Learners Academy Source Code :

Package Name: package com.learnersacademy.dao

Admin_db.java:

```
package com.learnersacademy.dao;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;


import com.learnersacademy.model.Admin;

public class Admin_db {
	private String dbUrl = "jdbc:mysql://localhost:3306/learnersacademy?useSSL=false";
	private String dbUname = "root";
	private String dbPassword = "root@123";
	private String dbDriver = "com.mysql.cj.jdbc.Driver";

	public void loadDriver(String dbDriver)
	{
		try {
			Class.forName(dbDriver);
		} catch (ClassNotFoundException e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		}
	}

	public Connection getConnection()
	{
		Connection con = null;
		try {
			con = DriverManager.getConnection(dbUrl, dbUname, dbPassword);
		} catch (SQLException e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		}
		return con;
	}

	public boolean validate(Admin loginBean)
	{
		boolean status = false;

		loadDriver(dbDriver);
		Connection con = getConnection();

		String sql = "select * from admin where username = ? and password =?";
		PreparedStatement ps;
		try {
		ps = con.prepareStatement(sql);
		ps.setString(1, loginBean.getUserName());
		ps.setString(2, loginBean.getPassword());
		ResultSet rs = ps.executeQuery();
		status = rs.next();

		} catch (SQLException e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		}
		return status;
	}
}
```

Classes_db.java:

```
package com.learnersacademy.dao;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;

import com.learnersacademy.model.Classes;


public class Classes_db {
	private String jdbcURL =
"jdbc:mysql://localhost:3306/learnersacademy?useSSL=false";
	private String jdbcUsername = "root";
	private String jdbcPassword = "root@123";
	private static final String INSERT_STUDENTS_SQL = "INSERT INTO classes" + "  (name,
teacher, duration,courses) VALUES "
				+ " (?, ?, ?, ?);";

	private static final String SELECT_STUDENTS_BY_ID = "select
id,name,teacher,duration,courses from classes where id =?";
	private static final String SELECT_ALL_STUDENTS = "select * from classes";
	private static final String DELETE_STUDENTS_SQL = "delete from classes where id =
?;";
	private static final String UPDATE_STUDENTS_SQL = "update classes set name =
?,teacher= ?, duration =?,courses=? where id = ?;";

	public Classes_db() {
	}
	protected Connection getConnection() {
		Connection connection = null;
		try {
			Class.forName("com.mysql.jdbc.Driver");
			connection = DriverManager.getConnection(jdbcURL, jdbcUsername,
jdbcPassword);
		} catch (SQLException e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		} catch (ClassNotFoundException e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		}
		return connection;
	}

	public void insertSubject(Classes student) throws SQLException {
		System.out.println(INSERT_STUDENTS_SQL);
		try (Connection connection = getConnection();
				PreparedStatement preparedStatement =
connection.prepareStatement(INSERT_STUDENTS_SQL)) {
			preparedStatement.setString(1, student.getName());
			preparedStatement.setString(2, student.getTeacher());
			preparedStatement.setString(3, student.getDuration());
			preparedStatement.setString(4, student.getCourses());

			System.out.println(preparedStatement);
			preparedStatement.executeUpdate();
		} catch (SQLException e) {
			printSQLException(e);
		}
	}
```

```
          public Classes selectSubject(int id) {
                    Classes student = null;
                                        try (Connection connection = getConnection();
                                                  PreparedStatement preparedStatement =
connection.prepareStatement(SELECT_STUDENTS_BY_ID);) {
                              preparedStatement.setInt(1, id);
                              System.out.println(preparedStatement);
                       ResultSet rs = preparedStatement.executeQuery();

                                                            while (rs.next()) {
                                        String name = rs.getString("name");
                                        String email = rs.getString("teacher");
                                        String duration = rs.getString("duration");
                                        String courses = rs.getString("courses");


                                        student = new Classes(id, name, email, duration,courses);
                              }
                    } catch (SQLException e) {
                              printSQLException(e);
                    }
                    return student;
          }

          public List<Classes> selectAllSubjects() {


                    List<Classes> users = new ArrayList<>();

                    try (Connection connection = getConnection();

                              PreparedStatement preparedStatement =
connection.prepareStatement(SELECT_ALL_STUDENTS);) {
                              System.out.println(preparedStatement);
                              ResultSet rs = preparedStatement.executeQuery();

                              while (rs.next()) {
                                        int id = rs.getInt("id");
                                        String name = rs.getString("name");
                                        String teachername = rs.getString("teacher");
                                        String duration = rs.getString("duration");
                                        String courses = rs.getString("courses");
                                        users.add(new Classes(id, name, teachername,
duration,courses));


                              }
```

```
		} catch (SQLException e) {
			printSQLException(e);
		}
		return users;
	}

	public boolean deleteClasses(int id) throws SQLException {
		boolean rowDeleted;
		try (Connection connection = getConnection();
				PreparedStatement statement =
connection.prepareStatement(DELETE_STUDENTS_SQL);) {
			statement.setInt(1, id);
			rowDeleted = statement.executeUpdate() > 0;
		}
		return rowDeleted;
	}

	public boolean updateClasses(Classes student) throws SQLException {
		boolean rowUpdated;
		try (Connection connection = getConnection();
				PreparedStatement statement =
connection.prepareStatement(UPDATE_STUDENTS_SQL);) {
			System.out.println("updated student list:"+statement);
			statement.setString(1, student.getName());
			statement.setString(2, student.getTeacher());
			statement.setString(3, student.getDuration());
			statement.setString(4, student.getCourses());
			statement.setInt(5, student.getId());

			rowUpdated = statement.executeUpdate() > 0;
		}
		return rowUpdated;
	}

	private void printSQLException(SQLException ex) {
		for (Throwable e : ex) {
			if (e instanceof SQLException) {
				e.printStackTrace(System.err);
				System.err.println("SQLState: " + ((SQLException)
e).getSQLState());
				System.err.println("Error Code: " + ((SQLException)
e).getErrorCode());
				System.err.println("Message: " + e.getMessage());
				Throwable t = ex.getCause();
				while (t != null) {
					System.out.println("Cause: " + t);
					t = t.getCause();
```

```
                              }
                        }
                  }
            }


}
```

Courses_db.java:

```
package com.learnersacademy.dao;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;

import com.learnersacademy.model.Courses;


public class Courses_db {
      private String jdbcURL =
"jdbc:mysql://localhost:3306/learnersacademy?useSSL=false";
      private String jdbcUsername = "root";
      private String jdbcPassword = "root@123";

      private static final String INSERT_STUDENTS_SQL = "INSERT INTO courses" + "
(course) VALUES "
                        + " (?);";

      private static final String SELECT_STUDENTS_BY_ID = "select id,course from courses
where id =?";
      private static final String SELECT_ALL_STUDENTS = "select * from courses";
      private static final String DELETE_STUDENTS_SQL = "delete from courses where id =
?;";
      private static final String UPDATE_STUDENTS_SQL = "update courses set course = ?
where id = ?;";


      public Courses_db() {
      }
      protected Connection getConnection() {
            Connection connection = null;
            try {
```

```
				Class.forName("com.mysql.jdbc.Driver");
				connection = DriverManager.getConnection(jdbcURL, jdbcUsername,
jdbcPassword);
			} catch (SQLException e) {
				// TODO Auto-generated catch block
				e.printStackTrace();
			} catch (ClassNotFoundException e) {
				// TODO Auto-generated catch block
				e.printStackTrace();
			}
			return connection;
		}

		public void insertSubject(Courses student) throws SQLException {
			System.out.println(INSERT_STUDENTS_SQL);
			try (Connection connection = getConnection();
					PreparedStatement preparedStatement =
connection.prepareStatement(INSERT_STUDENTS_SQL)) {
				preparedStatement.setString(1, student.getCourse());
				System.out.println(preparedStatement);
				preparedStatement.executeUpdate();
			} catch (SQLException e) {
				printSQLException(e);
			}
		}


		public Courses selectSubject(int id) {
			Courses student = null;
					try (Connection connection = getConnection();
						PreparedStatement preparedStatement =
connection.prepareStatement(SELECT_STUDENTS_BY_ID);) {
				preparedStatement.setInt(1, id);
				System.out.println(preparedStatement);
			   ResultSet rs = preparedStatement.executeQuery();

							while (rs.next()) {
					String course = rs.getString("course");

					student = new Courses(id, course);
				}
			} catch (SQLException e) {
				printSQLException(e);
			}
			return student;
		}
```

```
	public List<Courses> selectAllSubjects() {


		List<Courses> users = new ArrayList<>();

		try (Connection connection = getConnection();

				PreparedStatement preparedStatement = 
connection.prepareStatement(SELECT_ALL_STUDENTS);) {
				System.out.println(preparedStatement);
				ResultSet rs = preparedStatement.executeQuery();

				while (rs.next()) {
					int id = rs.getInt("id");
					String course = rs.getString("course");

					users.add(new Courses(id, course));


				}
		} catch (SQLException e) {
				printSQLException(e);
		}
		return users;
	}

	public boolean deleteClasses(int id) throws SQLException {
		boolean rowDeleted;
		try (Connection connection = getConnection();
					PreparedStatement statement = 
connection.prepareStatement(DELETE_STUDENTS_SQL);) {
				statement.setInt(1, id);
				rowDeleted = statement.executeUpdate() > 0;
		}
		return rowDeleted;
	}

	public boolean updateClasses(Courses student) throws SQLException {
		boolean rowUpdated;
		try (Connection connection = getConnection();
					PreparedStatement statement = 
connection.prepareStatement(UPDATE_STUDENTS_SQL);) {
				System.out.println("updated student list:"+statement);
				statement.setString(1, student.getCourse());
				statement.setInt(2, student.getId());

				rowUpdated = statement.executeUpdate() > 0;
		}
		return rowUpdated;
	}

	private void printSQLException(SQLException ex) {
		for (Throwable e : ex) {
			if (e instanceof SQLException) {
				e.printStackTrace(System.err);
				System.err.println("SQLState: " + ((SQLException)
e).getSQLState());
				System.err.println("Error Code: " + ((SQLException)
e).getErrorCode());
				System.err.println("Message: " + e.getMessage());
				Throwable t = ex.getCause();
				while (t != null) {
					System.out.println("Cause: " + t);
					t = t.getCause();
				}
			}
		}
	}


}
```

Students_db.java:

```
package com.learnersacademy.dao;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;

import com.learnersacademy.model.Students;


public class Students_db {
	private String jdbcURL =
"jdbc:mysql://localhost:3306/learnersacademy?useSSL=false";
	private String jdbcUsername = "root";
	private String jdbcPassword = "root@123";
	private static final String INSERT_STUDENTS_SQL = "INSERT INTO students" + "
(name, email, country,course) VALUES "
			+ " (?, ?, ?, ?);";

	private static final String SELECT_STUDENTS_BY_ID = "select
id,name,email,country,course from students where id =?";
	private static final String SELECT_ALL_STUDENTS = "select * from students";
	private static final String DELETE_STUDENTS_SQL = "delete from students where id =
?;";
	private static final String UPDATE_STUDENTS_SQL = "update students set name =
?,email= ?, country =?,course=? where id = ?;";

	public Students_db() {
	}
	protected Connection getConnection() {
		Connection connection = null;
		try {
			Class.forName("com.mysql.jdbc.Driver");
			connection = DriverManager.getConnection(jdbcURL, jdbcUsername,
jdbcPassword);
		} catch (SQLException e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		} catch (ClassNotFoundException e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		}
		return connection;
	}

	public void insertStudent(Students student) throws SQLException {
		System.out.println(INSERT_STUDENTS_SQL);
		try (Connection connection = getConnection();
				PreparedStatement preparedStatement =
connection.prepareStatement(INSERT_STUDENTS_SQL)) {
			preparedStatement.setString(1, student.getName());
			preparedStatement.setString(2, student.getEmail());
			preparedStatement.setString(3, student.getCountry());
			preparedStatement.setString(4, student.getCourse());

			System.out.println(preparedStatement);
			preparedStatement.executeUpdate();
		} catch (SQLException e) {
			printSQLException(e);
		}
	}
	public Students selectStudent(int id) {
		Students student = null;
				try (Connection connection = getConnection();
					PreparedStatement preparedStatement =
connection.prepareStatement(SELECT_STUDENTS_BY_ID);) {
			preparedStatement.setInt(1, id);
			System.out.println(preparedStatement);
		   ResultSet rs = preparedStatement.executeQuery();

						while (rs.next()) {
				String name = rs.getString("name");
				String email = rs.getString("email");
				String country = rs.getString("country");
				String course = rs.getString("course");
				student = new Students(id, name, email, country,course);
			}
		} catch (SQLException e) {
			printSQLException(e);
		}
		return student;
	}

	public List<Students> selectAllStudents() {

		List<Students> users = new ArrayList<>();

		try (Connection connection = getConnection();

			PreparedStatement preparedStatement =
connection.prepareStatement(SELECT_ALL_STUDENTS);) {
			System.out.println(preparedStatement);

			ResultSet rs = preparedStatement.executeQuery();

			while (rs.next()) {
				int id = rs.getInt("id");
				String name = rs.getString("name");
				String email = rs.getString("email");
				String country = rs.getString("country");
				String course = rs.getString("course");

				users.add(new Students(id, name, email, country,course));
			}
		} catch (SQLException e) {
			printSQLException(e);
		}
		return users;
	}

	public boolean deleteStudent(int id) throws SQLException {
		boolean rowDeleted;
		try (Connection connection = getConnection();
				PreparedStatement statement =
connection.prepareStatement(DELETE_STUDENTS_SQL);) {
			statement.setInt(1, id);
			rowDeleted = statement.executeUpdate() > 0;
		}
		return rowDeleted;
	}

	public boolean updateStudent(Students student) throws SQLException {
		boolean rowUpdated;
		try (Connection connection = getConnection();
				PreparedStatement statement =
connection.prepareStatement(UPDATE_STUDENTS_SQL);) {
			System.out.println("updated student list:"+statement);
			statement.setString(1, student.getName());
			statement.setString(2, student.getEmail());
			statement.setString(3, student.getCountry());
			statement.setString(4, student.getCourse());
			statement.setInt(5, student.getId());

			rowUpdated = statement.executeUpdate() > 0;
		}
		return rowUpdated;
	}

	private void printSQLException(SQLException ex) {
		for (Throwable e : ex) {
			if (e instanceof SQLException) {
				e.printStackTrace(System.err);
				System.err.println("SQLState: " + ((SQLException)
e).getSQLState());
				System.err.println("Error Code: " + ((SQLException)
e).getErrorCode());
				System.err.println("Message: " + e.getMessage());
				Throwable t = ex.getCause();
				while (t != null) {
					System.out.println("Cause: " + t);
					t = t.getCause();
				}
			}
		}
	}
```

```
}
```

Teachers_db.java:

```
package com.learnersacademy.dao;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;

import com.learnersacademy.model.Teachers;


public class Teachers_db {
        private String jdbcURL =
"jdbc:mysql://localhost:3306/learnersacademy?useSSL=false";
        private String jdbcUsername = "root";
        private String jdbcPassword = "root@123";

        private static final String INSERT_TEACHERS_SQL = "INSERT INTO teachers" + "
(name, email, experience) VALUES "
                        + " (?, ?, ?);";

        private static final String SELECT_TEACHERS_BY_ID = "select
id,name,email,experience from teachers where id =?";
        private static final String SELECT_ALL_TEACHERS = "select * from teachers";
        private static final String DELETE_TEACHERS_SQL = "delete from teachers where id =
?;";
        private static final String UPDATE_TEACHERS_SQL = "update teachers set name =
?,email= ?, experience =? where id = ?;";

        public Teachers_db() {
        }
        protected Connection getConnection() {
                Connection connection = null;
                try {
                        Class.forName("com.mysql.jdbc.Driver");
                        connection = DriverManager.getConnection(jdbcURL, jdbcUsername,
jdbcPassword);
                } catch (SQLException e) {
                        e.printStackTrace();
                } catch (ClassNotFoundException e) {
```

```
				// TODO Auto-generated catch block
				e.printStackTrace();
			}
			return connection;
		}

		public void insertTeacher(Teachers teacher) throws SQLException {
			System.out.println(INSERT_TEACHERS_SQL);
			try (Connection connection = getConnection();
					PreparedStatement preparedStatement =
connection.prepareStatement(INSERT_TEACHERS_SQL)) {
				preparedStatement.setString(1, teacher.getName());
				preparedStatement.setString(2, teacher.getEmail());
				preparedStatement.setString(3, teacher.getExperience());
				System.out.println(preparedStatement);
				preparedStatement.executeUpdate();
			} catch (SQLException e) {
				printSQLException(e);
			}
		}

		public Teachers selectTeacher(int id) {
			Teachers teacher = null;
					try (Connection connection = getConnection();
						PreparedStatement preparedStatement =
connection.prepareStatement(SELECT_TEACHERS_BY_ID);) {
				preparedStatement.setInt(1, id);
				System.out.println(preparedStatement);
			   ResultSet rs = preparedStatement.executeQuery();

							while (rs.next()) {
					String name = rs.getString("name");
					String email = rs.getString("email");
					String exp = rs.getString("experience");
					teacher = new Teachers(id, name, email, exp);
				}
			} catch (SQLException e) {
				printSQLException(e);
			}
			return teacher;
		}

		public List<Teachers> selectAllTeachers() {

			List<Teachers> users = new ArrayList<>();

			try (Connection connection = getConnection();
                        PreparedStatement preparedStatement =
connection.prepareStatement(SELECT_ALL_TEACHERS);) {
                        System.out.println(preparedStatement);

                        ResultSet rs = preparedStatement.executeQuery();


                        while (rs.next()) {
                                int id = rs.getInt("id");
                                String name = rs.getString("name");
                                String email = rs.getString("email");
                                String exp = rs.getString("experience");
                                users.add(new Teachers(id, name, email, exp));
                        }
                } catch (SQLException e) {
                        printSQLException(e);
                }
                return users;
        }

        public boolean deleteTeacher(int id) throws SQLException {
                boolean rowDeleted;
                try (Connection connection = getConnection();
                                PreparedStatement statement =
connection.prepareStatement(DELETE_TEACHERS_SQL);) {
                        statement.setInt(1, id);
                        rowDeleted = statement.executeUpdate() > 0;
                }
                return rowDeleted;
        }

        public boolean updateTeacher(Teachers teacher) throws SQLException {
                boolean rowUpdated;
                try (Connection connection = getConnection();
                                PreparedStatement statement =
connection.prepareStatement(UPDATE_TEACHERS_SQL);) {
                        System.out.println("updated Teachers list:"+statement);
                        statement.setString(1, teacher.getName());
                        statement.setString(2, teacher.getEmail());
                        statement.setString(3, teacher.getExperience());
                        statement.setInt(4, teacher.getId());

                        rowUpdated = statement.executeUpdate() > 0;
                }
                return rowUpdated;
        }
```

```
	private void printSQLException(SQLException ex) {
		for (Throwable e : ex) {
			if (e instanceof SQLException) {
				e.printStackTrace(System.err);
				System.err.println("SQLState: " + ((SQLException)
e).getSQLState());
				System.err.println("Error Code: " + ((SQLException)
e).getErrorCode());
				System.err.println("Message: " + e.getMessage());
				Throwable t = ex.getCause();
				while (t != null) {
					System.out.println("Cause: " + t);
					t = t.getCause();
				}
			}
		}
	}


}
```

Package Name : `com.learnersacademy.model`

Admin.java:

```
package com.learnersacademy.model;

public class Admin {

    protected String userName;
    protected String password;

    public Admin() {

    }

    public String getUserName() {
        return userName;
    }
    public void setUserName(String userName) {
        this.userName = userName;
    }
    public String getPassword() {
        return password;
```

```
	}
	public void setPassword(String password) {
		this.password = password;
	}

}
```

Clases.java:

```
package com.learnersacademy.model;

public class Classes {
	protected int id;
	protected String name;
	protected String teacher;
	protected String duration;
	protected String courses;

	public Classes(int id, String name, String teacher, String duration, String courses) {
		// TODO Auto-generated constructor stub
		super();
		this.id = id;
		this.name = name;
		this.teacher = teacher;
		this.duration = duration;
		this.courses = courses;

	}
	public Classes(String name, String teacher, String duration, String courses) {
		// TODO Auto-generated constructor stub
		super();
		this.name = name;
		this.teacher = teacher;
		this.duration = duration;
		this.courses = courses;


	}

	public int getId() {
		return id;
	}
	public void setId(int id) {
		this.id = id;
	}
	public String getName() {
		return name;
	}
	public void setName(String name) {
		this.name = name;
	}
	public String getTeacher() {
		return teacher;
	}
	public void setTeacher(String teacher) {
		this.teacher = teacher;
	}
	public String getCourses() {
		return courses;
	}
	public void setCourses(String courses) {
		this.courses = courses;
	}
	public String getDuration() {
		return duration;
	}
	public void setDuration(String duration) {
		this.duration = duration;
	}

}
```

Courses.java:

```
package com.learnersacademy.model;

public class Courses {
    protected int id;
    protected String course;


    public Courses(int id, String course) {
        // TODO Auto-generated constructor stub
        super();
        this.id = id;
        this.course = course;

    }
    public Courses(String course) {
```

```
		// TODO Auto-generated constructor stub
		super();
		this.course = course;



	}

	public int getId() {
		return id;
	}
	public void setId(int id) {
		this.id = id;
	}
	public String getCourse() {
		return course;
	}
	public void setCourse(String course) {
		this.course = course;
	}


}
```

Students.java:

```
package com.learnersacademy.model;

public class Students {
	protected int id;
	protected String name;
	protected String email;
	protected String country;
	protected String course;

	public Students(int id, String name, String email, String country, String course) {
		// TODO Auto-generated constructor stub
		super();
		this.id = id;
		this.name = name;
		this.email = email;
		this.country = country;
		this.course =course;
	}
	public Students(String name, String email, String country,String course) {
		// TODO Auto-generated constructor stub
		super();
		this.name = name;
		this.email = email;
		this.country = country;
		this.course =course;

	}

	public int getId() {
		return id;
	}
	public void setId(int id) {
		this.id = id;
	}
	public String getName() {
		return name;
	}
	public void setName(String name) {
		this.name = name;
	}
	public String getEmail() {
		return email;
	}
	public void setEmail(String email) {
		this.email = email;
	}
	public String getCountry() {
		return country;
	}
	public void setCountry(String country) {
		this.country = country;
	}
	public String getCourse() {
		return course;
	}
	public void setCourse(String course) {
		this.course = course;
	}


}
```

Teachers.java:

```
package com.learnersacademy.model;
```

```
public class Teachers {
	protected int id;
	protected String name;
	protected String email;
	protected String experience;

	public Teachers(int id, String name, String email, String experience) {
		// TODO Auto-generated constructor stub
		super();
		this.id = id;
		this.name = name;
		this.email = email;
		this.experience = experience;
	}
	public Teachers(String name, String email, String experience) {
		// TODO Auto-generated constructor stub
		super();
		this.name = name;
		this.email = email;
		this.experience = experience;
	}

	public int getId() {
		return id;
	}
	public void setId(int id) {
		this.id = id;
	}
	public String getName() {
		return name;
	}
	public void setName(String name) {
		this.name = name;
	}
	public String getEmail() {
		return email;
	}
	public void setEmail(String email) {
		this.email = email;
	}
	public String getExperience() {
		return experience;
	}
	public void setExperience(String experience) {
		this.experience = experience;
	}
}
```

Package Name: com.learnersacademy.web

AdminLogin.java

```
package com.learnersacademy.web;

import java.io.IOException;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.Cookie;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;



import com.learnersacademy.dao.Admin_db;
import com.learnersacademy.model.Admin;

/**
 * Servlet implementation class AdminLogin
 */
@WebServlet("/verifylogin")
public class AdminLogin extends HttpServlet {
	private static final long serialVersionUID = 1L;

   /**
    * @see HttpServlet#HttpServlet()
    */
   public AdminLogin() {
      super();
      // TODO Auto-generated constructor stub
   }

	/**
	 * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse
response)
	 */
	protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
		// TODO Auto-generated method stub
		response.getWriter().append("Served at:
").append(request.getContextPath());
```

```
	}

	/**
	 * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse
response)
	 */
	protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
		// TODO Auto-generated method stub
		Admin_db admin = new Admin_db();
		String username = request.getParameter("username");
		String password = request.getParameter("password");
		Admin login = new Admin();
		login.setUserName(username);
		login.setPassword(password);

		if(admin.validate(login)) {
			Cookie c = new Cookie("login","true");
			c.setMaxAge(1800);
			response.addCookie(c);
			response.sendRedirect("/LearnersAcademy/");
		}
		else {
			response.sendRedirect("adminLogin.jsp");
		}
	}

}
```

ListClasses.java

```
package com.learnersacademy.web;

import java.io.IOException;
import java.sql.SQLException;
import java.util.List;

import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.Cookie;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import com.learnersacademy.dao.Classes_db;
import com.learnersacademy.model.Classes;
import com.learnersacademy.model.Students;

/**
 * Servlet implementation class ListClasses
 */
@WebServlet("/list-class")
public class ListClasses extends HttpServlet {
	private static final long serialVersionUID = 1L;
	private Classes_db classesdb;

   /**
    * @see HttpServlet#HttpServlet()
    */
   public ListClasses() {
      super();
      // TODO Auto-generated constructor stub
   }
   public void init() {
	classesdb = new Classes_db();

	}
	/**
	 * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse
response)
	 */
	protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {

		try {

			listUser(request, response);

		} catch (SQLException ex) {
			throw new ServletException(ex);
		}



	}
	  private void listUser(HttpServletRequest request, HttpServletResponse response)
			throws SQLException, IOException, ServletException {
		  Boolean login =false;
		Cookie c[]=request.getCookies();
		for(int i=0;i<c.length;i++){
			if(c[i].getName().equals("login" ) && c[i].getValue().equals("true")) {
				System.out.println("Peace Bro");
				login=true;
```

```
                                }
                               }
                    if(login) {
                               List<Classes> listUser = classesdb.selectAllSubjects();
                                          request.setAttribute("listClasses", listUser);
                                          RequestDispatcher dispatcher =
request.getRequestDispatcher("class-list.jsp");
                                          dispatcher.forward(request, response);
                    }
                    else {
                               RequestDispatcher dispatcher =
request.getRequestDispatcher("adminLogin.jsp");
                               dispatcher.forward(request, response);
                    }

                    }
          /**
           * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse
response)
           */


}
```

ListCourses.java:

```
package com.learnersacademy.web;

import java.io.IOException;
import java.sql.SQLException;
import java.util.List;

import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.Cookie;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;


import com.learnersacademy.dao.Courses_db;
import com.learnersacademy.model.Classes;
import com.learnersacademy.model.Courses;

/**
```

```
 * Servlet implementation class ListClasses
 */
@WebServlet("/list-courses")
public class ListCourses extends HttpServlet {
	private static final long serialVersionUID = 1L;
	private Courses_db coursesdb;

   /**
    * @see HttpServlet#HttpServlet()
    */
   public ListCourses() {
      super();
      // TODO Auto-generated constructor stub
   }
   public void init() {
	coursesdb = new Courses_db();

	}
	/**
	 * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse
response)
	 */
	protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {

		try {
				listUser(request, response);

		} catch (SQLException ex) {
			throw new ServletException(ex);
		}



	}
	 private void listUser(HttpServletRequest request, HttpServletResponse response)
				throws SQLException, IOException, ServletException {
		  Boolean login =false;
		Cookie c[]=request.getCookies();
		for(int i=0;i<c.length;i++){
			 if(c[i].getName().equals("login" ) && c[i].getValue().equals("true")) {
				 System.out.println("Peace Bro");
				 login=true;
			 }
			}
		if(login) {
			List<Courses> listUser = coursesdb.selectAllSubjects();
```

```
					request.setAttribute("listCourses", listUser);
					RequestDispatcher dispatcher = 
request.getRequestDispatcher("list-courses.jsp");
					dispatcher.forward(request, response);
		}
		else {
				RequestDispatcher dispatcher = 
request.getRequestDispatcher("adminLogin.jsp");
				dispatcher.forward(request, response);
		}

		}
	/**
	 * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse 
response)
	 */


}
```

ListServlet.java

```
package com.learnersacademy.web;

import java.io.IOException;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;




import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.Cookie;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import com.learnersacademy.dao.Admin_db;
import com.learnersacademy.dao.Classes_db;
import com.learnersacademy.dao.Courses_db;
import com.learnersacademy.dao.Students_db;
import com.learnersacademy.model.Admin;
import com.learnersacademy.model.Classes;
import com.learnersacademy.model.Courses;
import com.learnersacademy.model.Students;

import com.learnersacademy.dao.Teachers_db;
import com.learnersacademy.model.Teachers;

/**
 * Servlet implementation class ListServlet
 */
@WebServlet("/")
public class ListServlet extends HttpServlet {
	private static final long serialVersionUID = 1L;
	private Students_db studentDb;
	private Teachers_db teacherDb;
	private Classes_db classesDb;
	private Courses_db coursesdb;
	private Admin_db adminDb;


	public void init() {
		studentDb = new Students_db();
		teacherDb = new Teachers_db();
		classesDb = new Classes_db();
		coursesdb = new Courses_db();
		adminDb = new Admin_db();
	}

    /**
     * @see HttpServlet#HttpServlet()
     */
    public ListServlet() {
        super();
        // TODO Auto-generated constructor stub
    }

	/**
	 * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse
response)
	 */
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
				throws ServletException, IOException {
			String action = request.getServletPath();

			try {
				switch (action) {
				case "/new":
            showNewForm(request, response);
            break;
case "/insert":
            insertUser(request, response);
            break;
case "/delete":
            deleteUser(request, response);
            break;
case "/edit":
            showEditForm(request, response);
            break;
case "/update":
            updateUser(request, response);
            break;

case "/newteacher":
            showNewFormTeacher(request, response);
            break;


case "/insertteacher":
            insertUserTeacher(request, response);
            break;
case "/deleteteacher":
            deleteUserTeacher(request, response);
            break;
case "/editteacher":
            showEditFormTeacher(request, response);
            break;
case "/updateteacher":
            updateUserTeacher(request, response);
            break;

case "/newclassroom":
            newClassRoom(request, response);
            break;

case "/insertclassroom":
            insertClassRoom(request, response);
            break;

case "/updateclassroom":
            updateClassRoom(request, response);
            break;
case "/deleteclassroom":
            deleteClassRoom(request, response);
            break;
```

```
                case "/editclassroom":
                        showEditClassRoom(request, response);
                        break;


                case "/newcourse":
                        newCourse(request, response);
                        break;

                case "/insertcourse":
                        insertCourse(request, response);
                        break;

                case "/updatecourse":
                        updateCourses(request, response);
                        break;
                case "/deletecourse":
                        deleteCourses(request, response);
                        break;
                case "/editcourse":
                        showEditCourses(request, response);
                        break;

                case "/list-report":
                         showReport(request,response);
                         break;
                case "/login":
                        login(request,response);
                        break;
                default:
                        listUser(request, response);
                        break;
                }
        } catch (SQLException ex) {
                throw new ServletException(ex);
        }
    }


private void login(HttpServletRequest request, HttpServletResponse response)
        throws SQLException, IOException, ServletException {
    Cookie c = new Cookie("login","false");
        c.setMaxAge(0);
        response.addCookie(c);
    RequestDispatcher dispatcher = request.getRequestDispatcher("adminLogin.jsp");
        dispatcher.forward(request, response);
}
```

```
    private void showReport(HttpServletRequest request, HttpServletResponse response)
                    throws SQLException, IOException, ServletException {
            RequestDispatcher dispatcher = request.getRequestDispatcher("report.jsp");
                    dispatcher.forward(request, response);
    }

    private void newClassRoom(HttpServletRequest request, HttpServletResponse response)
                    throws SQLException, IOException, ServletException {
            List <Teachers> teachers = teacherDb.selectAllTeachers();
            request.setAttribute("teachersList", teachers);
            List <Courses> mycourse = coursesdb.selectAllSubjects();
            request.setAttribute("courses", mycourse);
            RequestDispatcher dispatcher = request.getRequestDispatcher("newClassRoom-
form.jsp");
                    dispatcher.forward(request, response);
    }

    private void newCourse(HttpServletRequest request, HttpServletResponse response)
                    throws SQLException, IOException, ServletException {
            List <Courses> mycourses = coursesdb.selectAllSubjects();
            request.setAttribute("coursesList", mycourses);
            RequestDispatcher dispatcher = request.getRequestDispatcher("courses-form.jsp");
                    dispatcher.forward(request, response);
    }

    private void listUser(HttpServletRequest request, HttpServletResponse response)
                            throws SQLException, IOException, ServletException {
            Boolean login =false;
            Cookie c[]=request.getCookies();
            for(int i=0;i<c.length;i++){
                     if(c[i].getName().equals("login" ) && c[i].getValue().equals("true")) {
                             login=true;
                     }
                    }
            if(login) {
                    List<Students> listUser = studentDb.selectAllStudents();
                    request.setAttribute("listStudent", listUser);
                    RequestDispatcher dispatcher = request.getRequestDispatcher("student-
list.jsp");
                    dispatcher.forward(request, response);
            }
            else {
                    RequestDispatcher dispatcher =
request.getRequestDispatcher("adminLogin.jsp");
                    dispatcher.forward(request, response);
            }
```

```
	}

	private void showNewForm(HttpServletRequest request, HttpServletResponse
response)
			throws ServletException, IOException {
		List <Courses> mycourse = coursesdb.selectAllSubjects();
	request.setAttribute("listCourses", mycourse);
		RequestDispatcher dispatcher = request.getRequestDispatcher("student-
form.jsp");
		dispatcher.forward(request, response);
	}
	private void showNewFormTeacher(HttpServletRequest request,
HttpServletResponse response)
			throws ServletException, IOException {
		RequestDispatcher dispatcher = request.getRequestDispatcher("teacher-
form.jsp");
		dispatcher.forward(request, response);
	}

	private void showEditForm(HttpServletRequest request, HttpServletResponse
response)
			throws SQLException, ServletException, IOException {
		int id = Integer.parseInt(request.getParameter("id"));
		Students existingUser = studentDb.selectStudent(id);
		List <Courses> mycourse = coursesdb.selectAllSubjects();
	request.setAttribute("listCourses", mycourse);
		RequestDispatcher dispatcher = request.getRequestDispatcher("student-
form.jsp");
		request.setAttribute("student", existingUser);
		dispatcher.forward(request, response);

	}

	private void showEditClassRoom(HttpServletRequest request, HttpServletResponse
response)
			throws SQLException, ServletException, IOException {
		int id = Integer.parseInt(request.getParameter("id"));
		Classes existingUser = classesDb.selectSubject(id);
		RequestDispatcher dispatcher =
request.getRequestDispatcher("newClassRoom-form.jsp");
		List <Teachers> teachers = teacherDb.selectAllTeachers();
	request.setAttribute("teachersList", teachers);
	List <Courses> mycourse = coursesdb.selectAllSubjects();
	request.setAttribute("courses", mycourse);
		request.setAttribute("c", existingUser);
		dispatcher.forward(request, response);
	}

	private void showEditCourses(HttpServletRequest request, HttpServletResponse
response)
			throws SQLException, ServletException, IOException {
		int id = Integer.parseInt(request.getParameter("id"));
		Courses existingUser = coursesdb.selectSubject(id);
		RequestDispatcher dispatcher = request.getRequestDispatcher("courses-
form.jsp");
		request.setAttribute("c", existingUser);
		dispatcher.forward(request, response);

	}

	private void insertUser(HttpServletRequest request, HttpServletResponse response)
			throws SQLException, IOException {
		String name = request.getParameter("name");
		String email = request.getParameter("email");
		String country = request.getParameter("country");
		String course = request.getParameter("course");

		Students newStudent = new Students(name, email, country,course);
		studentDb.insertStudent(newStudent);
		response.sendRedirect("/LearnersAcademy/");
	}

	private void updateUser(HttpServletRequest request, HttpServletResponse
response)
			throws SQLException, IOException {
		int id = Integer.parseInt(request.getParameter("id"));
		String name = request.getParameter("name");
		String email = request.getParameter("email");
		String country = request.getParameter("country");
		String course = request.getParameter("course");


		Students book = new Students(id, name, email, country,course);
		studentDb.updateStudent(book);
		response.sendRedirect("/LearnersAcademy/");
	}

	private void deleteUser(HttpServletRequest request, HttpServletResponse response)
			throws SQLException, IOException {
		int id = Integer.parseInt(request.getParameter("id"));
		studentDb.deleteStudent(id);
		response.sendRedirect("/LearnersAcademy/");
	}

//-------------------------------

	private void showEditFormTeacher(HttpServletRequest request,
HttpServletResponse response)
			throws SQLException, ServletException, IOException {
		int id = Integer.parseInt(request.getParameter("id"));
		Teachers existingUser = teacherDb.selectTeacher(id);
		RequestDispatcher dispatcher = request.getRequestDispatcher("teacher-
form.jsp");
		request.setAttribute("teacher", existingUser);
		dispatcher.forward(request, response);

	}




	private void insertClassRoom(HttpServletRequest request, HttpServletResponse
response)
			throws SQLException, IOException {
		String name = request.getParameter("name");
		String duration = request.getParameter("duration");
		String teacher = request.getParameter("teacher");
		String[] courses = request.getParameterValues("courses");
		String finalResult="";
		List<String> list = new ArrayList<String>();
		list = Arrays.asList(courses);
		for(int i=0; i<list.size();i++) {
			finalResult+=(list.get(i));
			if(i!=list.size()-1) {
				finalResult+=(" , ");
			}

		}


		Classes c = new Classes(name, teacher, duration, finalResult);
		classesDb.insertSubject(c);
		response.sendRedirect("list-class");
	}



	private void insertCourse(HttpServletRequest request, HttpServletResponse
response)
```

```
                    throws SQLException, IOException {
                String course = request.getParameter("course");
                Courses c = new Courses(course);

                coursesdb.insertSubject(c);
                response.sendRedirect("list-courses");
        }

        private void insertUserTeacher(HttpServletRequest request, HttpServletResponse
response)
                    throws SQLException, IOException {
                String name = request.getParameter("name");
                String email = request.getParameter("email");
                String exp = request.getParameter("experience");
                Teachers newTeacher = new Teachers(name, email, exp);
                teacherDb.insertTeacher(newTeacher);
                response.sendRedirect("list-teacher");
        }

        private void updateUserTeacher(HttpServletRequest request, HttpServletResponse
response)
                    throws SQLException, IOException {
                int id = Integer.parseInt(request.getParameter("id"));
                String name = request.getParameter("name");
                String email = request.getParameter("email");
                String exp = request.getParameter("experience");

                Teachers t = new Teachers(id, name, email, exp);
                teacherDb.updateTeacher(t);
                response.sendRedirect("list-teacher");
        }

        private void updateClassRoom(HttpServletRequest request, HttpServletResponse
response)
                    throws SQLException, IOException {
                int id = Integer.parseInt(request.getParameter("id"));
                String name = request.getParameter("name");
                String teacher = request.getParameter("teacher");
                String dur = request.getParameter("duration");
                String[] courses = request.getParameterValues("courses");
                String finalResult="";
                List<String> list = new ArrayList<String>();
                list = Arrays.asList(courses);
                for(int i=0; i<list.size();i++) {
                        finalResult+=(list.get(i));
                        if(i!=list.size()-1) {
                                finalResult+=(" , ");
```

```
				}

			}
			System.out.println(finalResult);
			System.out.println("check final");

			Classes t = new Classes(id, name, teacher, dur,finalResult);
			classesDb.updateClasses(t);
			response.sendRedirect("list-class");
		}

		private void updateCourses(HttpServletRequest request, HttpServletResponse
response)
				throws SQLException, IOException {
			int id = Integer.parseInt(request.getParameter("id"));
			String course = request.getParameter("course");


			Courses t = new Courses(id, course);
			coursesdb.updateClasses(t);
			response.sendRedirect("list-courses");
		}

		private void deleteUserTeacher(HttpServletRequest request, HttpServletResponse
response)
				throws SQLException, IOException {
			int id = Integer.parseInt(request.getParameter("id"));
			teacherDb.deleteTeacher(id);
			response.sendRedirect("list-teacher");

		}
		private void deleteClassRoom(HttpServletRequest request, HttpServletResponse
response)
				throws SQLException, IOException {
			int id = Integer.parseInt(request.getParameter("id"));
			classesDb.deleteClasses(id);
			response.sendRedirect("list-class");

		}

		private void deleteCourses(HttpServletRequest request, HttpServletResponse
response)
				throws SQLException, IOException {
			int id = Integer.parseInt(request.getParameter("id"));
			coursesdb.deleteClasses(id);
			response.sendRedirect("list-courses");
```

```
	}
	/**
	 * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse
response)
	 */
	protected void doPost(HttpServletRequest request, HttpServletResponse response)
			throws ServletException, IOException {
		doGet(request, response);
	}

}
```

TeacherListServlet.java:

```
package com.learnersacademy.web;

import java.io.IOException;
import java.sql.SQLException;
import java.util.List;

import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.Cookie;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import com.learnersacademy.dao.Teachers_db;
import com.learnersacademy.model.Courses;
import com.learnersacademy.model.Teachers;


@WebServlet("/list-teacher")
public class TeacherListServlet extends HttpServlet {
	private static final long serialVersionUID = 1L;
	private Teachers_db teacherDb;

	public void init() {
		teacherDb = new Teachers_db();
	}

   /**
    * @see HttpServlet#HttpServlet()
    */
   public TeacherListServlet() {
      super();
```

```
        // TODO Auto-generated constructor stub
    }

            /**
             * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse
response)
             */
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
                        throws ServletException, IOException {
                String action = request.getServletPath();

                try {
                        switch (action) {
                        case "/newteacher":
                                showNewForm(request, response);
                                break;
                        case "/insertTeacher":
                                insertUser(request, response);
                                break;
                        case "/deleteteacher":
                                deleteUser(request, response);
                                break;
                        case "/editteacher":
                                showEditForm(request, response);
                                break;
                        case "/updateteacher":
                                updateUser(request, response);
                                break;
                        default:
                                listUser(request, response);
                                break;
                        }
                } catch (SQLException ex) {
                        throw new ServletException(ex);
                }
            }
    private void listUser(HttpServletRequest request, HttpServletResponse response)
                        throws SQLException, IOException, ServletException {
            Boolean login =false;
                Cookie c[]=request.getCookies();
                for(int i=0;i<c.length;i++){
                         if(c[i].getName().equals("login" ) && c[i].getValue().equals("true")) {
                                 System.out.println("Peace Bro");
                                 login=true;
                         }
                        }
                if(login) {
```

```
				List<Teachers> listUser = teacherDb.selectAllTeachers();
				request.setAttribute("listTeacher", listUser);
				RequestDispatcher dispatcher =
request.getRequestDispatcher("teacher-list.jsp");
				dispatcher.forward(request, response);
			}
			else {
				RequestDispatcher dispatcher =
request.getRequestDispatcher("adminLogin.jsp");
				dispatcher.forward(request, response);
			}

	}

	private void showNewForm(HttpServletRequest request, HttpServletResponse
response)
				throws ServletException, IOException {
			RequestDispatcher dispatcher = request.getRequestDispatcher("teacher-
form.jsp");
			dispatcher.forward(request, response);
	}

	private void showEditForm(HttpServletRequest request, HttpServletResponse
response)
				throws SQLException, ServletException, IOException {
			int id = Integer.parseInt(request.getParameter("id"));
			Teachers existingUser = teacherDb.selectTeacher(id);
			RequestDispatcher dispatcher = request.getRequestDispatcher("student-
form.jsp");
			request.setAttribute("student", existingUser);
			dispatcher.forward(request, response);

	}

	private void insertUser(HttpServletRequest request, HttpServletResponse response)
				throws SQLException, IOException {
			String name = request.getParameter("name");
			String email = request.getParameter("email");
			String exp = request.getParameter("experience");
			Teachers newTeacher = new Teachers(name, email, exp);
			teacherDb.insertTeacher(newTeacher);
			response.sendRedirect("list-teacher");
	}

	private void updateUser(HttpServletRequest request, HttpServletResponse
response)
				throws SQLException, IOException {
```

```
		int id = Integer.parseInt(request.getParameter("id"));
		String name = request.getParameter("name");
		String email = request.getParameter("email");
		String exp = request.getParameter("experience");

		Teachers t = new Teachers(id, name, email, exp);
		teacherDb.updateTeacher(t);
		response.sendRedirect("list-teacher");
	}

	private void deleteUser(HttpServletRequest request, HttpServletResponse response)
			throws SQLException, IOException {
		int id = Integer.parseInt(request.getParameter("id"));
		teacherDb.deleteTeacher(id);
		response.sendRedirect("list-teacher");

	}

	/**
	 * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse
response)
	 */
	protected void doPost(HttpServletRequest request, HttpServletResponse response)
			throws ServletException, IOException {
		doGet(request, response);
	}

}
```

JSP codes:

AdminLogin.jsp:

```
<%@ page language="java" contentType="text/html;
charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ page import="java.io.*,java.util.*,java.sql.*"%>
<%@ page
import="javax.servlet.http.*,javax.servlet.*" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/sql"
prefix="sql"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core"
prefix="c"%>
<!DOCTYPE html>
```

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html;
charset=UTF-8">
<title>Admin Login</title>
<link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/di
st/css/bootstrap.min.css" rel="stylesheet"
integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65Vohh
puuCOmLASjC" crossorigin="anonymous">

</head>
<style>
.container{
width: 30%;
    height: 30%;
    border-color: antiquewhite;
    border-style: solid;
    padding: 50px;
}

</style>
<body>
<div align ="center">
<br/>
<h1>Learners Academy </h1>
<br/><br/><br/><br/><br/>
<h2>Admin Login</h2>

<div class="container">
<form action ="verifylogin" method="post">
  <div class="mb-3">
    <label for="exampleInputEmail1" class="form-
label">User Name</label>
    <input  type="text" name="username" class="form-
control" id="exampleInputEmail1" aria-
describedby="emailHelp">
  </div>
  <div class="mb-3">
    <label for="exampleInputPassword1" class="form-
label">Password</label>
```

```
    <input type="password" name="password"
class="form-control" id="exampleInputPassword1">
  </div>
  <div class="mb-3 form-check">
    <button type="submit" class="btn btn-
primary">Submit</button>
  </div>

</form>
</div>




</div>
</body>
</html>
```

Class-list.jsp:

```
<%@ page language="java" contentType="text/html;
charset=UTF-8"
    pageEncoding="UTF-8"%>
    <%@ taglib
uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<!DOCTYPE html>
<html>
<head>
<link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/di
st/css/bootstrap.min.css" rel="stylesheet"
integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65Vohh
puuCOmLASjC" crossorigin="anonymous">
<meta charset="UTF-8">
<title>Learners Academy</title>

</head>
<style>
.body-container{
padding:30px;
margin:0;
margin-top:50px;
}
.nav-container{
padding-left:30px;

}

.container-fluid {
display:flex;

}

.nav_contain{
display:flex;
}

.logo{
    align-self: flex-end;
    font-weight: 600;
    margin-left: 10px;
    margin-right: 30px;
    letter-spacing: 4px;
    font-size: 30px;
    width:500px;

}


</style>
<body>
<header>



        <nav class="navbar navbar-light bg-light">

    <div class="nav_contain">
    <p class="logo">Learners Academy</p>

      <a href="<%=request.getContextPath()%>/"
                     class="nav-link"><button
class="btn btn-outline-secondary"
>Students</button></a>
        <a href="<%=request.getContextPath()%>/list-
teacher"
```

```
                     class="nav-link"><button
class="btn btn-outline-secondary"
>Teachers</button></a>
        <a href="<%=request.getContextPath()%>/list-
class"
                     class="nav-link"><button
class="btn btn-outline-secondary"
>Subjects</button></a>
        <a href="<%=request.getContextPath()%>/list-
courses"
                     class="nav-link"><button
class="btn btn-outline-secondary"
>Classes</button></a>
        <a href="<%=request.getContextPath()%>/list-
report"
                     class="nav-link"><button
class="btn btn-warning" >Class Report</button></a>
        <a href="<%=request.getContextPath()%>/login"
                     class="nav-link"><button
class="btn btn-danger" >Logout</button></a>

    </div>




</nav>
    </header>
    <br>



<div class="body-container">
    <div class="row">

        <div class="container">
            <h3 class="text-center">Subjects Master
List</h3>
            <hr>
            <div >

                <a
href="<%=request.getContextPath()%>/newclassroom"
class="btn btn-success">Add
```

```
                        New Subjects</a>
                </div>
                <br>
                <table class="table table-bordered">
                    <thead>
                        <tr>
                            <th>ID</th>
                            <th>Name</th>
                            <th>Teacher Handling</th>
                            <th>Duration</th>
                            <th>Class</th>
                            <th>Actions</th>
                        </tr>
                    </thead>
                    <tbody>

                        <c:forEach var="c"
items="${listClasses}">

                            <tr>
                                <td><c:out
value="${c.id}" /></td>
                                <td><c:out
value="${c.name}" /></td>
                            <td><c:out
value="${c.teacher}" /></td>
                            <td><c:out
value="${c.duration}" /></td>
                             <td><c:out
value="${c.courses}" /></td>

                              <td><a
href="editclassroom?id=<c:out value='${c.id}'
/>">Edit</a>

    &nbsp;&nbsp;&nbsp;&nbsp; <a

    href="deleteclassroom?id=<c:out value='${c.id}'
/>">Delete</a></td>
                            </tr>
                        </c:forEach>
                    </tbody>

                </table>
            </div>
        </div>
</div>

</body>
</html>
```

Courses-form.jsp:

```
<%@ page language="java" contentType="text/html;
charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core"
prefix="c"%>
<html>
<head>
<title>Learners Academy</title>
<link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/di
st/css/bootstrap.min.css" rel="stylesheet"
integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65Vohh
puuCOmLASjC" crossorigin="anonymous">
</head>
<style>
.nav-container{
padding-left:30px;

}

.container-fluid {
display:flex;

}

.nav_contain{
display:flex;
}
```

```
.logo{
    align-self: flex-end;
    font-weight: 600;
    margin-left: 10px;
    margin-right: 30px;
    letter-spacing: 4px;
    font-size: 30px;

}
#teacher{
    width: 364px;
    height: 38;
}

</style>
<body>

    <header>
    <nav class="navbar navbar-light bg-light">

    <div class="nav_contain">
    <p class="logo">Add New Course</p>
      <a href="<%=request.getContextPath()%>/list-
courses"
                     class="nav-link"><button
class="btn btn-outline-secondary" > ≤ Go
Back</button></a>


  </div>
  </nav>

    </header>
    <br>
    <div class="container col-md-5">
        <div class="card">
            <div class="card-body">
                <c:if test="${c != null}">
                    <form action="updatecourse"
method="post">
                </c:if>
                <c:if test="${c == null}">
```

```
                              <form action="insertcourse"
method="post">
                        </c:if>

                        <caption>
                            <h2>
                                <c:if test="${c != null}">
                            Edit Class
                        </c:if>
                                <c:if test="${c == null}">
                            Add New Class
                        </c:if>
                            </h2>
                        </caption>

                        <c:if test="${c != null}">
                            <input type="hidden" name="id"
value="<c:out value='${c.id}' />" />
                        </c:if>

                        <fieldset class="form-group">
                            <label>Course Name</label>
<input type="text"
                                value="<c:out
value='${c.course}' />" class="form-control"
                                name="course"
required="required">
                        </fieldset>





                        <button style="margin-top:5px"
type="submit" class="btn btn-success">Save</button>


                    </div>
                </div>
            </div>
</body>
</html>
```

List-courses.jsp:

```
<%@ page language="java" contentType="text/html;
charset=UTF-8"
    pageEncoding="UTF-8"%>
    <%@ taglib
uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<!DOCTYPE html>
<html>
<head>
<link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/di
st/css/bootstrap.min.css" rel="stylesheet"
integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65Vohh
puuCOmLASjC" crossorigin="anonymous">
<meta charset="UTF-8">
<title>Learners Academy</title>

</head>
<style>
.body-container{
padding:30px;
margin:0;
margin-top:50px;
}
.nav-container{
padding-left:30px;

}

.container-fluid {
display:flex;

}

.nav_contain{
display:flex;
}

.logo{
    align-self: flex-end;
    font-weight: 600;
    margin-left: 10px;
    margin-right: 30px;
    letter-spacing: 4px;
    font-size: 30px;
    width:500px;

}

</style>
<body>
<header>


        <nav class="navbar navbar-light bg-light">

    <div class="nav_contain">
    <p class="logo">Learners Academy</p>

      <a href="<%=request.getContextPath()%>/"
                    class="nav-link"><button
class="btn btn-outline-secondary"
>Students</button></a>
        <a href="<%=request.getContextPath()%>/list-
teacher"
                    class="nav-link"><button
class="btn btn-outline-secondary"
>Teachers</button></a>
        <a href="<%=request.getContextPath()%>/list-
class"
                    class="nav-link"><button
class="btn btn-outline-secondary"
>Subjects</button></a>
        <a href="<%=request.getContextPath()%>/list-
courses"
                    class="nav-link"><button
class="btn btn-outline-secondary"
>Classes</button></a>
        <a href="<%=request.getContextPath()%>/list-
report"
                    class="nav-link"><button
class="btn btn-warning" >Class Report</button></a>
        <a href="<%=request.getContextPath()%>/login"
```

```
                        class="nav-link"><button
class="btn btn-danger" >Logout</button></a>

    </div>



</nav>
    </header>
    <br>


<div class="body-container">
    <div class="row">

        <div class="container">
            <h3 class="text-center">Classes Master
List</h3>
            <hr>
            <div >

                <a
href="<%=request.getContextPath()%>/newcourse"
class="btn btn-success">Add New Class Room</a>
            </div>
            <br>
            <table class="table table-bordered">
                <thead>
                    <tr>
                        <th>ID</th>
                        <th>Class Name</th>
                        <th>Actions</th>
                    </tr>
                </thead>
                <tbody>

                    <c:forEach var="c"
items="${listCourses}">

                        <tr>
                            <td><c:out
value="${c.id}" /></td>
```

```
                                        <td><c:out 
value="${c.course}" /></td>

                                      <td><a 
href="editcourse?id=<c:out value='${c.id}' 
/>">Edit</a>

    &nbsp;&nbsp;&nbsp;&nbsp; <a

    href="deletecourse?id=<c:out value='${c.id}' 
/>">Delete</a></td>
                                    </tr>
                                </c:forEach>

                            </tbody>

                        </table>
                    </div>
                </div>
        </div>

</body>
</html>
```

newClassRoom-form.jsp:

```
<%@ page language="java" contentType="text/html; 
charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" 
prefix="c"%>
<html>
<head>
<title>Learners Academy</title>
<link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/di
st/css/bootstrap.min.css" rel="stylesheet"
integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65Vohh
puuCOmLASjC" crossorigin="anonymous">
```

```
</head>
<style>
.nav-container{
padding-left:30px;

}

.container-fluid {
display:flex;

}

.nav_contain{
display:flex;
}

.logo{
    align-self: flex-end;
    font-weight: 600;
    margin-left: 10px;
    margin-right: 30px;
    letter-spacing: 4px;
    font-size: 30px;

}

#teacher{
    width: 364px;
    height: 38;
}

</style>
<body>

    <header>
    <nav class="navbar navbar-light bg-light">

    <div class="nav_contain">
    <p class="logo">Add New Class Room</p>
      <a href="<%=request.getContextPath()%>/list-
class"
                     class="nav-link"><button
class="btn btn-outline-secondary" > ≤ Go
Back</button></a>
```

```
  </div>
  </nav>

    </header>
    <br>
    <div class="container col-md-5">
        <div class="card">
            <div class="card-body">
                <c:if test="${c != null}">
                    <form action="updateclassroom"
method="post">
                </c:if>
                <c:if test="${c == null}">
                    <form action="insertclassroom"
method="post">
                </c:if>

                <caption>
                    <h2>
                        <c:if test="${c != null}">
                    Edit Subject
                </c:if>
                        <c:if test="${c == null}">
                    Add New Subject
                </c:if>
                    </h2>
                </caption>

                <c:if test="${c != null}">
                    <input type="hidden" name="id"
value="<c:out value='${c.id}' />" />
                </c:if>

                <fieldset class="form-group">
                    <label>Subject Name</label>
<input type="text"
                        value="<c:out
value='${c.name}' />" class="form-control"
                        name="name"
required="required">
                </fieldset>
```

```
                    <fieldset class="form-group">
                        <label>Duration</label> <input
type="text"
                            value="<c:out
value='${c.duration}' />" class="form-control"
                            name="duration">
                    </fieldset>

                    <fieldset class="form-group">
                         <label for="teacher">Choose a
Faculty</label>
                         <br/>
                              <select name="teacher"
id="teacher" >
                                <c:forEach var="list"
items="${teachersList}">
                                 <option value="<c:out
value="${list.name}" />"><c:out value="${list.name}"
/></option>
                                </c:forEach>

                              </select>
                    </fieldset>

                    <fieldset class="form-group">
                         <label for="course">Pick
Courses</label>
                         <br/>
                         <c:forEach var="list"
items="${courses}">
                         <input type="checkbox"
id="<c:out value="${list.course}" />" name="courses"
value="<c:out value="${list.course}" />">
                         <label for="<c:out
value="${list.course}" />"><c:out
value="${list.course}" /></label><br>
                         </c:forEach>
                        </fieldset>

                    <button style="margin-top:5px"
type="submit" class="btn btn-success">Save</button>
```

```
            </div>
        </div>
    </div>
</body>
</html>
```

Report.jsp:

```
<%@ page language="java" contentType="text/html;
charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ page import="java.io.*,java.util.*,java.sql.*"%>
<%@ page
import="javax.servlet.http.*,javax.servlet.*" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/sql"
prefix="sql"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core"
prefix="c"%>

<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Class Report</title>
<link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/di
st/css/bootstrap.min.css" rel="stylesheet"
integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65Vohh
puuCOmLASjC" crossorigin="anonymous">

<style>
.body-container{
padding:30px;
margin:0;
margin-top:50px;
display: flex;
flex-direction: column;
align-items: center;
}
.nav-container{
```

```
padding-left:30px;

}
.container-fluid {
display:flex;

}
.nav_contain{
display:flex;
}
.logo{
    align-self: flex-end;
    font-weight: 600;
    margin-left: 10px;
    margin-right: 30px;
    letter-spacing: 4px;
    font-size: 30px;
    width:500px;

}

</style>
</head>
<body>
<header>
<%!

public String getData(String i){

    return i;


}

%>


<nav class="navbar navbar-light bg-light">

    <div class="nav_contain">
    <p class="logo">Learners Academy</p>
```

```
      <a href="<%=request.getContextPath()%>/"
                        class="nav-link"><button
class="btn btn-outline-secondary"
>Students</button></a>
        <a href="<%=request.getContextPath()%>/list-
teacher"
                        class="nav-link"><button
class="btn btn-outline-secondary"
>Teachers</button></a>
        <a href="<%=request.getContextPath()%>/list-
class"
                        class="nav-link"><button
class="btn btn-outline-secondary"
>Subjects</button></a>
        <a href="<%=request.getContextPath()%>/list-
courses"
                        class="nav-link"><button
class="btn btn-outline-secondary"
>Classes</button></a>
        <a href="<%=request.getContextPath()%>/list-
report"
                        class="nav-link"><button
class="btn btn-warning" >Class Report</button></a>
        <a href="<%=request.getContextPath()%>/login"
                        class="nav-link"><button
class="btn btn-danger" >Logout</button></a>

    </div>




</nav>
</header>

<div class="body-container">




                 <sql:setDataSource var="db"
driver="com.mysql.jdbc.Driver"
url="jdbc:mysql://localhost:3306/learnersacademy"
user="root" password="root@123"/>
```

```
                 <sql:query var="rs"
dataSource="${db}">
                 SELECT * FROM courses
                 </sql:query>
                 <c:forEach var="main"
items="${rs.rows}">

                 <h2><c:out value="${main.course}" />
class</h2>


                 <sql:query var="rs2"
dataSource="${db}">
                 SELECT * FROM students where course
like "${main.course}";
                 </sql:query>
                 <sql:query var="rs3"
dataSource="${db}">

                       SELECT * FROM classes where
courses like "%${main.course}%"

                 </sql:query>
                 <h3>Students List</h3>
                 <table class="table table-bordered">
                 <thead>
                     <tr>
                         <th>ID</th>
                         <th>Name</th>
                         <th>Email</th>
                         <th>Class Name</th>
                         <th>Address</th>

                     </tr>
                 </thead>
                 <c:forEach var="list"
items="${rs2.rows}">


                 <tbody>
                 <tr>
```

```
                                    <td><c:out
value="${list.id}" /></td>
                                    <td><c:out
value="${list.name}" /></td>
                                    <td><c:out
value="${list.email}" /></td>
                                    <td><c:out
value="${list.course}" /></td>
                                    <td><c:out
value="${list.country}" /></td>

                                </tr>
                                </tbody>


                        </c:forEach>
                        </table>
                        <br/>

                        <h2>Faculty List and Subject
List</h2>

                        <table class="table table-bordered">
                        <thead>
                            <tr>
                                <th>ID</th>
                                <th>Subject Name</th>
                                <th>Faculty Name</th>
                                <th>Duration</th>
                                <th>Other Classes Handling
by Faculty</th>

                            </tr>
                        </thead>
                        <c:forEach var="l"
items="${rs3.rows}">


                        <tbody>
                        <tr>
                                    <td><c:out
value="${l.id}" /></td>
```

```
                                        <td><c:out
value="${l.name}" /></td>
                                        <td><c:out
value="${l.teacher}" /></td>
                                        <td><c:out
value="${l.duration}" /></td>
                                        <td><c:out
value="${l.courses}" /></td>


                                </tr>
                                </tbody>




                        </c:forEach>
                        </table>
                        <br/>

                        </c:forEach>
                        <br/>


</div>

</body>
</html>
```

Student-form.jsp:

```
<%@ page language="java" contentType="text/html;
charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core"
prefix="c"%>
<html>
<head>
<title>Learners Academy</title>
```

```
<link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/di
st/css/bootstrap.min.css" rel="stylesheet"
integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65Vohh
puuCOmLASjC" crossorigin="anonymous">
</head>
<style>
.nav-container{
padding-left:30px;

}

.container-fluid {
display:flex;

}


.nav_contain{
display:flex;
}

.logo{
    align-self: flex-end;
    font-weight: 600;
    margin-left: 10px;
    margin-right: 30px;
    letter-spacing: 4px;
    font-size: 30px;

}

#course{
    width: 364px;
    height: 38;
}


</style>
<body>

    <header>
    <nav class="navbar navbar-light bg-light">

    <div class="nav_contain">
    <p class="logo">Learners Academy</p>
      <a href="<%=request.getContextPath()%>/"
```

```
                        class="nav-link"><button
class="btn btn-outline-secondary" > ≤ Go
Back</button></a>


  </div>
  </nav>

    </header>
    <br>
    <div class="container col-md-5">
        <div class="card">
            <div class="card-body">
                <c:if test="${student != null}">
                    <form action="update"
method="post">
                </c:if>
                <c:if test="${student == null}">
                    <form action="insert"
method="post">
                </c:if>

                <caption>
                    <h2>
                        <c:if test="${student !=
null}">
                    Edit Student
                </c:if>
                        <c:if test="${student ==
null}">
                    Add New Student
                </c:if>
                    </h2>
                </caption>

                <c:if test="${student != null}">
                    <input type="hidden" name="id"
value="<c:out value='${student.id}' />" />
                </c:if>

                <fieldset class="form-group">
```

```
                              <label>Student Name</label>
<input type="text"
                                        value="<c:out
value='${student.name}' />" class="form-control"
                                        name="name"
required="required">
                          </fieldset>

                          <fieldset class="form-group">
                              <label>Student Email</label>
<input type="text"
                                        value="<c:out
value='${student.email}' />" class="form-control"
                                        name="email">
                          </fieldset>

                          <fieldset class="form-group">
                              <label>Student Address</label>
<input type="text"
                                        value="<c:out
value='${student.country}' />" class="form-control"
                                        name="country">
                          </fieldset>

                          <fieldset class="form-group">
                               <label for="course">Pick a
Class</label>
                               <br/>
                                    <select name="course"
id="course" >
                                       <c:forEach var="list"
items="${listCourses}">
                                        <option value="<c:out
value="${list.course}" />"><c:out
value="${list.course}" /></option>
                                       </c:forEach>

                                    </select>
                          </fieldset>

                          <button style="margin-top:5px"
type="submit" class="btn btn-success">Save</button>
```

```
                    </form>

                </div>
            </div>
        </div>
</body>
</html>
```

Student-list.jsp:

```
<%@ page language="java" contentType="text/html;
charset=UTF-8"
    pageEncoding="UTF-8"%>
    <%@ taglib
uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<!DOCTYPE html>
<html>
<head>
<link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/di
st/css/bootstrap.min.css" rel="stylesheet"
integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65Vohh
puuCOmLASjC" crossorigin="anonymous">
<meta charset="UTF-8">
<title>Learners Academy</title>

</head>
<style>
.body-container{
padding:30px;
margin:0;
margin-top:50px;
}
.nav-container{
padding-left:30px;

}
.container-fluid {
display:flex;

}
```

```
.nav_contain{
display:flex;
}
.logo{
    align-self: flex-end;
    font-weight: 600;
    margin-left: 10px;
    margin-right: 30px;
    letter-spacing: 4px;
    font-size: 30px;
    width:500px;

}

</style>
<body>
<header>


        <nav class="navbar navbar-light bg-light">

    <div class="nav_contain">
    <p class="logo">Learners Academy</p>

      <a href="<%=request.getContextPath()%>/"
                     class="nav-link"><button
class="btn btn-outline-secondary"
>Students</button></a>
        <a href="<%=request.getContextPath()%>/list-
teacher"
                     class="nav-link"><button
class="btn btn-outline-secondary"
>Teachers</button></a>
        <a href="<%=request.getContextPath()%>/list-
class"
                     class="nav-link"><button
class="btn btn-outline-secondary"
>Subjects</button></a>
        <a href="<%=request.getContextPath()%>/list-
courses"
```

```
                     class="nav-link"><button
class="btn btn-outline-secondary"
>Classes</button></a>
        <a href="<%=request.getContextPath()%>/list-
report"
                     class="nav-link"><button
class="btn btn-warning" >Class Report</button></a>


        <a href="<%=request.getContextPath()%>/login"
                     class="nav-link"><button
class="btn btn-danger" >Logout</button></a>

    </div>




</nav>
    </header>
    <br>


<div class="body-container">
    <div class="row">

        <div class="container">
            <h3 class="text-center">Students Master
List</h3>
            <hr>
            <div >

                <a
href="<%=request.getContextPath()%>/new" class="btn
btn-success">Add
                     New Student</a>
            </div>
            <br>
            <table class="table table-bordered">
                <thead>
                    <tr>
                        <th>ID</th>
                        <th>Name</th>
                        <th>Email</th>
                        <th>Class Name</th>
                        <th>Address</th>
                        <th>Actions</th>
                    </tr>
                </thead>
                <tbody>

                    <c:forEach var="student"
items="${listStudent}">

                        <tr>
                            <td><c:out
value="${student.id}" /></td>
                            <td><c:out
value="${student.name}" /></td>
                            <td><c:out
value="${student.email}" /></td>
                            <td><c:out
value="${student.course}" /></td>
                            <td><c:out
value="${student.country}" /></td>
                            <td><a
href="edit?id=<c:out value='${student.id}'
/>">Edit</a>

    &nbsp;&nbsp;&nbsp;&nbsp; <a

    href="delete?id=<c:out value='${student.id}'
/>">Delete</a></td>
                        </tr>
                    </c:forEach>

                </tbody>

            </table>
        </div>
    </div>
</div>

</body>
</html>
```

Teacher-form.jsp:

```
<%@ page language="java" contentType="text/html;
charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core"
prefix="c"%>
<html>
<head>
<title>Learners Academy</title>
<link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/di
st/css/bootstrap.min.css" rel="stylesheet"
integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65Vohh
puuCOmLASjC" crossorigin="anonymous">
</head>
<style>
.nav-container{
padding-left:30px;

}
.container-fluid {
display:flex;

}
.nav_contain{
display:flex;
}
.logo{
    align-self: flex-end;
    font-weight: 600;
    margin-left: 10px;
    margin-right: 30px;
    letter-spacing: 4px;
    font-size: 30px;

}
```

```
</style>
<body>

    <header>
    <nav class="navbar navbar-light bg-light">

    <div class="nav_contain">
    <p class="logo">Learners Academy</p>
      <a href="<%=request.getContextPath()%>/list-
teacher"
                    class="nav-link"><button
class="btn btn-outline-secondary" > ≤ Go
Back</button></a>


  </div>
  </nav>

    </header>
    <br>
    <div class="container col-md-5">
        <div class="card">
            <div class="card-body">
                <c:if test="${teacher != null}">
                    <form action="updateteacher"
method="post">
                </c:if>
                <c:if test="${teacher == null}">
                    <form action="insertteacher"
method="post">
                </c:if>

                <caption>
                    <h2>
                        <c:if test="${teacher !=
null}">
                    Edit Teacher
                </c:if>
                        <c:if test="${teacher ==
null}">
                    Add New Teacher
                </c:if>
```

```
                    </h2>
                </caption>

                <c:if test="${teacher != null}">
                    <input type="hidden" name="id"
value="<c:out value='${teacher.id}' />" />
                </c:if>

                <fieldset class="form-group">
                    <label> Name</label> <input
type="text"
                        value="<c:out
value='${teacher.name}' />" class="form-control"
                        name="name"
required="required">
                </fieldset>

                <fieldset class="form-group">
                    <label> Email</label> <input
type="text"
                        value="<c:out
value='${teacher.email}' />" class="form-control"
                        name="email"
required="required">
                </fieldset>

                <fieldset class="form-group">
                    <label> Experience</label>
<input type="text"
                        value="<c:out
value='${teacher.experience}' />" class="form-
control"
                        name="experience"
required="required">
                </fieldset>

                <button style="margin-top:5px"
type="submit" class="btn btn-success">Save</button>
                </form>

            </div>
        </div>
```

```
    </div>
</body>
</html>
```

Teacher-list.jsp:

```
<%@ page language="java" contentType="text/html;
charset=UTF-8"
    pageEncoding="UTF-8"%>
    <%@ taglib
uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<!DOCTYPE html>
<html>
<head>
<link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/di
st/css/bootstrap.min.css" rel="stylesheet"
integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65Vohh
puuCOmLASjC" crossorigin="anonymous">
<meta charset="UTF-8">
<title>Learners Academy</title>

</head>
<style>
.body-container{
padding:30px;
margin:0;
margin-top:50px;
}

.nav-container{
padding-left:30px;


}

.container-fluid {
display:flex;


}

.nav_contain{
display:flex;
}

.logo{
    align-self: flex-end;
    font-weight: 600;
    margin-left: 10px;
    margin-right: 30px;
    letter-spacing: 4px;
    font-size: 30px;
    width:500px;

}

</style>
<body>
<header>


        <nav class="navbar navbar-light bg-light">

    <div class="nav_contain">
    <p class="logo">Learners Academy</p>

      <a href="<%=request.getContextPath()%>/"
                      class="nav-link"><button
class="btn btn-outline-secondary"
>Students</button></a>
        <a href="<%=request.getContextPath()%>/list-
teacher"
                      class="nav-link"><button
class="btn btn-outline-secondary"
>Teachers</button></a>
        <a href="<%=request.getContextPath()%>/list-
class"
                      class="nav-link"><button
class="btn btn-outline-secondary"
>Subjects</button></a>
        <a href="<%=request.getContextPath()%>/list-
courses"
                      class="nav-link"><button
class="btn btn-outline-secondary"
>Classes</button></a>
        <a href="<%=request.getContextPath()%>/list-
report"
```

```
                     class="nav-link"><button
class="btn btn-warning" >Class Report</button></a>
        <a href="<%=request.getContextPath()%>/login"
                     class="nav-link"><button
class="btn btn-danger" >Logout</button></a>

    </div>




</nav>
    </header>
    <br>


<div class="body-container">
    <div class="row">

        <div class="container">
            <h3 class="text-center">Teachers Master
List</h3>
            <hr>
            <div >

                <a
href="<%=request.getContextPath()%>/newteacher"
class="btn btn-success">Add
                     New Teacher</a>
            </div>
            <br>
            <table class="table table-bordered">
                <thead>
                    <tr>
                        <th>ID</th>
                        <th>Name</th>
                        <th>Email</th>
                        <th>Experience</th>
                        <th>Actions</th>
                    </tr>
                </thead>
                <tbody>
```

```
                    <c:forEach var="teacher"
items="${listTeacher}">

                        <tr>
                            <td><c:out
value="${teacher.id}" /></td>
                            <td><c:out
value="${teacher.name}" /></td>
                            <td><c:out
value="${teacher.email}" /></td>
                            <td><c:out
value="${teacher.experience}" /></td>
                            <td><a
href="editteacher?id=<c:out value='${teacher.id}'
/>">Edit</a>

    &nbsp;&nbsp;&nbsp;&nbsp; <a

    href="deleteteacher?id=<c:out
value='${teacher.id}' />">Delete</a></td>
                        </tr>
                    </c:forEach>

                </tbody>

            </table>
        </div>
    </div>
</div>

</body>
</html>
```