Exemple **complet** d'une application **JEE MVC sans framework** (Servlet + JSP + JDBC) pour gérer une table user(id, nom, email) avec :

- CRUD (Lister / Ajouter / Modifier / Supprimer)
- **Validation d'email** côté serveur (Regex)
- **Recherche par nom** (recherche partielle)
- Code prêt à déployer sur **Tomcat** (ou tout conteneur servlet)

1) Schéma MySQL

Db.sql

```
CREATE DATABASE demo_db CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci;
USE demo_db;

CREATE TABLE users (
  id INT AUTO_INCREMENT PRIMARY KEY,
  nom VARCHAR(100) NOT NULL,
  email VARCHAR(150) NOT NULL
);
```

Il faut Installer le driver MySQL (mysql-connector-java-x.x.x.jar) dans WEB-INF/lib ou configure Maven.

2) Structure du projet (simplifiée)

Architecture

```
YourApp/
 ├─ src/
 |  └─ main/
 |     └─ java/
 |        └─ com.example/
 |           ├─ model/User.java
 |           ├─ dao/UserDAO.java
 |           └─ web/UserServlet.java
 ├─ src/main/webapp/
 |   ├─ WEB-INF/
 |   |   ├─ web.xml
 |   |   └─ lib/ (mysql driver si pas Maven)
 |   ├─ users/
 |   |   ├─ list.jsp
 |   |   └─ form.jsp
 |   └─ index.jsp (redirect vers /users)
 └─ pom.xml (optionnel)
```

3) Fichier de configuration (connexion)

Il faut mettre les paramètres directement dans le DAO ou dans le cas d'un framework dans un fichier simple db.properties (optionnel) .

4) Code Java

model/User.java

```
package com.example.model;

public class User {
   private int id;
   private String nom;
   private String email;

   public User() {}
   public User(int id, String nom, String email) {
      this.id = id; this.nom = nom; this.email = email;
   }
   public User(String nom, String email) {
      this.nom = nom; this.email = email;
   }
   public int getId() { return id; }
   public void setId(int id) { this.id = id; }
   public String getNom() { return nom; }
   public void setNom(String nom) { this.nom = nom; }
   public String getEmail() { return email; }
   public void setEmail(String email) { this.email = email; }
}
```

dao/UserDAO.java

```
package com.example.dao;

import com.example.model.User;
import java.sql.*;
import java.util.*;

public class UserDAO {
   private String jdbcURL =
"jdbc:mysql://localhost:3306/demo_db?useSSL=false&serverTimezone=UTC";
   private String jdbcUser = "root";
   private String jdbcPass = "your_password";

   private static final String INSERT_SQL = "INSERT INTO users (nom, email) VALUES (?, ?)";
   private static final String SELECT_BY_ID = "SELECT id, nom, email FROM users WHERE id = ?";
   private static final String SELECT_ALL = "SELECT id, nom, email FROM users ORDER BY id DESC";
   private static final String DELETE_SQL = "DELETE FROM users WHERE id = ?";
   private static final String UPDATE_SQL = "UPDATE users SET nom = ?, email = ? WHERE id = ?";
```

```
    private static final String SEARCH_BY_NAME = "SELECT id, nom, email FROM users WHERE nom
LIKE ? ORDER BY id DESC";

    public UserDAO() {
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
        } catch (ClassNotFoundException e) {
            e.printStackTrace();
        }
    }

    private Connection getConnection() throws SQLException {
        return DriverManager.getConnection(jdbcURL, jdbcUser, jdbcPass);
    }

    public void insertUser(User user) throws SQLException {
        try (Connection conn = getConnection();
             PreparedStatement ps = conn.prepareStatement(INSERT_SQL)) {
            ps.setString(1, user.getNom());
            ps.setString(2, user.getEmail());
            ps.executeUpdate();
        }
    }

    public User selectUser(int id) throws SQLException {
        User user = null;
        try (Connection conn = getConnection();
             PreparedStatement ps = conn.prepareStatement(SELECT_BY_ID)) {
            ps.setInt(1, id);
            try (ResultSet rs = ps.executeQuery()) {
                if (rs.next()) user = new User(rs.getInt("id"), rs.getString("nom"), rs.getString("email"));
            }
        }
        return user;
    }

    public List<User> selectAllUsers() throws SQLException {
        List<User> list = new ArrayList<>();
        try (Connection conn = getConnection();
             PreparedStatement ps = conn.prepareStatement(SELECT_ALL);
             ResultSet rs = ps.executeQuery()) {
            while (rs.next()) {
                list.add(new User(rs.getInt("id"), rs.getString("nom"), rs.getString("email")));
            }
        }
        return list;
    }
```

```
    public boolean deleteUser(int id) throws SQLException {
        try (Connection conn = getConnection();
             PreparedStatement ps = conn.prepareStatement(DELETE_SQL)) {
            ps.setInt(1, id);
            return ps.executeUpdate() > 0;
        }
    }

    public boolean updateUser(User user) throws SQLException {
        try (Connection conn = getConnection();
             PreparedStatement ps = conn.prepareStatement(UPDATE_SQL)) {
            ps.setString(1, user.getNom());
            ps.setString(2, user.getEmail());
            ps.setInt(3, user.getId());
            return ps.executeUpdate() > 0;
        }
    }

    public List<User> searchByName(String namePattern) throws SQLException {
        List<User> list = new ArrayList<>();
        try (Connection conn = getConnection();
             PreparedStatement ps = conn.prepareStatement(SEARCH_BY_NAME)) {
            ps.setString(1, "%" + namePattern + "%");
            try (ResultSet rs = ps.executeQuery()) {
                while (rs.next()) {
                    list.add(new User(rs.getInt("id"), rs.getString("nom"), rs.getString("email")));
                }
            }
        }
        return list;
    }
}
```

web/UserServlet.java

```
package com.example.web;

import com.example.dao.UserDAO;
import com.example.model.User;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.*;
import java.io.IOException;
import java.sql.SQLException;
import java.util.List;
import java.util.regex.Pattern;
```

```
@WebServlet("/users")
public class UserServlet extends HttpServlet {
   private UserDAO userDAO;
   // regex pour email
   private static final String EMAIL_REGEX = "^[A-Za-z0-9+_.-]+@[A-Za-z0-9.-]+\\.[A-Za-z]{2,}$";
   private static final Pattern EMAIL_PATTERN = Pattern.compile(EMAIL_REGEX);

   @Override
   public void init() {
      userDAO = new UserDAO();
   }

   @Override
   protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws
ServletException, IOException {
      String action = req.getParameter("action");
      if (action == null) action = "list";

      try {
         switch (action) {
            case "new":
               showNewForm(req, resp);
               break;
            case "insert":
               insertUser(req, resp);
               break;
            case "delete":
               deleteUser(req, resp);
               break;
            case "edit":
               showEditForm(req, resp);
               break;
            case "update":
               updateUser(req, resp);
               break;
            case "search":
               searchUser(req, resp);
               break;
            default:
               listUser(req, resp);
               break;
         }
      } catch (SQLException ex) {
         throw new ServletException(ex);
      }
   }
```

```
    // GET and POST both handled via doGet for simplicity; real app would separate
    @Override
    protected void doPost(HttpServletRequest req, HttpServletResponse resp) throws
ServletException, IOException {
        doGet(req, resp);
    }

    private void listUser(HttpServletRequest req, HttpServletResponse resp) throws SQLException,
ServletException, IOException {
        List<User> list = userDAO.selectAllUsers();
        req.setAttribute("userList", list);
        req.getRequestDispatcher("/users/list.jsp").forward(req, resp);
    }

    private void showNewForm(HttpServletRequest req, HttpServletResponse resp) throws
ServletException, IOException {
        req.setAttribute("action", "insert");
        req.getRequestDispatcher("/users/form.jsp").forward(req, resp);
    }

    private void showEditForm(HttpServletRequest req, HttpServletResponse resp) throws
SQLException, ServletException, IOException {
        int id = Integer.parseInt(req.getParameter("id"));
        User existing = userDAO.selectUser(id);
        req.setAttribute("user", existing);
        req.setAttribute("action", "update");
        req.getRequestDispatcher("/users/form.jsp").forward(req, resp);
    }

    private void insertUser(HttpServletRequest req, HttpServletResponse resp) throws SQLException,
IOException, ServletException {
        String nom = req.getParameter("nom");
        String email = req.getParameter("email");

        String error = validate(nom, email);
        if (error != null) {
            req.setAttribute("error", error);
            req.setAttribute("nom", nom);
            req.setAttribute("email", email);
            req.setAttribute("action", "insert");
            req.getRequestDispatcher("/users/form.jsp").forward(req, resp);
            return;
        }

        User newUser = new User(nom, email);
        userDAO.insertUser(newUser);
        resp.sendRedirect(req.getContextPath() + "/users");
    }
```

```
    private void updateUser(HttpServletRequest req, HttpServletResponse resp) throws
SQLException, IOException, ServletException {
        int id = Integer.parseInt(req.getParameter("id"));
        String nom = req.getParameter("nom");
        String email = req.getParameter("email");

        String error = validate(nom, email);
        if (error != null) {
            req.setAttribute("error", error);
            User user = new User(id, nom, email);
            req.setAttribute("user", user);
            req.setAttribute("action", "update");
            req.getRequestDispatcher("/users/form.jsp").forward(req, resp);
            return;
        }

        User user = new User(id, nom, email);
        userDAO.updateUser(user);
        resp.sendRedirect(req.getContextPath() + "/users");
    }

    private void deleteUser(HttpServletRequest req, HttpServletResponse resp) throws
SQLException, IOException {
        int id = Integer.parseInt(req.getParameter("id"));
        userDAO.deleteUser(id);
        resp.sendRedirect(req.getContextPath() + "/users");
    }

    private void searchUser(HttpServletRequest req, HttpServletResponse resp) throws
SQLException, ServletException, IOException {
        String name = req.getParameter("q");
        List<User> list = userDAO.searchByName(name == null ? "" : name);
        req.setAttribute("userList", list);
        req.setAttribute("q", name);
        req.getRequestDispatcher("/users/list.jsp").forward(req, resp);
    }

    private String validate(String nom, String email) {
        if (nom == null || nom.trim().isEmpty()) return "Le nom est obligatoire.";
        if (email == null || email.trim().isEmpty()) return "L'email est obligatoire.";
        if (!EMAIL_PATTERN.matcher(email).matches()) return "Format d'email invalide.";
        return null;
    }
}
```

Remarque : pour simplicité doGet gère actions via action parameter. Tu peux séparer POST/GET en production.

5) JSP : affichage et formulaire

users/list.jsp

```
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<%@ page import="com.example.model.User" %>
<%@ page import="java.util.List" %>
<%
   List<User> userList = (List<User>) request.getAttribute("userList");
   String q = request.getParameter("q");
%>
<!doctype html>
<html>
<head>
   <meta charset="utf-8">
   <title>Liste des utilisateurs</title>
   <style>
      table { border-collapse: collapse; width: 80%; margin: 20px auto;}
      th, td { border: 1px solid #ccc; padding: 8px; text-align:left;}
      a { text-decoration: none; color: blue; }
   </style>
</head>
<body>
   <div style="width:80%; margin:auto;">
      <h2>Utilisateurs</h2>
      <form action="${pageContext.request.contextPath}/users" method="get">
         <input type="hidden" name="action" value="search"/>
         <input type="text" name="q" placeholder="Recherche par nom" value="<%= q != null ? q :
"" %>"/>
         <button type="submit">Rechercher</button>
         <a href="${pageContext.request.contextPath}/users?action=new" style="margin-
left:20px;">Ajouter un utilisateur</a>
```

```
    </form>

    <table>
      <thead><tr><th>ID</th><th>Nom</th><th>Email</th><th>Actions</th></tr></thead>
      <tbody>
      <%
        if (userList != null) {
          for (User u : userList) {
      %>
        <tr>
          <td><%= u.getId() %></td>
          <td><%= u.getNom() %></td>
          <td><%= u.getEmail() %></td>
          <td>
            <a href="${pageContext.request.contextPath}/users?action=edit&id=<%= u.getId()
%>">Modifier</a> |
            <a href="${pageContext.request.contextPath}/users?action=delete&id=<%= u.getId()
%>"
              onclick="return confirm('Supprimer cet utilisateur ?');">Supprimer</a>
          </td>
        </tr>
      <%
          }
        }
      %>
      </tbody>
    </table>
  </div>
</body>
</html>
```

users/form.jsp

```
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<%@ page import="com.example.model.User" %>
<%
  String action = (String) request.getAttribute("action");
  User user = (User) request.getAttribute("user");
  if (action == null) action = "insert";
  String nom = request.getParameter("nom");
  String email = request.getParameter("email");
  if (user != null) { nom = user.getNom(); email = user.getEmail(); }
  String error = (String) request.getAttribute("error");
%>
<!doctype html>
<html>
<head>
  <meta charset="utf-8">
  <title><%= ("update".equals(action) ? "Modifier" : "Ajouter") %> Utilisateur</title>
  <style> label { display:block; margin-top:8px; } </style>
</head>
<body>
  <div style="width:400px; margin:40px auto;">
    <h2><%= ("update".equals(action) ? "Modifier" : "Ajouter") %> Utilisateur</h2>
    <% if (error != null) { %>
      <div style="color:red;"><%= error %></div>
    <% } %>

    <form action="${pageContext.request.contextPath}/users" method="post">
      <input type="hidden" name="action" value="<%= action %>"/>
      <% if ("update".equals(action)) { %>
        <input type="hidden" name="id" value="<%= user.getId() %>"/>
      <% } %>
      <label>Nom</label>
      <input type="text" name="nom" value="<%= nom != null ? nom : "" %>" required />

      <label>Email</label>
      <input type="email" name="email" value="<%= email != null ? email : "" %>" required />

      <div style="margin-top:12px;">
        <button type="submit"><%= ("update".equals(action) ? "Mettre à jour" : "Ajouter")
%></button>
        <a href="${pageContext.request.contextPath}/users">Annuler</a>
      </div>
    </form>
  </div>
</body>
</html>
```

6) web.xml (optionnel si @WebServlet utilisé)

Si tu veux déclarer dans web.xml (ou si ton conteneur n'accepte pas @WebServlet) :

web.xml

```
<web-app xmlns="http://xmlns.jcp.org/xml/ns/javaee" version="4.0">
  <servlet>
    <servlet-name>UserServlet</servlet-name>
    <servlet-class>com.example.web.UserServlet</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>UserServlet</servlet-name>
    <url-pattern>/users</url-pattern>
  </servlet-mapping>
</web-app>
```

---

7) Déploiement

1. Construis l'archive WAR (via IDE ou Maven).
2. Vérifie que le driver JDBC (MySQL Connector) est dans WEB-INF/lib.
3. Déploie sur Tomcat (ex : webapps/YourApp.war).
4. Navigate to: http://localhost:8080/YourApp/users