

9. Deveta laboratorijska vježba

9.1. Tema vježbe

Svrha laboratorijske vježbe je zamjena datoteka s relacijskom bazom podataka koja će sadržavati podatke koje aplikacija koristi.

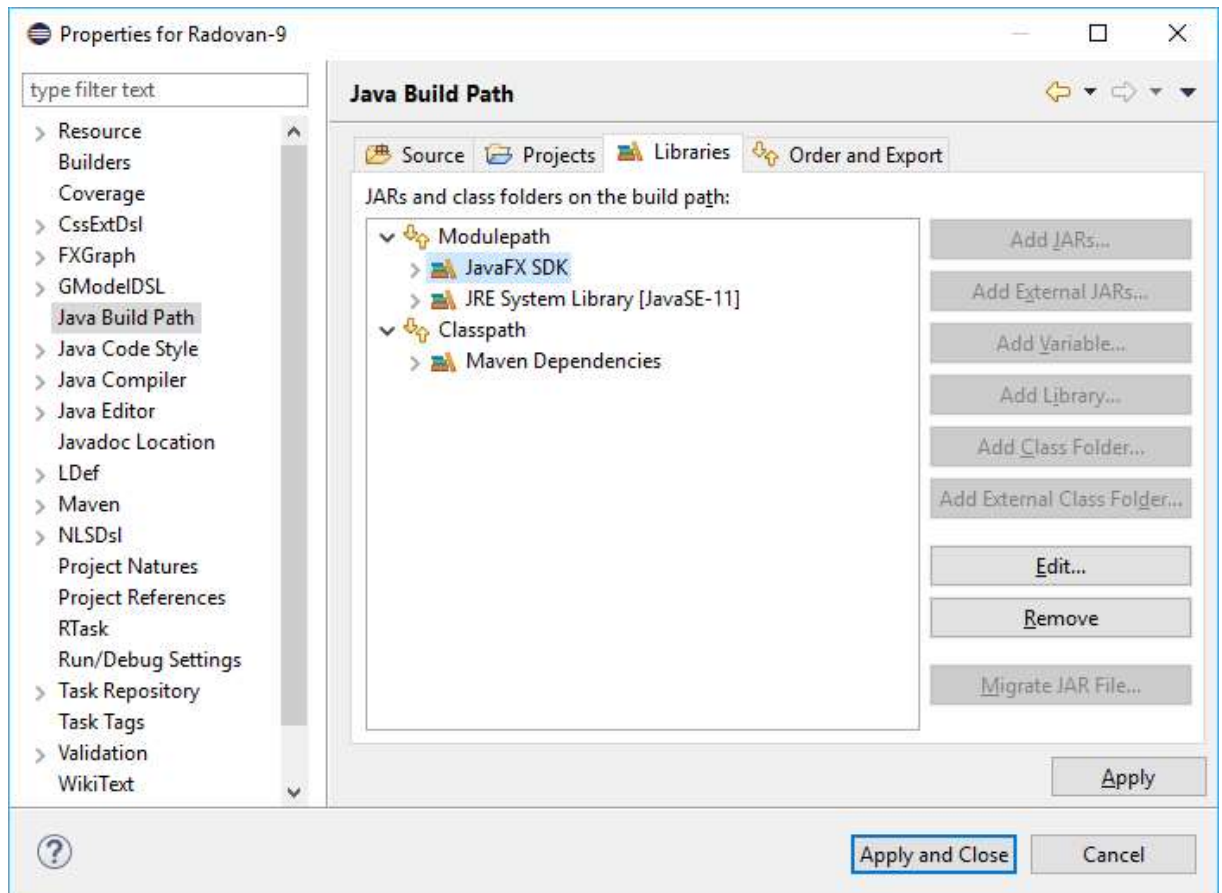
9.2. Zadatak za pripremu

Proširiti rješenje osme laboratorijske vježbe na način kako je opisano u sljedećim koracima:

1. Instalirati novu inačicu Eclipse razvojnog okruženja pod nazivom „Eclipse IDE for Java Developers” s poveznice <https://www.eclipse.org/downloads/packages/release/2018-12/r/eclipse-ide-java-developers>. Osim toga je potrebno instalirati „e(fx)clipse” *plugin* za Eclipse kako je prikazano na predavanjima.
2. Kopirati projekt iz osme laboratorijske vježbe te ga preimenovati da sadrži broj „9” umjesto broja „8”.
3. Korištenjem opcije „Java Build Path” unutar izbornika „Properties” na razini projekta unutar razvojnog okruženja Eclipse postaviti inačicu Jave na Javu 11 te razinu prevoditelja (korištenjem opcije „Java Compiler”) također na razinu „11”. Nakon toga će se pojaviti pogreške kod prevođenja radi toga što JavaFX nije uključena u distribuciju Jave 11 i potrebno je dodati nove ovisnosti kroz Maven alat.
4. Unutar „pom.xml” datoteke potrebno je sljedeće „dependency” zapise:

```
<dependency>
  <groupId>org.openjfx</groupId>
  <artifactId>javafx-controls</artifactId>
  <version>11</version>
</dependency>
<dependency>
  <groupId>org.openjfx</groupId>
  <artifactId>javafx-fxml</artifactId>
  <version>11</version>
</dependency>
```

5. Korištenjem opcije „Java Build Path” potrebno je maknuti „JavaFX SDK” pritiskom na gumb „Remove” kao što je prikazano na slici 1.



Slika 1. Prikaz „Java Build Path“ konfiguracije projekta

6. Korištenjem stranice <https://gluonhq.com/products/javafx/> preuzeti arhivu „JavaFX Windows SDK“ (kao što je prikazano na slici 2) te je raspakirati.

JavaFX

The JavaFX 11 runtime is available as a platform-specific SDK, as a number of jmods, and as a set of artifacts in maven central.

The OpenJFX page at openjfx.io is a great starting place to learn more about JavaFX 11.

The Release Notes for JavaFX 11 are available in the OpenJFX GitHub repository: [Release Notes](#).

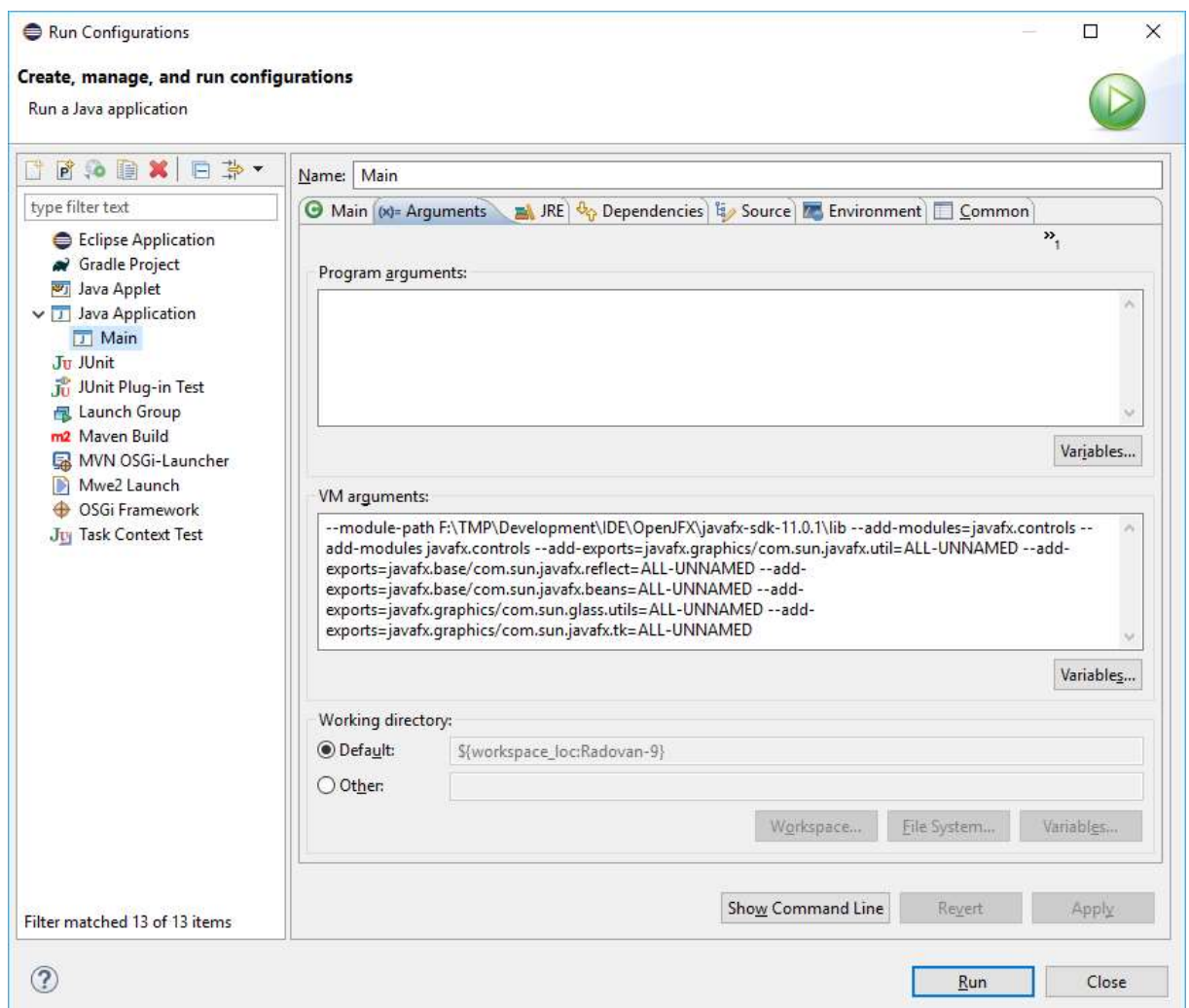
This software is licensed under GPL v2 + Classpath (see <http://openjdk.java.net/legal/gplv2+ce.html>).

Product	Version	Platform	Download
JavaFX Windows SDK	11.0.1	Windows	Download [SHA256]
JavaFX Windows jmods	11.0.1	Windows	Download [SHA256]
JavaFX Mac OS X SDK	11.0.1	Mac	Download [SHA256]
JavaFX Mac OS X jmods	11.0.1	Mac	Download [SHA256]
JavaFX Linux SDK	11.0.1	Linux	Download [SHA256]
JavaFX Linux jmods	11.0.1	Linux	Download [SHA256]

Slika 2. Stranica za preuzimanje JavaFX Windows SDK okruženja

7. Pokrenuti aplikaciju korištenjem opcije „Run As -> Java Application“ kako bi se kreirala „Run Configuration“ konfiguracija (bez obzira što kod pokretanja Eclipse može dojaviti pogrešku). Nakon toga korištenjem opcije „Run As -> Run Configurations...“ otvoriti dijalog za kreiranje konfiguracije za pokretanje aplikacije, otvoriti „Arguments“ tab i u polje „VM arguments“ (slika 3) postaviti sljedeće parametre, pri čemu putanja do „lib“ mape unutar raspakirane mape JavaFX SDK okruženja (iz šestog koraka) mora biti postavljena kako bi odgovarala lokaciji na lokalnom disku:

```
--module-path F:\TMP\Development\IDE\OpenJFX\javafx-sdk-11.0.1\lib --add-  
modules=javafx.controls --add-modules javafx.controls --add-  
exports=javafx.graphics/com.sun.javafx.util=ALL-UNNAMED --add-  
exports=javafx.base/com.sun.javafx.reflect=ALL-UNNAMED --add-  
exports=javafx.base/com.sun.javafx.beans=ALL-UNNAMED --add-  
exports=javafx.graphics/com.sun.glass.utils=ALL-UNNAMED --add-  
exports=javafx.graphics/com.sun.javafx.tk=ALL-UNNAMED
```



Slika 3. Konfiguracija za pokretanje JavaFX aplikacije

8. Sa stranica <http://www.h2database.com/html/download.html> preuzeti H2 bazu podataka verzije 1.4.196., raspakirati je i pokrenuti kako je prikazano na predavanjima.
9. Korištenjem web konzole za administraciju baze podataka kako je prikazano na predavanjima kreirati shemu baze prema sljedećoj SQL skripti:

```
CREATE TABLE STUDENT(  
id LONG NOT NULL GENERATED ALWAYS AS IDENTITY,  
ime VARCHAR(30) NOT NULL,  
prezime VARCHAR(30) NOT NULL,  
jmbag VARCHAR(15) NOT NULL,  
datum_rođenja DATE NOT NULL,  
PRIMARY KEY(id)  
);  
  
CREATE TABLE PROFESOR(  
id LONG NOT NULL GENERATED ALWAYS AS IDENTITY,  
ime VARCHAR(30) NOT NULL,  
prezime VARCHAR(30) NOT NULL,  
sifra VARCHAR(10) NOT NULL,  
titula VARCHAR(50) NOT NULL,  
PRIMARY KEY(id)  
);  
  
CREATE TABLE PREDMET(  
id LONG NOT NULL GENERATED ALWAYS AS IDENTITY,  
sifra VARCHAR(10) NOT NULL,  
naziv VARCHAR(50) NOT NULL,  
broj_ects_bodova SMALLINT NOT NULL,  
profesor_id SMALLINT NOT NULL,  
PRIMARY KEY(id),  
FOREIGN KEY(profesor_id) REFERENCES PROFESOR(id)  
);  
  
CREATE TABLE PREDMET_STUDENT(  
predmet_id LONG NOT NULL,  
student_id LONG NOT NULL,  
PRIMARY KEY(predmet_id, student_id),  
FOREIGN KEY(predmet_id) REFERENCES PREDMET(id),  
FOREIGN KEY(student_id) REFERENCES STUDENT(id)  
);  
  
CREATE TABLE ISPIT(  
id LONG NOT NULL GENERATED ALWAYS AS IDENTITY,  
predmet_id LONG NOT NULL,  
student_id LONG NOT NULL,  
ocjena SMALLINT NOT NULL,  
datum_i_vrijeme TIMESTAMP NOT NULL,  
PRIMARY KEY(id),  
FOREIGN KEY(predmet_id) REFERENCES PREDMET(id),  
FOREIGN KEY(student_id) REFERENCES STUDENT(id)  
);  
  
CREATE TABLE OBRAZOVNA_USTANOVA(  
id LONG NOT NULL GENERATED ALWAYS AS IDENTITY,  
naziv VARCHAR(50) NOT NULL,
```

```
PRIMARY KEY(id)
);

CREATE TABLE OBRAZOVNA_USTANOVA_ISPIT(
obrazovna_ustanova_id LONG NOT NULL,
ispit_id LONG NOT NULL,
PRIMARY KEY(obrazovna_ustanova_id, ispit_id),
FOREIGN KEY(obrazovna_ustanova_id) REFERENCES OBRAZOVNA_USTANOVA(id),
FOREIGN KEY(ispit_id) REFERENCES ISPIT(id)
);

CREATE TABLE OBRAZOVNA_USTANOVA_PROFESOR(
obrazovna_ustanova_id LONG NOT NULL,
profesor_id LONG NOT NULL,
PRIMARY KEY(obrazovna_ustanova_id, profesor_id),
FOREIGN KEY(obrazovna_ustanova_id) REFERENCES OBRAZOVNA_USTANOVA(id),
FOREIGN KEY(profesor_id) REFERENCES PROFESOR(id)
);

CREATE TABLE OBRAZOVNA_USTANOVA_PREDMET(
obrazovna_ustanova_id LONG NOT NULL,
predmet_id LONG NOT NULL,
PRIMARY KEY(obrazovna_ustanova_id, predmet_id),
FOREIGN KEY(obrazovna_ustanova_id) REFERENCES OBRAZOVNA_USTANOVA(id),
FOREIGN KEY(predmet_id) REFERENCES PREDMET(id)
);

CREATE TABLE OBRAZOVNA_USTANOVA_STUDENT(
obrazovna_ustanova_id LONG NOT NULL,
student_id LONG NOT NULL,
PRIMARY KEY(obrazovna_ustanova_id, student_id),
FOREIGN KEY(obrazovna_ustanova_id) REFERENCES OBRAZOVNA_USTANOVA(id),
FOREIGN KEY(student_id) REFERENCES STUDENT(id)
);

INSERT INTO PROFESOR(ime, prezime, sifra, titula) VALUES ('Ivan', 'Kovačević',
'P98765', 'Predavač');
INSERT INTO PROFESOR(ime, prezime, sifra, titula) VALUES ('Marija', 'Horvat',
'P2342', 'Viši predavač');
INSERT INTO PROFESOR(ime, prezime, sifra, titula) VALUES ('Marko', 'Župan',
'P72367', 'Profesor visoke škole');

INSERT INTO STUDENT(ime, prezime, jmbag, datum_rodjenja) VALUES ('Alen',
'Badalić', '024602348923', '1999-12-01');
INSERT INTO STUDENT(ime, prezime, jmbag, datum_rodjenja) VALUES ('Branko',
'Dalenović', '0036387651', '2000-06-11');
INSERT INTO STUDENT(ime, prezime, jmbag, datum_rodjenja) VALUES ('Josipa',
'Marić', '1108293842', '1999-02-28');
INSERT INTO STUDENT(ime, prezime, jmbag, datum_rodjenja) VALUES ('Ivana', 'Nikić',
'298347293', '2001-01-12');

INSERT INTO PREDMET(sifra, naziv, broj_ects_bodova, profesor_id) VALUES ('PR9283',
'Programiranje u jeziku Java', 6, 1);
INSERT INTO PREDMET(sifra, naziv, broj_ects_bodova, profesor_id) VALUES ('PR2138',
'Web aplikacije u Javi', 7, 2);
INSERT INTO PREDMET(sifra, naziv, broj_ects_bodova, profesor_id) VALUES
('PR98234', 'Nekonvencionalni računalni postupci', 6, 3);
```

```
INSERT INTO PREDMET_STUDENT(predmet_id, student_id) VALUES (1, 1);
INSERT INTO PREDMET_STUDENT(predmet_id, student_id) VALUES (1, 2);
INSERT INTO PREDMET_STUDENT(predmet_id, student_id) VALUES (1, 3);
INSERT INTO PREDMET_STUDENT(predmet_id, student_id) VALUES (2, 1);
INSERT INTO PREDMET_STUDENT(predmet_id, student_id) VALUES (2, 3);
INSERT INTO PREDMET_STUDENT(predmet_id, student_id) VALUES (3, 2);
INSERT INTO PREDMET_STUDENT(predmet_id, student_id) VALUES (3, 3);

INSERT INTO ISPIT(predmet_id, student_id, ocjena, datum_i_vrijeme) VALUES (1, 1,
5, parsedatetime('28-11-2018 17:00', 'dd-MM-yyyy hh:mm'));
INSERT INTO ISPIT(predmet_id, student_id, ocjena, datum_i_vrijeme) VALUES (1, 2,
4, parsedatetime('02-02-2018 18:00', 'dd-MM-yyyy hh:mm'));
INSERT INTO ISPIT(predmet_id, student_id, ocjena, datum_i_vrijeme) VALUES (1, 3,
3, parsedatetime('06-07-2018 15:00', 'dd-MM-yyyy hh:mm'));
INSERT INTO ISPIT(predmet_id, student_id, ocjena, datum_i_vrijeme) VALUES (2, 1,
4, parsedatetime('01-09-2018 18:00', 'dd-MM-yyyy hh:mm'));
INSERT INTO ISPIT(predmet_id, student_id, ocjena, datum_i_vrijeme) VALUES (2, 3,
5, parsedatetime('11-09-2018 16:00', 'dd-MM-yyyy hh:mm'));
INSERT INTO ISPIT(predmet_id, student_id, ocjena, datum_i_vrijeme) VALUES (3, 2,
4, parsedatetime('15-09-2018 17:00', 'dd-MM-yyyy hh:mm'));
INSERT INTO ISPIT(predmet_id, student_id, ocjena, datum_i_vrijeme) VALUES (3, 3,
5, parsedatetime('22-09-2018 18:00', 'dd-MM-yyyy hh:mm'));

INSERT INTO OBRAZOVNA_USTANOVA(naziv) VALUES ('Fakultet računarstva');
INSERT INTO OBRAZOVNA_USTANOVA(naziv) VALUES ('Tehničko veleučilište');
```

10. U „pom.xml“ dodati sljedeći „dependency“ koji će omogućiti korištenje H2 baze podataka i pripadajućih „drivera“ u Java implementaciji aplikacije:

```
<dependency>
  <groupId>com.h2database</groupId>
  <artifactId>h2</artifactId>
  <version>1.4.196</version>
</dependency>
```

11. Kreirati novi paket pod nazivom „**hr.java.vjezbe.baza**“ i u njemu novu klasu pod nazivom „**BazaPodataka**“. U toj klasi potrebno je implementirati statičke metode koje će podržavati operacije s bazom podataka kao što su, na primjer, dohvaćanje podataka o profesorima prema zadanim parametrima, dodavanje novih zapisa o profesorima itd. Osim toga je potrebno napisati privatnu statičku metodu „**spajanjeNaBazu**“ kako je objašnjeno na predavanjima uz korištenje „properties“ datoteke koja sadrži podatke o URL-u, korisničkom imenu te lozinki za pristup bazi podataka. Ta metoda mora vraćati objekt klase „**Connection**“. Te metode moraju u potpunosti zamijeniti korištenje podataka iz datoteka iz prošle laboratorijske vježbe. Osim toga je potrebno unutar paketa „**hr.java.vjezbe.iznimke**“ potrebno kreirati novu označenu iznimku „**BazaPodatakException**“ koja će se bacati iz klase „**BazaPodataka**“, a hvatati u „**Controller**“ klasama i ispisivati dijalog koji opisuje koja pogreška se dogodila. Primjer metode za pretragu podataka o profesorima i spremanje podataka o profesorima prikazan je u nastavku:

```
public static List<Profesor> dohvatiProfesorePremaKriterijima(Profesor profesor) throws
BazaPodatakaException {

    List<Profesor> listaProfesora = new ArrayList<>();

    try (Connection veza = spajanjeNaBazu()) {
        StringBuilder sqlUpit = new StringBuilder(
            "SELECT * FROM PROFESOR WHERE 1 = 1");

        if (Optional.ofNullable(profesor).isEmpty() == false) {

            if (Optional.ofNullable(profesor).map(
                Profesor::getId).isPresent()) {
                sqlUpit.append(" AND ID = " + profesor.getId());
            }

            if (Optional.ofNullable(profesor.getSifra()).map(
                String::isBlank).orElse(true) == false) {
                sqlUpit.append(" AND SIFRA LIKE '%" +
                    profesor.getSifra() + "%'");
            }

            if (Optional.ofNullable(profesor.getIme()).map(
                String::isBlank).orElse(true) == false) {
                sqlUpit.append(" AND IME LIKE '%" +
                    profesor.getIme() + "%'");
            }

            if (Optional.ofNullable(profesor.getPrezime()).map(
                String::isBlank).orElse(true) == false) {
                sqlUpit.append(" AND PREZIME LIKE '%" +
                    profesor.getPrezime() + "%'");
            }

            if (Optional.ofNullable(profesor.getTitula()).map(
                String::isBlank).orElse(true) == false) {
                sqlUpit.append(" AND TITULA LIKE '%" +
                    profesor.getTitula() + "%'");
            }
        }

        Statement upit = veza.createStatement();

        ResultSet resultSet = upit.executeQuery(sqlUpit.toString());

        while (resultSet.next()) {
```

```
        Long id = resultSet.getLong("id");
        String sifra = resultSet.getString("sifra");
        String ime = resultSet.getString("ime");
        String prezime = resultSet.getString("prezime");
        String titula = resultSet.getString("titula");
        Profesor noviProfesor = new Profesor(id, sifra, ime, prezime,
                                              titula);

        listaProfesora.add(noviProfesor);
    }
} catch (SQLException | IOException ex) {
    String poruka = "Došlo je do pogreške u radu s bazom podataka";
    logger.error(poruka, ex);
    throw new BazaPodatakaException(poruka, ex);
}

return listaProfesora;
}

public static void spremiNovogProfesora(Profesor profesor) throws
    BazaPodatakaException {
    try (Connection veza = spajanjeNaBazu()) {
        PreparedStatement preparedStatement = veza
            .prepareStatement(
                "INSERT INTO PROFESOR(ime, prezime, sifra, titula) VALUES (?, ?, ?, ?)");
        preparedStatement.setString(1, profesor.getIme());
        preparedStatement.setString(2, profesor.getPrezime());
        preparedStatement.setString(3, profesor.getSifra());
        preparedStatement.setString(4, profesor.getTitula());
        preparedStatement.executeUpdate();
    } catch (SQLException | IOException ex) {
        String poruka = "Došlo je do pogreške u radu s bazom podataka";
        logger.error(poruka, ex);
        throw new BazaPodatakaException(poruka, ex);
    }
}
```

U slučajevima kad se iz baze podataka dohvaća samo identifikator entiteta, a potrebno je koristiti odgovarajući objekt koji pripada tom identifikatoru, u tom slučaju je potrebno koristiti novi upit kojim će se dohvatiti cijeli objekt koji odgovara tom identifikatoru.

12. Aplikacija treba podržavati funkcionalnosti opisane u ovom Youtube videu:
<https://www.youtube.com/watch?v=T8KVYU8nbKU>.

NAPOMENE:

- 1) Osim implementacija vježbe prema uputama, dozvoljeno je uvoditi i promjene ako su opravdane i ne narušavaju koncepte objektno-orijentiranog programiranja.
- 2) U svrhu korištenja komponente „DateTimePicker“ potrebno je dodati sljedeći „dependency“ u „pom.xml“ datoteku:

```
<dependency>
    <groupId>no.tornado</groupId>
    <artifactId>tornadofx-controls</artifactId>
    <version>1.0.6</version>
</dependency>
```

- 3) Izdvajanje „MenuBar“ komponente moguće je implementirati na način da se sam izbornik izdvoji u zasebnu „fxml“ komponentu kao u nastavku, a ta datoteka se može dodavati u druge „fxml“ datoteke korištenjem oznake „fx:include“ kao što je također prikazano u nastavku:

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.control.Menu?>
<?import javafx.scene.control.MenuBar?>
<?import javafx.scene.control.MenuItem?>

<MenuBar xmlns="http://javafx.com/javafx/10.0.1"
          xmlns:fx="http://javafx.com/fxml/1"
          fx:controller="hr.java.vjezbe.javafx.PocetnaController">

    <menus>
        <Menu mnemonicParsing="false" text="Profesor">
            <items>
                <MenuItem mnemonicParsing="false"
                    onAction="#prikaziPretraguProfesora" text="Pretraga" />
                <MenuItem mnemonicParsing="false"
                    onAction="#prikaziDodavanjeProfesora" text="Unos" />
            </items>
        </Menu>
        <Menu mnemonicParsing="false" text="Student">
            <items>
                <MenuItem mnemonicParsing="false"
                    onAction="#prikaziPretraguStudenta" text="Pretraga" />
                <MenuItem mnemonicParsing="false"
                    onAction="#prikaziDodavanjeStudenta" text="Unos" />
            </items>
        </Menu>
        <Menu mnemonicParsing="false" text="Predmet">
            <items>
                <MenuItem mnemonicParsing="false"
                    onAction="#prikaziPretraguPredmeta" text="Pretraga" />
                <MenuItem mnemonicParsing="false"
                    onAction="#prikaziDodavanjePredmeta" text="Unos" />
            </items>
        </Menu>
        <Menu mnemonicParsing="false" text="Ispit">
            <items>
                <MenuItem mnemonicParsing="false"
                    onAction="#prikaziPretraguIspita" text="Pretraga" />
                <MenuItem mnemonicParsing="false"
                    onAction="#prikaziDodavanjeIspita" text="Unos" />
            </items>
        </Menu>
    </menus>
</MenuBar>
```

```
        </items>
    </Menu>
</menus>
</MenuBar>
```

```
<BorderPane prefHeight="551.0" prefWidth="320.0"
xmlns="http://javafx.com/javafx/10.0.1" xmlns:fx="http://javafx.com/fxml/1"
fx:controller="hr.java.vjezbe.javafx.NoviIspitController">
    <top>
        <fx:include source="menuBar.fxml" />
    </top>
    <center>...
```

- 4) Pretvorba iz „**LocalDate**“ tipa u „**Date**“ koji se zapisuje u bazu podataka (kod objekta klase „**Student**“) i obrnuto moguće je obaviti na sljedeći način:

```
if (Optional.ofNullable(student.getDatumRodjenja()).isPresent()) {
    sqlUpit.append(" AND DATUM_RODJENJA = '"
        + student.getDatumRodjenja().format(
            DateTimeFormatter.ISO_DATE) + "'");
}
```

```
LocalDate datumRodjenja =
resultSet.getTimestamp("datum_rodjenja").toInstant().atZone(
ZoneId.systemDefault()).toLocalDate();
```

```
preparedStatement.setDate(4,
    Date.valueOf(student.getDatumRodjenja()));
```

- 5) Na sličan način, ako je potrebno pretvarati tip „**LocalDateTime**“ u „**Timestamp**“, to je moguće napraviti na sljedeći način:

```
if (Optional.ofNullable(ispit.getDatumIVrijeme()).isPresent()) {
    DateTimeFormatter formatter =
        DateTimeFormatter.ofPattern("yyyy-MM-dd'T'HH:mm:ss.SS");
    sqlUpit.append(" AND datum_i_vrijeme = '"
        + ispit.getDatumIVrijeme().format(formatter) + "'");
}
```

```
LocalDateTime datumIVrijemeIspita =
resultSet.getTimestamp("datum_i_vrijeme").toLocalDateTime();
```

```
preparedStatement.setTimestamp(4,
    Timestamp.valueOf(ispit.getDatumIVrijeme()));
```

- 6) Dodavanje „**DatePicker**“ komponente u FXML datoteke moguće je obaviti dodavanjem sljedeće „import“ naredbe (označene crvenom bojom):

```
<?import javafx.scene.control.ComboBox?>  
<?import tornadofx.control.DateTimePicker?>  
<?import javafx.scene.control.Label?>
```

Sama komponenta se nakon toga može koristiti na sljedeći način:

```
<DateTimePicker fx:id="datumIVrijemeDateTimePicker" GridPane.columnIndex="2"  
    GridPane.columnSpan="2" GridPane.rowIndex="7">  
    <GridPane.margin>  
        <Insets bottom="5.0" left="5.0" right="5.0" top="5.0" />  
    </GridPane.margin>  
</DateTimePicker>
```