9. Deveta laboratorijska vježba

9.1. JDBC

Svrha laboratorijske vježbe korištenje baze podataka umjesto datoteka od strane JavaFX aplikacije kroz operacije dohvaćanja i spremanja podataka. Funkcionalnosti aplikacije koje treba uključivati rješenje laboratorijske vježbe prikazan je na sljedećem YouTube videu: https://www.youtube.com/watch?v=J-Kzpf_E_Sw.

9.2. Zadatak za pripremu

Nastaviti razvoj aplikacije iz osme laboratorijske vježbe i implementirati podršku za rad s bazom podataka prema koracima u nastavku.

1. Kopirati projekt iz osme laboratorijske vježbe i preimenovati ga na način da mu se promijeni redni broj u „9".
2. U „pom.xml" dodati zavisnost o ispravnoj verziji *drivera* za H2 bazu podataka.
3. Kreirati datoteku „bazaPodataka.properties" i u njoj dodati parove „ključ-vrijednost" za URL baze podataka, korisničko ime i lozinku, npr.

```
#Osnovni podaci za spajanje na bazu podataka
bazaPodatakaUrl = jdbc:h2:tcp://localhost/~/JavaFX-2015

#Podaci za pristupanje bazi podataka
korisnickoIme = student
lozinka = student
```

4. Kreirati bazu podataka prema podacima u datoteci „bazaPodataka.properties" i u njoj izvršiti sljedeću skriptu koja će kreirati potrebne tablice i relacije:

```
CREATE SCHEMA RACUNALA;

CREATE TABLE RACUNALA.KOMPONENTA(
ID INT NOT NULL GENERATED ALWAYS AS IDENTITY,
NAZIV_PROIZVODJACA VARCHAR(50) NOT NULL,
CIJENA DECIMAL(8,2) NOT NULL,
PRIMARY KEY (ID)
);

CREATE TABLE RACUNALA.MATICNA_PLOCA(
ID INT NOT NULL,
TIP VARCHAR(50) NOT NULL,
TIP_SUCELJA_ZA_PROCESOR VARCHAR(50) NOT NULL,
MAX_MEMORIJSKIH_MODULA INT NOT NULL,
PRIMARY KEY (ID),
FOREIGN KEY (ID) REFERENCES RACUNALA.KOMPONENTA(ID)
);

CREATE TABLE RACUNALA.PROCESOR(
ID INT NOT NULL,
TIP VARCHAR(50) NOT NULL,
```

```
TIP_SUCELJA VARCHAR(50) NOT NULL,
BRZINA DECIMAL(4,2) NOT NULL,
PRIMARY KEY (ID),
FOREIGN KEY (ID) REFERENCES RACUNALA.KOMPONENTA(ID)
);

CREATE TABLE RACUNALA.RADNA_MEMORIJA(
ID INT NOT NULL,
TIP VARCHAR(50) NOT NULL,
KAPACITET INT NOT NULL,
PRIMARY KEY (ID),
FOREIGN KEY (ID) REFERENCES RACUNALA.KOMPONENTA(ID)
);

CREATE TABLE RACUNALA.TVRDI_DISK(
ID INT NOT NULL,
TIP VARCHAR(50) NOT NULL,
KAPACITET DECIMAL(6,2) NOT NULL,
PRIMARY KEY (ID),
FOREIGN KEY (ID) REFERENCES RACUNALA.KOMPONENTA(ID)
);

CREATE TABLE RACUNALA.RACUNALO(
ID INT NOT NULL GENERATED ALWAYS AS IDENTITY,
MATICNA_PLOCA_ID INT NOT NULL,
PROCESOR_ID INT NOT NULL,
RADNA_MEMORIJA_ID INT NOT NULL,
BROJ_MODULA_RADNE_MEMORIJE INT NOT NULL,
TVRDI_DISK_ID INT NOT NULL,
PRIMARY KEY (ID),
FOREIGN KEY (MATICNA_PLOCA_ID) REFERENCES RACUNALA.MATICNA_PLOCA(ID),
FOREIGN KEY (PROCESOR_ID) REFERENCES RACUNALA.PROCESOR(ID),
FOREIGN KEY (RADNA_MEMORIJA_ID) REFERENCES RACUNALA.RADNA_MEMORIJA(ID),
FOREIGN KEY (TVRDI_DISK_ID) REFERENCES RACUNALA.TVRDI_DISK(ID)
);
```

5. Kreirati novi paket „hr.java.vjezbe.baza.podataka" i u njemu klasu „BazaPodataka".
6. Unutar klase „BazaPodataka" kreirati privatnu metodu „spajanjeNaBazuPodataka" koja ne prima nikakve parametre, a vraća objekt tipa „Connection" koji predstavlja vezu na bazu podataka. Prilikom kreiranja veze potrebno je koristiti podatke iz datoteke „bazaPodataka.properties". Osim metode za spajanje na bazu podataka potrebno je kreirati i privatnu metodu za zatvaranje veze s bazom podataka, npr. „zatvaranjeVezeNaBazuPodataka", koja prima objekt tipa „Connection" i zatvara vezu koju on predstavlja. Te metode će se morati pozivati prije i nakon izvršavanja upita nad bazom podataka.
7. U klase „Komponenta" i „Racunalo" dodati novo cjelobrojno polje „id" koje će sadržavati identifikator tih entiteta. Osim toga generirati „getter" i „setter" metode za „id".
8. Unutar klase „BazaPodataka" za svaku komponentu napisati metodu koja služi za spremanje podataka u bazu. Primjer programskog koda za spremanje nove matične ploče u bazu podataka izgleda ovako:

```
public static void spremiMaticnuPlocu(MaticnaPloca maticnaPloca) throws
Exception {
      Connection veza = spajanjeNaBazuPodataka();
      veza.setAutoCommit(false);

      try {
            PreparedStatement insertMaticnaKomponenta = veza
                  .prepareStatement("INSERT INTO RACUNALA.KOMPONENTA
(NAZIV_PROIZVODJACA, CIJENA) VALUES (?, ?)");
            insertMaticnaKomponenta.setString(1,
                  maticnaPloca.getNazivProizvodjaca());
            insertMaticnaKomponenta.setBigDecimal(2,
                  maticnaPloca.getCijena());
            insertMaticnaKomponenta.executeUpdate();

            ResultSet generatedKeys =
                  insertMaticnaKomponenta.getGeneratedKeys();
            if (generatedKeys.next()) {
                  maticnaPloca.setId(generatedKeys.getInt(1));
            }

            PreparedStatement insertMaticnaPloca = veza
                 .prepareStatement("INSERT INTO RACUNALA.MATICNA_PLOCA (ID,
TIP, TIP_SUCELJA_ZA_PROCESOR, MAX_MEMORIJSKIH_MODULA) VALUES (?, ?, ?,
?)");
            insertMaticnaPloca.setInt(1, maticnaPloca.getId());
            insertMaticnaPloca.setString(2, maticnaPloca.getTip());
            insertMaticnaPloca.setString(3,
                  maticnaPloca.getTipSuceljaZaProcesor());
            insertMaticnaPloca.setInt(4,
            maticnaPloca.getMaksimalanBrojMemorijskihModula());

            insertMaticnaPloca.executeUpdate();

            veza.commit();
     }
     catch(Throwable ex) {
          veza.rollback();
          throw ex;
     }

     zatvaranjeVezeNaBazuPodataka(veza);
}
```

Na sličan način treba implementirati i metode za spremanje svih ostalih komponenata u bazu podataka.

9. U klasi „BazaPodataka" za svaku komponentu kreirati metodu za dohvat svih podataka iz tablice u bazi podataka i vratiti listu. Primjer metode koja dohvaća sve podatke o matičnim pločama prikazan je u nastavku:

```
public static List<MaticnaPloca> dohvatiMaticnePloce() throws Exception {
            Connection veza = spajanjeNaBazuPodataka();
```

```
            Statement statementKomponenta = veza.createStatement();
            ResultSet resultSetKomponenta =
statementKomponenta.executeQuery("SELECT * FROM RACUNALA.KOMPONENTA");
            PreparedStatement statementMaticnaPloca =
veza.prepareStatement("SELECT * FROM RACUNALA.MATICNA_PLOCA WHERE ID = ?");
            List<MaticnaPloca> listaMaticnihPloca = new ArrayList<>();

            while(resultSetKomponenta.next()) {
                  Integer id = resultSetKomponenta.getInt("ID");
                  String nazivProizvodjaca =
resultSetKomponenta.getString("NAZIV_PROIZVODJACA");
                  BigDecimal cijena =
resultSetKomponenta.getBigDecimal("CIJENA");
                  statementMaticnaPloca.setInt(1, id);
                  ResultSet resultSetMaticnaPloca =
statementMaticnaPloca.executeQuery();
                  while(resultSetMaticnaPloca.next()) {
                        String tip =
resultSetMaticnaPloca.getString("TIP");
                        String tipSucelja =
resultSetMaticnaPloca.getString("TIP_SUCELJA_ZA_PROCESOR");
                        Integer maxModula =
resultSetMaticnaPloca.getInt("MAX_MEMORIJSKIH_MODULA");
                        MaticnaPloca maticnaPloca = new
MaticnaPloca(nazivProizvodjaca, tip, tipSucelja, maxModula, cijena);
                        maticnaPloca.setId(id);
                        listaMaticnihPloca.add(maticnaPloca);
                  }
            }

            zatvaranjeVezeNaBazuPodataka(veza);

            return listaMaticnihPloca;
      }
```

Te metode potrebno je pozivati u klasi „PocetniEkranController" umjesto metoda koje podatke dohvaćaju iz datoteka.

10. U klasu „BazaPodataka" dodati metodu koja će služiti za spremanje podataka o konfiguraciji te tu metodu povezati s gumbom koji je do sada spremao podatke u datoteku.

MOGUĆNOST UNAPREĐENJA ZADATKA:

1. Umjesto da se koristi „VARCHAR" tip podatka za podatak „sučelje" koji imaju procesor i matična ploča, uvesti novu tablicu u bazi podataka s nazivom „SUČELJE" koja će sadržavati „šifarnik" različitih sučelja.

NAPOMENA:

1. Svi detalji koji nisu definirani potrebno je proizvoljno implementirati.
2. Na svim mjestima na kojima su se koristile datoteke u osmoj vježbi treba zamijeniti s

bazom podataka u devetoj laboratorijskoj vježbi.