE

Kao što sam već napomenuo aplikacija se sastoji od tri paketa, pevi paket sadrži JFrame Form klase,a ima ih jedanaest, drugi sa klasama u kojima se deklarišu varijable i sa jednom klasom za provjeru unosa i jedan paket sa main klasom za pokretanje programa.

4.1. IZVRŠNA KLASA I AUTENTIKACIJA

Prvi paket pod nazivom "javaaplikacija1" sadrži jednu izvršnu klasu javaaplikacija1.java. Izvršna se naziva zato šro posjeduje main metodu, a koja se prva izvršava prilikom pokretanja Java aplikaicje. Unutar main metode postavlja se grafička tema Nimbus koja je sveprisutna u cjelokupnoj aplikaciji. Klasa UIManager koja je sadržana u biblioteci javax.sqing sadrži podklasu LookAndFeelInfo sa informacijama o postojanju grafičkih tema. Metodom getInstalledLookAndFeels teži se dobijanju LookAndFeelInfo objekata koji opisuju raspoložive grafičke teme. Ako tema postoji metodom getName se konvertuje naziv teme i vraća u prihvatljivi oblik.

Nakon toga setLookAndFeel postavlja željenu temu, a pomoću getClassName() vraća se ime klase koja implementirau grafičku temu. Dakle prilikom pokretanja aplikacije dolazi prvo do izvršenja main teme, a potom se otvara prozor nasato primjenom Nimbus teme,takozvani JFrame u ovom slučaju to je PrijavaForm.



4.0. Prozor Prijave

Prozor se sastoji od dva JTextField i tri Button komponente, odnosno polja preko kojih se uzima tekst pomoćnu ključne riječi nazivPolja.getText(). Kako bi se usporedili uneseni rezultati sa vrijednostima u bazi podataka, potrebno je prvo dodati metod pomoću kojeg se vrši konekcija na bazu.

```
void prijavaKorisnika(String Ime, String Lozinka) {
    //Adresa servera baze
    String url ="jdbc:mysql://localhost:3306/desktopAplikacija?serverTimezone=UTC";
    boolean flag = false;
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        //Konekcija
        Connection konekcija = DriverManager.getConnection(url,"root","MMOLSX268++-");
        //Statement
        Statement st = konekcija.createStatement();
        //Željena operacija nad bazom podataka
        String izvod = "SELECT * FROM registracija_i_prijava WHERE Ime=\"" +Ime+ "\" and Lozinka=\"" +Lozinka+ "\" ;";
        //Izvršavanje operacije
        ResultSet rs = st.executeQuery(izvod);
        while(rs.next()) {
            flag = true;
        }
        if(flag) {
            JOptionPane.showMessageDialog(null,"Uspjesno ste se prijavili! Dobrodosli " +Ime);
            setVisible(false);
            new PocetnaForm().setVisible(true);
        }else {
            JOptionPane.showMessageDialog(null,"Doslo je do greske prilikom prijave!");
        }
    }catch(HeadlessException | ClassNotFoundException | SQLException greska) {
        JOptionPane.showMessageDialog(null,"Doslo je do greske prilikom povezivanja" +greska);
    }
}
```

4.1. Metoda koja pretražuje vrijednost polja u bazi podataka

Ukoliko korisnik nema još stvoren nalog, može pomoću Button-a namijenjenog za preusmjeravanje otići na GUI klasu za registraciju. Prebacivanje sa jednog prozora na drugi se može ostvariti pomoću akcije: new NazivKlase().setVisible(true);

Nakon izvršene operacije otvoriće se prozor za registraciju korisnika. Ova klasa posjeduje pet polja za unos teksta, a kao i u prethodnom primjeru i ovdje je potrebno stvoriti konekciju na bazu, ali je razlika u funkciji koja se unosi kao upit na bazu :

String izvod = "" + "INSERT INTO registracija_i_prijava VALUES("

+"'" + Ime + "',"

+"'" + Prezime + "',"

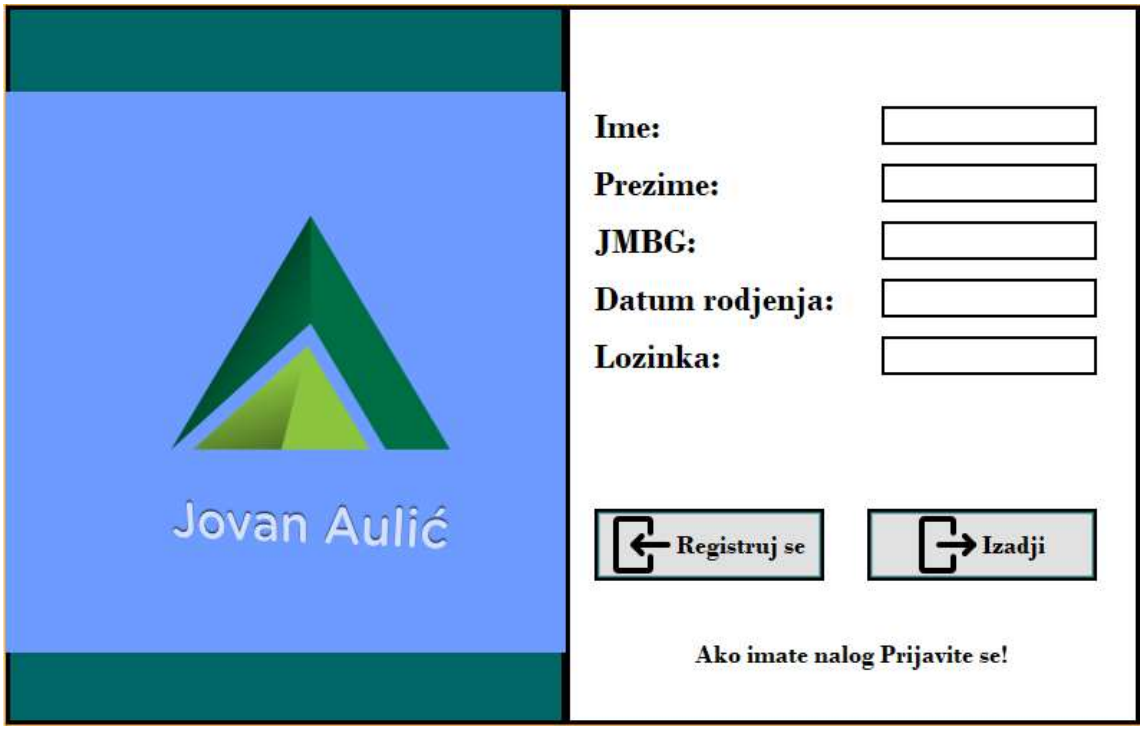
+"'" + JMBG + "',"

+"'" + Datum + "',"

+"'" + Lozinka + "'"

+ ");";

Gdje navedena polja predstavljaju varijable koje su uzelete tekstualne zapisi ključnom riječju getText().



4.2. Izgled prozora za registraciju

4.2 POČETNA STRANA APLIKACIJE

Nakon što se korisnik uspješno prijavio dobiće poruku dobrodošlice “Dobrodošao” sa njegovim imenom. Nakon što se prozor za prijavu zatvori otvoriće se takozvana “početna strana” aplikacije,a koja sadrži ukupno devet elemenata tipa Button za svaku klasu koja izvršava određenu funkciju. Prikaz se kao što sam već napomenuo ostvaruje preko ključnih riječi new NazivGuiKlase().setVisible(True);

Pored fukncionalnih klasa koje izvršavaju određene administrativne zadatke, postoji i jedan Button za akciju “Odjavi se” koja vraća korisnika na stranu prijave.

4.3 KLASA ARTIKLI

Prilikom izdrade prvobitno je potrebno napraviti bazu podataka koja će skladištiti informacije,a samim tim i deklarisati datatype podataka. Pored GUI klase i tabele u MySql bazi podataka bilo je potrebno napraviti još jednu klasu kako bi se stvorile varijable koje će preuzeti na sebe zapis i ispis podataka u i iz baze,a koji moraju odgovarati tipovima podataka deklarisanih u bazu. Shodno tome imamo klasu Artikli u paketu “javaaplikacija1.klase”.

Takođe deklarisanjem varijabli i metoda, potrebno je napraviti takozvane Gettere koji će se uzeti u obzir prilikom upisa u tabelu.

```java
public class Artikli {

    private int id;
    private String nazivArtikla;
    private float kupovnaCijena;
    private float prodajnaCijena;
    private String sifraArtikla;

    public Artikli(int Id, String Naziv_artikla, float Kupovna_cijena, float Prodajna_cijena, String Sifra_artikla ) {

        this.id = Id;
        this.nazivArtikla = Naziv_artikla;
        this.kupovnaCijena = Kupovna_cijena;
        this.prodajnaCijena = Prodajna_cijena;
        this.sifraArtikla = Sifra_artikla;

    }

    public int getId() {
        return id;
    }

    public String getNazivArtikla() {
        return nazivArtikla;
    }
```

4.3. Klasa Artikli

Nakon ovoga prelazimo na GUI klasu ArtikliForm koja posjeduje nekoliko metoda. Na početku uvodimo bibiloteku sql, to postižemo sa import java.sql.*; kako bi smo mogli deklarisati atribute:

- Statement st;
- ResultSet rs;
- PreparedStatement ps;

Koje ćemo koristiti prilikom metoda za rad sa bazama podataka.

Prvi metod ove klase jeste getConnection pomoću kojeg stvaramo konekciju na bazu:

```
public Connection getConnection() {
    Connection con;
    try{
        con = DriverManager.getConnection("jdbc:mysql://localhost:3306/desktopAplikacija?serverTimezone=UTC","root","MMOLSX268++-");
        return con;
    } catch(SQLException e) {
        JOptionPane.showMessageDialog(null,e);
        return null;
    }
}
```

Nakon ovog metoda imamo i metod za izvršenje operacije nad bazom podataka, odnosno metod koji će uzimati parametre String upita nad bazom i String poruke koja će biti ispisana ukoliko je operacija izvršena ili nije izvršena.

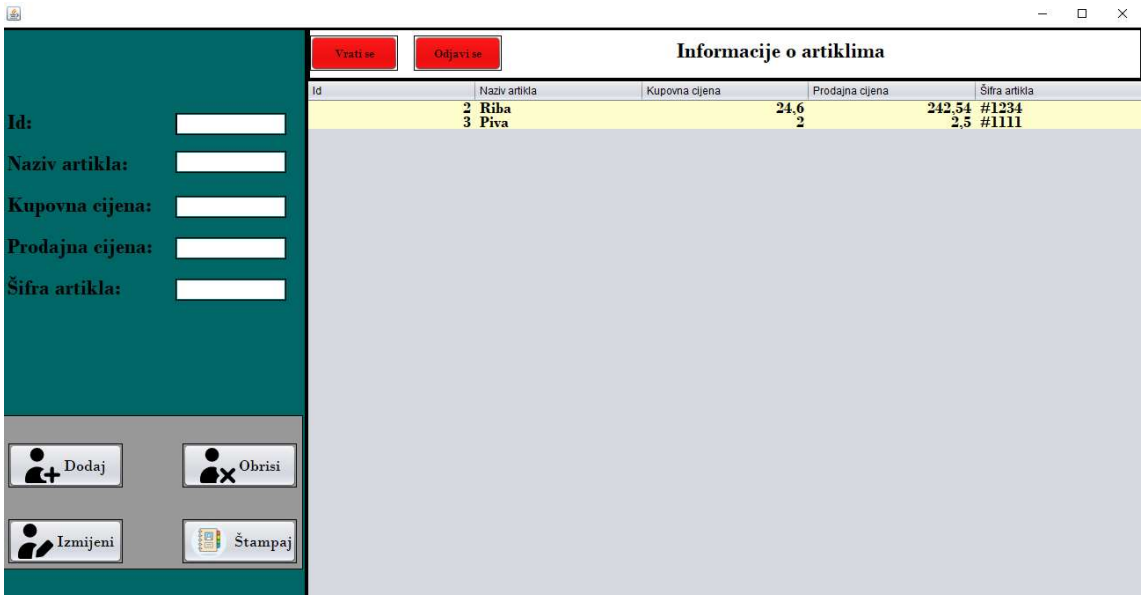
```
//Izvršenje SQL Query

public void executeSQLQuery(String query, String message) {
    Connection con = getConnection();
    Statement st;
    try{
        st = con.createStatement();
        if((st.executeUpdate(query)) == 1) {
            JOptionPane.showMessageDialog(null,"Podaci " + message+ " Uspiješno");
        } else {
            JOptionPane.showMessageDialog(null,"Podaci nisu " + message);
        }
        st.close();
        con.close();
    } catch(HeadlessException | SQLException e){
            JOptionPane.showMessageDialog(null,e);

    }
}
```

4.4 Metod executeSQLQuery koji izvršava operaciju i ispisuje poruku

Izgled GUI ArtikliForm klase:



4.5. Izgled prozora

Obzirom da smo već deklarisali varijable i stvorili metod kako bi uzeli vrijednosti podataka iz tabele potrebno je napisati još dva metoda. Prvi koji stvara niz u koji će se ispisivati vrijednosti tabele artikli i drugi pomoću kojeg će se isti ti podaci prikazivati u tabeli.

```
public ArrayList<Artikli> getArtikliList() {
    ArrayList<Artikli> artikliList = new ArrayList<Artikli>();
    Connection connection = getConnection();

    String query ="SELECT * FROM artikli";

    try{
        st = connection.createStatement();
        rs = st.executeQuery(query);
        Artikli artikli;
        while (rs.next()){
            artikli = new Artikli(rs.getInt("Id"),rs.getString("Naziv_artikla"),
            rs.getFloat("Kupovna_cijena"),rs.getFloat("Prodajna_Cijena"),rs.getString("Sifra_Artikla"));
            artikliList.add(artikli);
        }
    }catch(SQLException e){
        JOptionPane.showMessageDialog(null, e);
    }
    return artikliList;
}
```

4.6. Metod koji vraća niz podataka iz baze

```
//Prikazivanje podataka u JTable
public void prikaziUTabeli(){
    ArrayList<Artikli> list = getArtikliList();
    DefaultTableModel model = (DefaultTableModel)Tabla.getModel();
    Object [] row = new Object[5];
    for(int i = 0; i<list.size();i++) {
        row[0] = list.get(i).getId();
        row[1] = list.get(i).getNazivArtikla();
        row[2] = list.get(i).getKupovnaCijena();
        row[3] = list.get(i).getProdajnaCijena();
        row[4] = list.get(i).getSifraArtikla();

        model.addRow(row);

    }
```

4.7. Metod koji će prikazivati podatke u tabeli

Obzirom da je sve sada spremno mogu se napraviti akcije pomoću koji će se podaci ubacivati (INSERT), mijenjati (UPDATE) i brisati (DELETE).

Dakle zahvaljujući metodu executeSQLQuery unosimo parametre String upita na bazu i poruku koju želimo da piše ukoliko je akcija uspiješno izvršena.

Prilikom selektovanja reda u tabeli, a da bi olakšali akciju izmjene vrijednosti podataka urađen je sledeći JTable akcija na Mouse Clicked:

```
private void TablaMouseClicked(java.awt.event.MouseEvent evt) {
    // Prikazivanja podataka selektovanog reda iz tabele u odgovarajuća polja
    int i = Tabla.getSelectedRow();
    TableModel model = Tabla.getModel();
    txtId.setText(model.getValueAt(i,0).toString());
    txtIme.setText(model.getValueAt(i,1).toString());
    txtKupovna.setText(model.getValueAt(i,2).toString());
    txtProdajna.setText(model.getValueAt(i,3).toString());
    txtSifra.setText(model.getValueAt(i,4).toString());
```

4.6. Prikazivanje podataka selektovanog reda u odgovarajuća polja

Na kraju ostala je samo akcija na Button predviđen za štampanje podataka iz Tabele. Uvođenjem

- import java.util.logging.Level;
- import java.util.logging.Logger;
- import java.text.MessageFormat;

pišemo sledeće:

```
// Akcija za štampanje
try {
    MessageFormat header = new MessageFormat("----- Podaci o korisnicima smeštaja ----");
    MessageFormat footer = new MessageFormat("Desktop aplikacija za knjigovodstvo");
    Tabla.print(JTable.PrintMode.FIT_WIDTH,header,footer);
}
catch(PrinterException e){
    Logger.getLogger(ArtikliForm.class.getName()).log(Level.SEVERE,null,e);
}
```

4.7. Akcija za štampanje

Gotovo identičnim postupkom je odrađena većina klasa, shodno tome u nastavku one neće biti opisane.

4.4. FORMA ZA PRIKAZ TROŠKOVA

GUI klasa TroskoviForm se sastoji od pet JPanel sekcija.

Prva sekcija pored naslova sadrži dva Button elemente sa metodama sa povratak na početnu stranu i odjave na stranu prijave.

Druga sekcija predstavlja informacije o radnicima sa dva JTextField-a, Label poljem koje je korišteno za naslov i dva koje pomažu korisniku da zna koji podatak se gdje traži. Ova polja su postavljena na nazivPolja.setEditable(false); kako bi se onemogućio unos bilo kakvog teksta. Njihova primarna uloga jeste ispis podataka o ukupnom broju radnika i o ukupnom trošku zasnovan na platama radnika.

```
//Ukupni troškovi plata radnika

public void UkupniTroskoviPlate() {
    float suma = 0;
    try{
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/desktopAplikacija?serverTimezone=UTC","root","MMOLSX268++-");
        st = con.createStatement();
        rs = st.executeQuery("SELECT SUM(Plata) FROM radnici");
        while(rs.next()) {
            float a = rs.getFloat(1);
            suma = suma + a;

            String poruka = Float.toString(suma);
            txtTroskovi.setText(poruka);

        }
    }
    catch(Exception e) {
        JOptionPane.showMessageDialog(null,e,"Poruka!",JOptionPane.ERROR_MESSAGE);
    }
}
```

4.8. Metod za prikaz ukupnog troška zasnovanog na plate radnicima

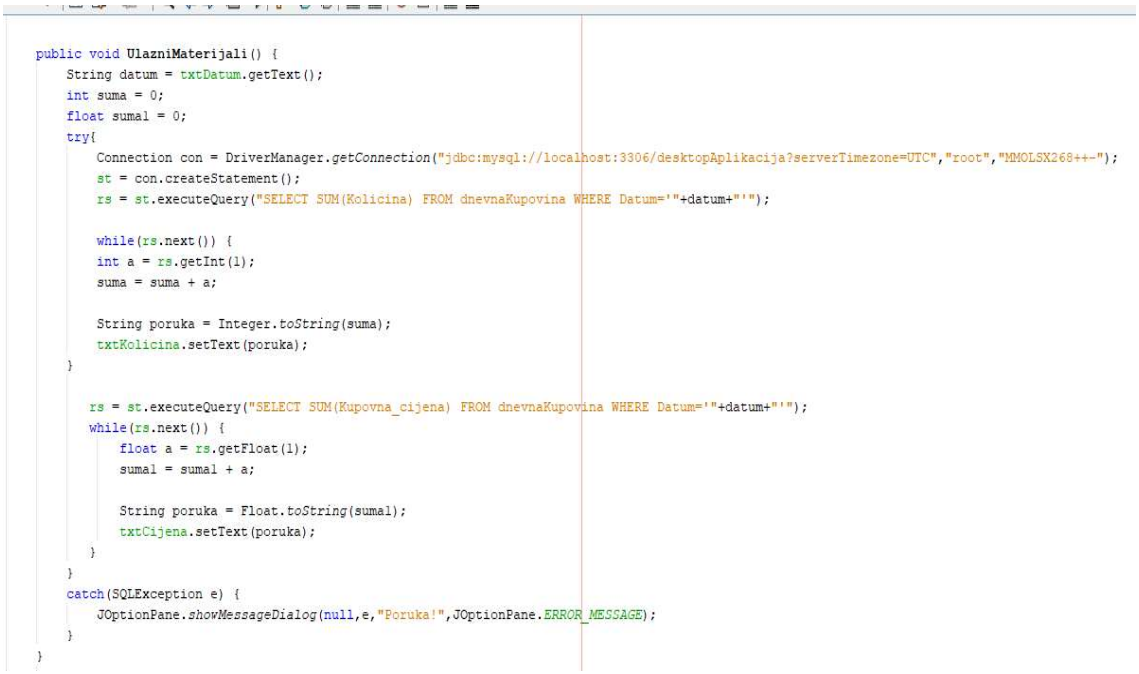
Metod sa slike 4.8. koristi funkciju SUM koja sabira kolonu plata,a korištenjem while petlje izvlačimo vrijednost koja je deklarisana sa tipom podatka float, odnosno decimalnim tipom. Ako se poveća broj redova,automatski će se povećati i vrijednost sume. Na kraju je samo potrebno prebaciti vrijednost float u vrijednost String pomoću ključnih riječi Float.toString() i pomoću setter-a postaviti vrijednost u naznačeno polje.

```
//Ukupan broj radnika
    public void UkupanBrojRadnika() {
        try {
            Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/desktopAplikacija?serverTimezone=UTC","root","MMOLSX268++-");
            st = con.createStatement();
            rs = st.executeQuery("select MAX(Id) AS Max_Id from radnici");
            while (rs.next()) {
               int a = rs.getInt(1);

               String broj = Integer.toString(a);
               txtBroj.setText(broj);
                }
            rs.close();
            st.close();
            con.close();
        }
        catch(SQLException e) {
            JOptionPane.showMessageDialog(null,e,"Poruka!",JOptionPane.ERROR_MESSAGE);
        }
    }
```

4.9. Metoda za prikaz najvećeg Id table radnici

Sledeća sekcija je identična drugoj s tim što je prvo polje datum slobodno za pisanje.Elem prilikom unosa određenog datuma metod uzima vrijednost iz tabele korištenjem ključne riječi WHERE, a koji se piše prilikom upita na bazu i ispisuje na drugo polje količinu i cijenu troškova nabavke toga dana. Pomenuta dva polja se ne mogu mijenjati.



```
public void UlazniMaterijali() {
    String datum = txtDatum.getText();
    int suma = 0;
    float sumal = 0;
    try{
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/desktopAplikacija?serverTimezone=UTC","root","MMOLSX268++-");
        st = con.createStatement();
        rs = st.executeQuery("SELECT SUM(Kolicina) FROM dnevnaKupovina WHERE Datum='"+datum+"'");

        while(rs.next()) {
        int a = rs.getInt(1);
        suma = suma + a;

        String poruka = Integer.toString(suma);
        txtKolicina.setText(poruka);
    }

       rs = st.executeQuery("SELECT SUM(Kupovna_cijena) FROM dnevnaKupovina WHERE Datum='"+datum+"'");
       while(rs.next()) {
           float a = rs.getFloat(1);
           sumal = sumal + a;

           String poruka = Float.toString(sumal);
           txtCijena.setText(poruka);
       }
    }
    catch(SQLException e) {
        JOptionPane.showMessageDialog(null,e,"Poruka!",JOptionPane.ERROR_MESSAGE);
    }
}
```

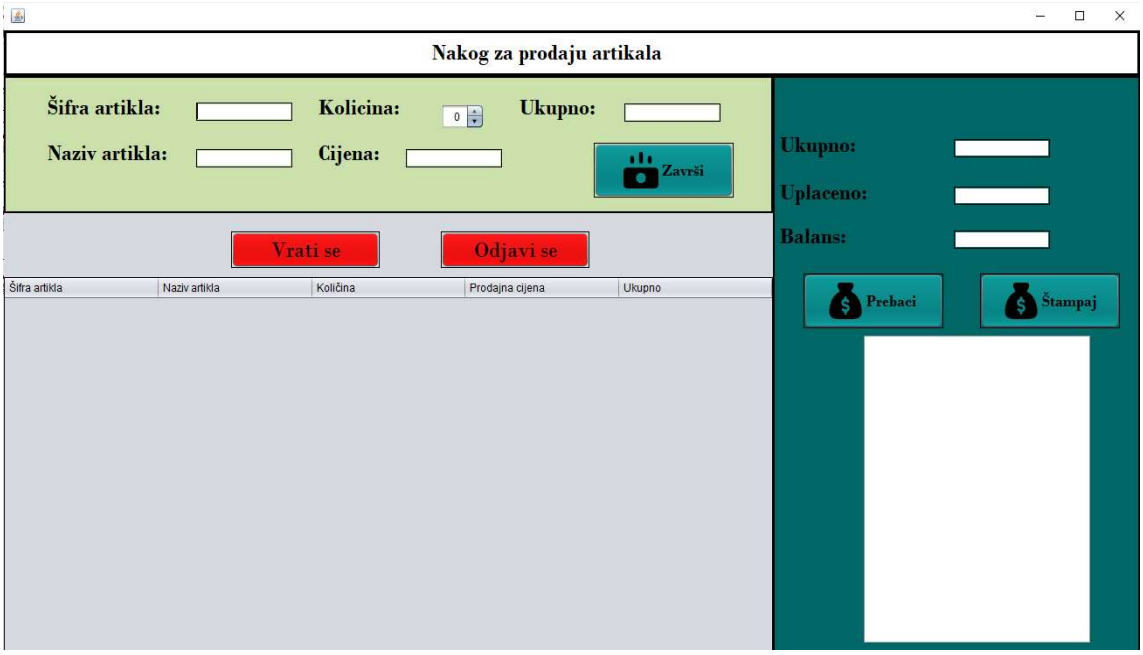
5.0. Metod sa dva upita

Obzirom da ova akcija zahtijeva unos i potom pritisak tastera ENTER sa tastature, mora se dati doznanja sa IF-uslovljavanjem kada će se zapis u poljima pojaviti, odnosno kada će se izvršiti upit na bazu. Postižemo to preko Event/Key/KeyPressed i ključnim riječima:

```
if(evt.getKeyCode() == KeyEvent.VK_ENTER){
        UlazniMaterijali();
    }
```

4.5. NALOG ZA PRODAJU

Ova funkcija nije pripadajuća usmjerenju knjigovodstva, ali se može izvesti preko stvorenih tabela. GUI klasa NalogZaProdajuForm predstavlja funkciju desktop aplikacije za uzimanja vrijednosti iz tabele artikli i izdavanje računa prilikom prodaje određene količine.



5.1. Izgled grafičkog prozora GUI klase NalogZaProdajuForm

Što se tiče grafičkog prikaza I dva standardna Button-a za povratak na početnu i prozor prijave, ova interfejs sadrži i sekciju sa četiri JTextField-a u kojem se kod Šifre artikla ili naziva artikla može unijeti kod jednog od njih sa tastature vrijednost, dok će u drugom i polju predviđenom za predstavljanje cijene izađi vrijednost prodajne cijene zapisanog u bazi podataka. Povećanjem preko komponente Spinner povećavaće se i ukupna vrijednost prikaza na predviđenom polju. Button sa natpisom Završi unijeće podatke u tabelu i prikazati ukupnu cijenu u sekciju desno. Povećavanjem cijene preko različitih artikala na tom polju prikazivaće se ukupna cijena, odnosno suma čitave kolone Ukupno iz tabele. Dodavanjem u polje uplaćeno i pritiskom na taster ENTER izaće razlika koja će se prikazati u polju za Balans. Button sa tekstom Prebaci uzima vrijednosti iz tabele i prijednosti iz polja sekcije prebacujući ih u JTextArea koja će prikazivati izgled računa. Akcija štampaj se postiže preko Button sa tekstom Štampaj.

```
    // Polje unosa za šifru artikla
    if(evt.getKeyCode() == KeyEvent.VK_ENTER) {
        String sifra = txtSifra.getText();

        try{
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/desktopAplikacija?serverTimezone=UTC","root","MMOLSX268++-");
        ps = con.prepareStatement("SELECT * FROM artikli WHERE Sifra_artikla = ?");

        ps.setString(1, sifra);
        rs = ps.executeQuery();

        if(rs.next() == false){
            JOptionPane.showMessageDialog(null,"Artikal nije pronaden","Poruka!",JOptionPane.ERROR_MESSAGE);
        }
        else {
            String nazivArtikla = rs.getString("Naziv_artikla");
            String prodajnaCijena = rs.getString("Prodajna_cijena");

            txtNaziv.setText(nazivArtikla.trim());
            txtCijena.setText(prodajnaCijena.trim());
        }

        }catch(SQLException e) {
          JOptionPane.showMessageDialog(null,"Došlo je do greške!","Poruka!",JOptionPane.ERROR_MESSAGE);
        }
    }
```

5.2. Metod koji odgovara akciji KeyPressed na polju za šifru

```
    private void txtKolicinaStateChanged(javax.swing.event.ChangeEvent evt) {
        // TODO add your handling code here:
        int kolicina = Integer.parseInt(txtKolicina.getValue().toString());
        float cijena = Float.parseFloat(txtCijena.getText());

        float ukupno = kolicina * cijena;
        txtUkupno.setText(String.valueOf(ukupno));
    }

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        // Upis u tabelu
        float cijena = Float.parseFloat(txtCijena.getText());
        int kolicina = (int) txtKolicina.getValue();
        float ukupno = kolicina * cijena;
        DefaultTableModel model = new DefaultTableModel();
        model = (DefaultTableModel)Tabla.getModel();
        model.addRow(new Object[] {txtSifra.getText(),txtNaziv.getText(),kolicina,cijena,ukupno});

        //Ispis na polju Ukupno
        finalSuma = finalSuma + ukupno;
        String sumaIspis = String.valueOf(finalSuma);
        txtUkupno1.setText(sumaIspis);
```

5.3. Metodi

Slika 5.3. predstavlja dva metoda. Prvi metod uzima vrijednosti iz polja Spinner i polja cijene i vrši aritmetičku operaciju koja će omogućiti promjenu na polju ukupno svaki put kada se poveća količina artikla.

Drugi metod upisuje u tabelu i na kraju ispisuje u polje ukupno sa desne strane.

Sledeće što je potrebno jeste uzeti pomoću ključne riječi getText() vrijednost iz polja uplaćeno i izvršiti aritmetičku operaciju oduzimanja. Dobijeni rezultat ispisati na polju balans.

Pored funkcije za štampanje potrebno je prebaciti rezultate iz tabele u textArea kao i vrijednosti Ukupno, Uplaćeno i Balanas. Pored toga postaviti tekst za zaglavlje i podnožije

```
public void racun() {
    String ukupno = txtUkupno.getText();
    String uplaceno = txtUplaceno.getText();
    String balans = txtBilans.getText();

    DefaultTableModel model = new DefaultTableModel();

    model = (DefaultTableModel)Tabla.getModel();
    //Zaglavlje racuna
    txtRacun.setText(txtRacun.getText()+    "*****************************************************"+"\n"
                                           +"*****************************************************"+"\n"
                                           +""+"\n"
                                           +"             Ugostiteljska radnja:                "+"\n"
                                           +"             Adresa:                              "+"\n"
                                           +"             Broj telefona:                       "+"\n"
                                           +""+"\n"
                                           +"*****************************************************"+"\n"
                                           +"*****************************************************"+"\n"
                                           +""+"\n" );
    txtRacun.setText(txtRacun.getText() + "Artikal" + "  "+ "Kolicina" + "  "+"Cijena" + "  "+"Ukupno" + "\n");
    //Uzimanje podataka iz tabele preko reda
    for (int i = 0; i<model.getRowCount(); i++) {
    //Deklarisanje dobijenih vrijednosti iz reda
        String artikal = model.getValueAt(i,1).toString();
        String kolicina = model.getValueAt(i,2).toString();
        String cijena = model.getValueAt(i,3).toString();
        String ukupnaCijena = model.getValueAt(i,4).toString();
        // Ispis
        txtRacun.setText(txtRacun.getText() + artikal + "\t " + kolicina + "\t " + cijena + "\t " + ukupnaCijena + "\n");

    }
```

5.4.1. Zaglavlje i uzimanje vrijednosti iz tabele

```
}
txtRacun.setText(txtRacun.getText()+"\n");
txtRacun.setText(txtRacun.getText()+"*************************************************"+"\n");
//Ispis vrijednosti deklarisanih na početku
txtRacun.setText(txtRacun.getText() + "Ukupno" +"\t" + "Uplaćeno" + "\t" + "Balans" + "\n");
txtRacun.setText(txtRacun.getText() + ukupno +"\t " + uplaceno + "\t " +  balans + "\n");

//Footer
txtRacun.setText(txtRacun.getText() +   "*************************************************"+"\n"
                                        +"*************************************************"+"\n"
                                        +""+"\n"
                                        +"             Hvala na posjeti!                 "+"\n"
                                        +"             Diznajnirao Jovan Aulić           "+"\n");
```

5.4.2. Podnožije i uzimanje teksta iz polja Ukupno,Uplaceno i Balans

ZAKLJUČAK

Tema mog seminarskog rada jeste Izrada desktop aplikacije “Desktop aplikacija za knjigovodstvo u ugostiteljskim objektima” u programskom jeziku Java. Aplikacija se sastoji od tri paketa u kojima su smještene jedna izvršna main metoda, jedanaest GUI klasa i sedam Java klasa i osam tabela svrstanih u bazu podatak “desktopAplikacija”.

LITERATURA

(n.d.). Retrieved from youtube: www.youtube.com

Java. (n.d.). Retrieved from wikipedia.

Java JDBC Tutorial. (n.d.). Retrieved from javapoint.

MySQL Java tutorial. (2020, July 6). Retrieved from ZetCode: www.zetcode.com

Salapura, S. (n.d.). *Java*. Retrieved from salapura: www.salapura.com/Java