

## MỤC TIÊU:

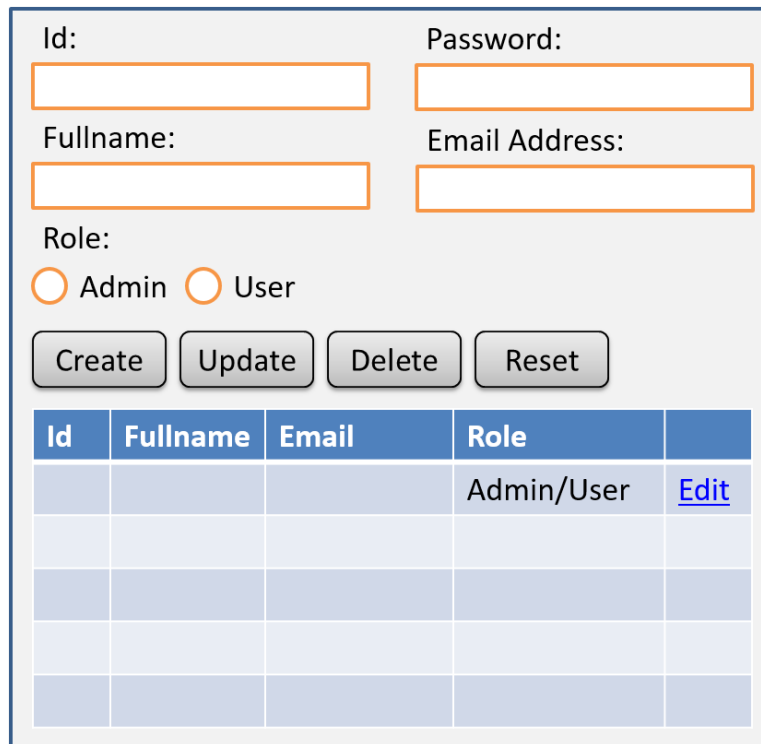
Kết thúc bài thực hành này bạn có khả năng

- Áp dụng Hibernate/JPA để xây dựng ứng dụng CRUD

## YÊU CẦU:

Xây dựng ứng dụng thực hiện quản lý thông tin User theo yêu cầu được mô tả như các hình sau:

- ✓ Giao diện ứng dụng



The mockup shows a user management interface. It includes input fields for Id, Password, Fullname, and Email Address. There are radio buttons for Role (Admin, User) and four buttons: Create, Update, Delete, and Reset. Below these is a table with columns Id, Fullname, Email, Role, and an Edit link.

Id	Fullname	Email	Role	
			Admin/User	<a href="#">Edit</a>

- ✓ Xử lý tương tác người dùng

Sự kiện	Điều khiển sự kiện
Page.Open	<ul style="list-style-type: none"> <li>• Hiện thị bảng</li> <li>• Xóa trắng form</li> </ul>
Create.Click	<ul style="list-style-type: none"> <li>• Thêm mới</li> <li>• Hiện thị lại bảng</li> <li>• Xóa trắng form</li> </ul>
Update.Click	<ul style="list-style-type: none"> <li>• Cập nhật</li> <li>• Hiện thị lại bảng</li> <li>• Giữ nguyên dữ liệu trên form</li> </ul>
Delete.Click	<ul style="list-style-type: none"> <li>• Xóa mục theo id trên form</li> <li>• Hiện thị lại bảng</li> <li>• Xóa trắng form</li> </ul>
Reset.Click	<ul style="list-style-type: none"> <li>• Xóa trắng form</li> <li>• Hiện thị lại bảng</li> </ul>
Edit.Click	<ul style="list-style-type: none"> <li>• Hiện thị mục chọn lên form</li> <li>• Hiện thị lại bảng</li> </ul>

## PHẦN I

### BÀI 1 (2 ĐIỂM)

- ✓ Tạo và tổ chức dự án có cấu hình tương tự Lab1
  - CSDL PolyOE, table Users
  - Persistence.xml
  - Pom.xml
  - Lớp thực thể User (di chuyển vào package poly.entity)
- ✓ Bổ sung khai báo các thư viện phụ thuộc cần thiết khác
  - BeanUtils để đơn giản việc nhận tham số
  - JSTL để lập trình trên JSP

### BÀI 2 (2 ĐIỂM)

- ✓ Tạo Servlet poly.servlet.UserCRUDServlet có cấu trúc như sau

```
@WebServlet({
    "/user/crud/index",
    "/user/crud/edit/*",
    "/user/crud/create",
    "/user/crud/update",
    "/user/crud/delete",
    "/user/crud/reset"
```

```

    })
    public class UserCRUDServlet extends HttpServlet {
        @Override
        protected void service(HttpServletRequest req, HttpServletResponse resp) throws
        ServletException, IOException {
            User form = new User();
            try {
                BeanUtils.populate(form, req.getParameterMap());
            } catch (IllegalAccessException | InvocationTargetException e) {
                e.printStackTrace();
            }
            String message = "Enter user information";
            String path = req.getServletPath();
            if (path.contains("edit")) {
                String id = req.getPathInfo().substring(1);
                message = "Edit: " + id;
            } else if (path.contains("create")) {
                message = "Create: " + form.getId();
                form = new User();
            } else if (path.contains("update")) {
                message = "Update: " + form.getId();
            } else if (path.contains("delete")) {
                message = "Delete: " + form.getId();
                form = new User();
            } else if (path.contains("reset")) {
                form = new User();
            }
            List<User> list = List.of(new User(), new User(), new User());

            req.setAttribute("message", message);
            req.setAttribute("user", form);
            req.setAttribute("users", list);
            req.getRequestDispatcher("/pages/user-crud.jsp").forward(req, resp);
        }
    }
}

```

✓ Tạo trang /pages/user-crud.jsp có cấu trúc như sau

```

<%@ page pageEncoding="utf-8"%>
<%@ taglib uri="http://java.sun.com/jstl/core_rt" prefix="c" %>
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title>User CRUD</title>

```

```

</head>
<body>
    <i>${message}</i>
    <c:url var="url" value="/user/crud"/>
    <form method="post">...</form>
    <hr>
    <table>    ...</table>
</body>
</html>

<form method="post">
    <input name="id" value="${user.id}"><br>
    <input name="password" type="password" value="${user.password}"><br>
    <input name="fullname" value="${user.fullname}"><br>
    <input name="email" value="${user.email}"><br>
    <input name="gender" type="radio" value="true" ${user.admin?'checked':''}>
Male
    <input name="gender" type="radio" value="false" ${user.admin?'':'checked'}>
Female
    <hr>
    <button formaction="${url}/create">Create</button>
    <button formaction="${url}/update">Update</button>
    <button formaction="${url}/delete">Delete</button>
    <button formaction="${url}/reset">Reset</button>
</form>

<table>
    <thead>
        <tr>
            <th>Id</th>
            <th>Password</th>
            <th>Fullname</th>
            <th>Email</th>
            <th>Role</th>
            <th></th>
        </tr>
    </thead>
    <tbody>
    <c:forEach var="u" items="${users}">
        <tr>
            <td>${u.id}</td>
            <td>${u.password}</td>
            <td>${u.fullname}</td>
            <td>${u.email}</td>
            <td>${u.admin?'Admin':'User'}</td>
            <td><a href="${url}/edit/${u.id}">Edit</a></td>
        </tr>
    </tbody>
</table>

```

```

        </tr>
    </c:forEach>
</tbody>
</table>

```

- ✓ Chạy servlet và nhấp vào các nút và liên kết trên table để cảm nhận sự hoạt động của ứng dụng.

## PHẦN II

### BÀI 3 (2 ĐIỂM)

- ✓ Tạo lớp tiện ích XJPA

```

public class XJPA {
    private static EntityManagerFactory factory;
    static {
        factory = Persistence.createEntityManagerFactory("PolyOE");
    }
    public static EntityManager getEntityManager(){
        return factory.createEntityManager();
    }
}

```

- ✓ Tạo UserDAO và thiết kế các chức năng truy xuất dữ liệu

```

public interface UserDAO {
    /**Truy vấn tất cả*/
    List<User> findAll();
    /**Truy vấn theo mã*/
    User findById(String id);
    /**Thêm mới*/
    void create(User item);
    /**Cập nhật*/
    void update(User item);
    /**Xóa theo mã*/
    void deleteById(String id);
}

```

- ✓ Tạo lớp UserDAOImpl và cài đặt mã nguồn cho UserDAO

```

public class UserDAOImpl implements UserDAO {
    EntityManager em = XJPA.getEntityManager();
    @Override
    protected void finalize() throws Throwable {
        em.close();
    }
}

```

```
@Override
public List<User> findAll() {
    String jpql = "SELECT o FROM User o";
    TypedQuery<User> query = em.createQuery(jpql, User.class);
    return query.getResultList();
}

@Override
public User findById(String id) {
    return em.find(User.class, id);
}

@Override
public void create(User entity) {
    try {
        em.getTransaction().begin();
        em.persist(entity);
        em.getTransaction().commit();
    } catch (Exception e) {
        em.getTransaction().rollback();
    }
}

@Override
public void update(User entity) {
    try {
        em.getTransaction().begin();
        em.merge(entity);
        em.getTransaction().commit();
    } catch (Exception e) {
        em.getTransaction().rollback();
    }
}

@Override
public void deleteById(String id) {
    User entity = em.find(User.class, id);
    try {
        em.getTransaction().begin();
        em.remove(entity);
        em.getTransaction().commit();
    } catch (Exception e) {
        em.getTransaction().rollback();
    }
}
}
```

**BÀI 4 (2 ĐIỂM)**

Hiệu chỉnh Servlet để thực hiện CRUD theo hướng dẫn sau

- ✓ Khai báo ***UserDAO dao = new UserDAOImpl();*** vào đầu phương thức service
- ✓ Bổ sung ***form = dao.findById(id);*** vào cuối khối lệnh xử lý edit để truy vấn user theo id khi nhấp liên kết Edit trên bảng
- ✓ Bổ sung ***dao.create(form);*** vào đầu khối lệnh xử lý create để thực hiện thêm User mới vào CSDL khi nhấp nút Create
- ✓ Bổ sung ***dao.update(form);*** vào đầu khối lệnh xử lý update để thực hiện cập nhật User vào CSDL khi nhấp nút Update
- ✓ Bổ sung ***dao.deleteById(form.getId());*** vào đầu khối lệnh xử lý delete để thực hiện xóa User khỏi CSDL khi nhấp nút Delete
- ✓ Thay thế ***List.of(new User(), new User(), new User());*** bằng ***dao.findAll()*** để truy vấn tất cả User từ CSDL
- ✓ Hiệu chỉnh thông báo (***message***) phù hợp với các hành động tương tác.
- ✓ Chạy ứng dụng hoàn chỉnh

**BÀI 5 GIẢNG VIÊN CHO THÊM (2 ĐIỂM)**