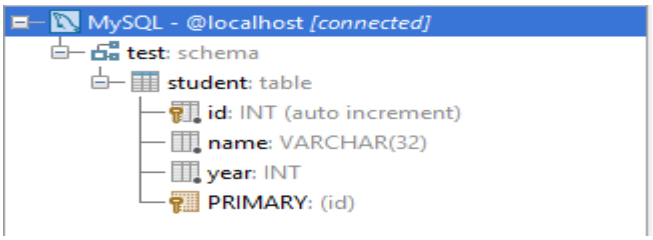
Laboratory work №4

In this laboratory work I had to implement a data access object pattern that will hide all interaction with the database from the rest of the application. I needed the same GUI as in the previous laboratory with edit and delete buttons for each student and a form for adding new student. All data was stored in an SQL database, which had access through the StudentDAOImpl class, using SQL queries.

During this laboratory I learned how to do basic database operations (insert, update, delete in SQL) using JDBC API. I used MySQL database system (test) with one table Student with the following structure:



According to exercise I created StudentDAO interface:

```
public List<Student> findAll();

Student findById(int id);

Student findByName();

boolean insert(Student student);

boolean update(Student student);

boolean delete(Student student);
```

The following code snippet connects to the database by the user *root* and password *secret*:

```
String dbURL = "jdbc:mysql://localhost:3306/test";

String username = "root";

String password = "";
```

In connecting to database I took advantage of the try with-resources statement which will close the connection automatically, as shown for findById: public Student findById(int id) {

```
try {
                    Connection conn = DriverManager.getConnection(dbURL, username,
password);

if (conn != null) { // Connected

   String sql = "SELECT * FROM Student WHERE id = " + id;

Statement statement = conn.createStatement();

ResultSet result = statement.executeQuery(sql);
```

```
        if (result.next()){

                return new Student(result.getInt("id"), result.getString("name"), result.getInt("year"));
}          }

      } catch (SQLException ex) {

         ex.printStackTrace();

      } return null;
```

- DriverManager: this class is used to register driver for a specific database type and to establish a database connection with the server via its getConnection() method.
- Statement: interface is used to execute static SQL query
- ResultSet executeQuery(String sql): executes a SELECT statement and returns a ResultSet object which contains results returned by the query.
- SQLException: this checked exception is declared to be thrown by all the above methods, so we have to catch this exception explicitly when calling the above classes' methods.

Servlets are the Java platform technology of choice for extending and enhancing Web servers. They have access to the entire family of Java APIs, including the JDBC API to access enterprise databases. Also can access a library of HTTP-specific calls and receive all the benefits of the mature Java language.

A web Server means Handling HTTP requests (usually from browsers). A Servlet Container means that it can handle servlets & JSP. An Application Server (GlassFish) means it can manage Java EE applications (usually both servlet/JSP).

INSERT statement:

```
String name = request.getParameter("name");

      int year = Integer.parseInt(request.getParameter("year"));

      Student student = new Student(0, name, year);

      StudentDAOImpl dao = new StudentDAOImpl();

if (dao.insert(student)) {

         response.sendRedirect("main");
```

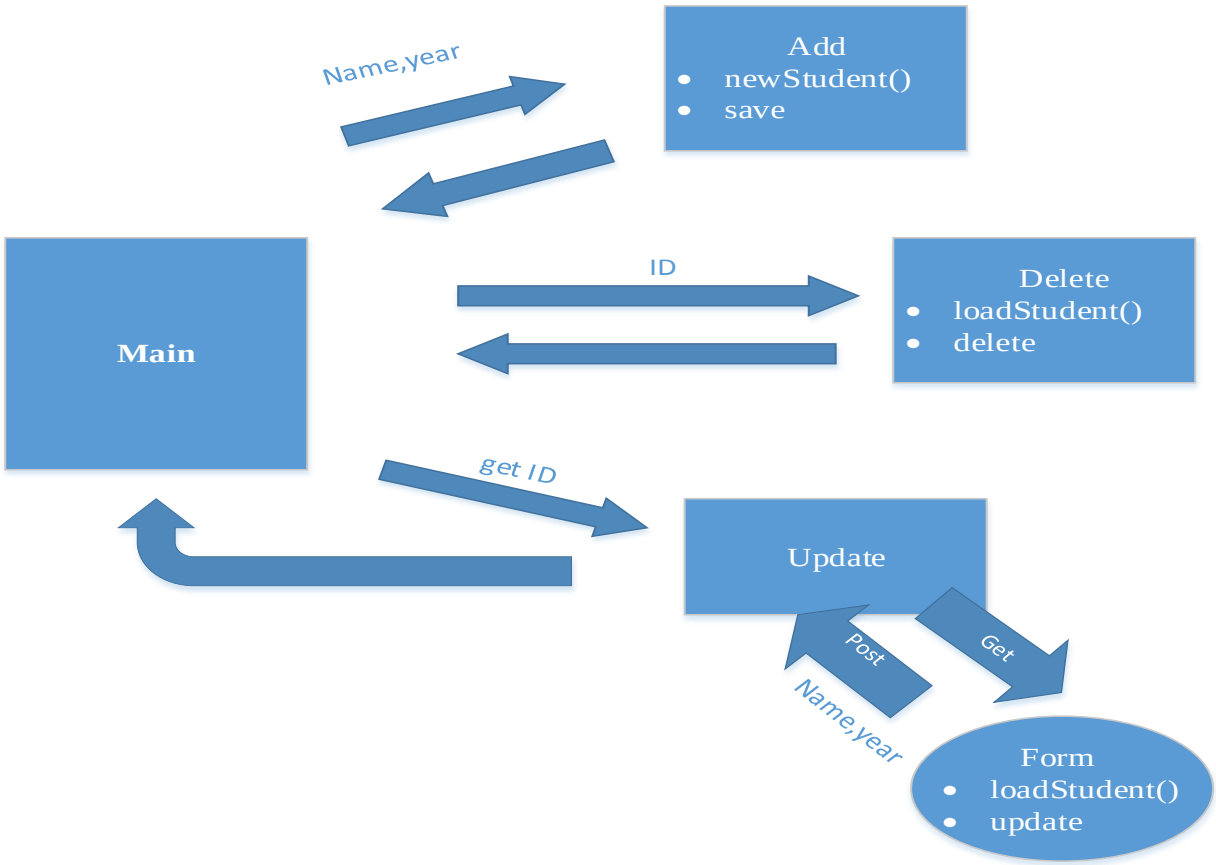
DELETE statement:

```
      int id = Integer.parseInt(request.getParameter("id"));

      StudentDAOImpl dao= new StudentDAOImpl();

      Student student = dao.findById(id);

      if (dao.delete(student)) {

        response.sendRedirect("main");
```

UPDATE statement for doPost:

```
int id = Integer.parseInt(request.getParameter("id"));
        String name = request.getParameter("name");
        int year = Integer.parseInt(request.getParameter("year"));
Student student = new Student(id, name, year);
        StudentDAOImpl dao = new StudentDAOImpl();
        if (dao.update(student)) {
            response.sendRedirect("main");
    } else {
            response.sendRedirect("update?id="+id);     // error
```

As on the previous laboratory for interface of my application I used two pages, where one of them is main page and another for editing the information about them. A JSP page is basically a web page with traditional HTML and bits of Java code. When a JSP page is called, it will be compiled (by the JSP engine) into a Java servlet. At this point the servlet is handled by the servlet engine, just like any other servlet. The servlet engine then loads the servlet class and executes it to create dynamic HTML to be sent to the browser. The servlet creates any necessary object, and writes any object as a string to an output stream to the browser.



Finally after switching on the Open Server.exe I got my web application:

The main page:

List of all students:

ID	Name	Year		
1	Adema	19	[Update]	[Delete]
5	Dosya	19	[Update]	[Delete]
8	Konya	19	[Update]	[Delete]
9	Akerke	20	[Update]	[Delete]

Add new student:

Name

Year

Отправить

and for editing:

Add new student:

Name: Dosya

Year: 19

Отправить