

APUNTES L1 Pag2

APUNTES L2 Pag22

Back pt1 DAO&SESION

hasta servlet para SALUDO (cuando el usuario se logee)

..... **Pag23**

Back pt2 DAO & SUBJECT SESION *para mostrar listado
de cursos al iniciar Sesion*

Pag26

FRONT (JSP)

Pag29

Algunas Plantillas

NOTEPAD

EXPLORADOR

pag33

RESUMEN

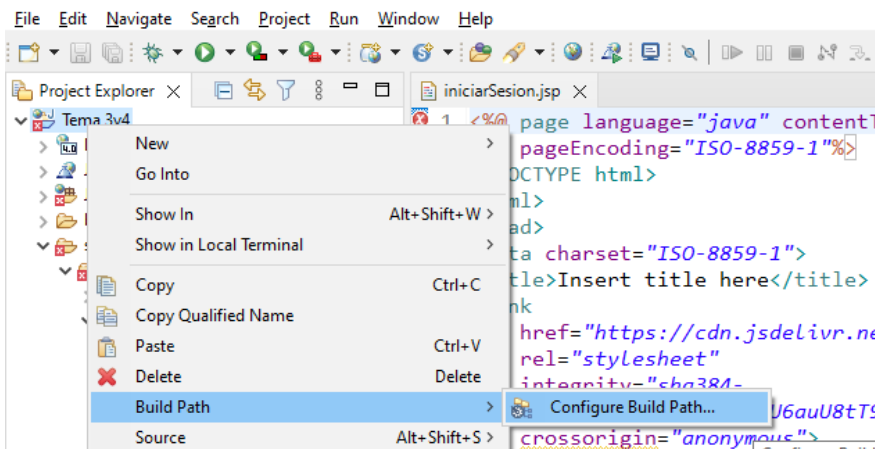
pag34

APUNTES EF

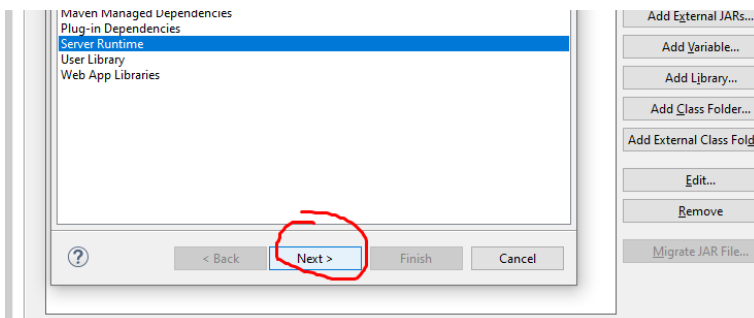
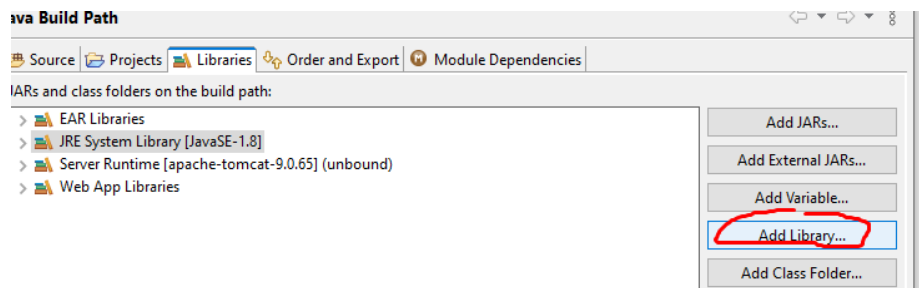
pag36

APUNTES L1

PRIMERO: CREA SERVER TOMCAT

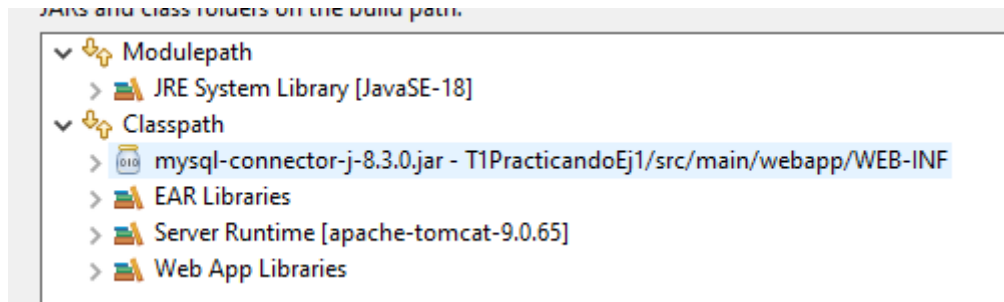


PARAR SOLUCIONAR ERROR
CON <% %>



+ apply and close

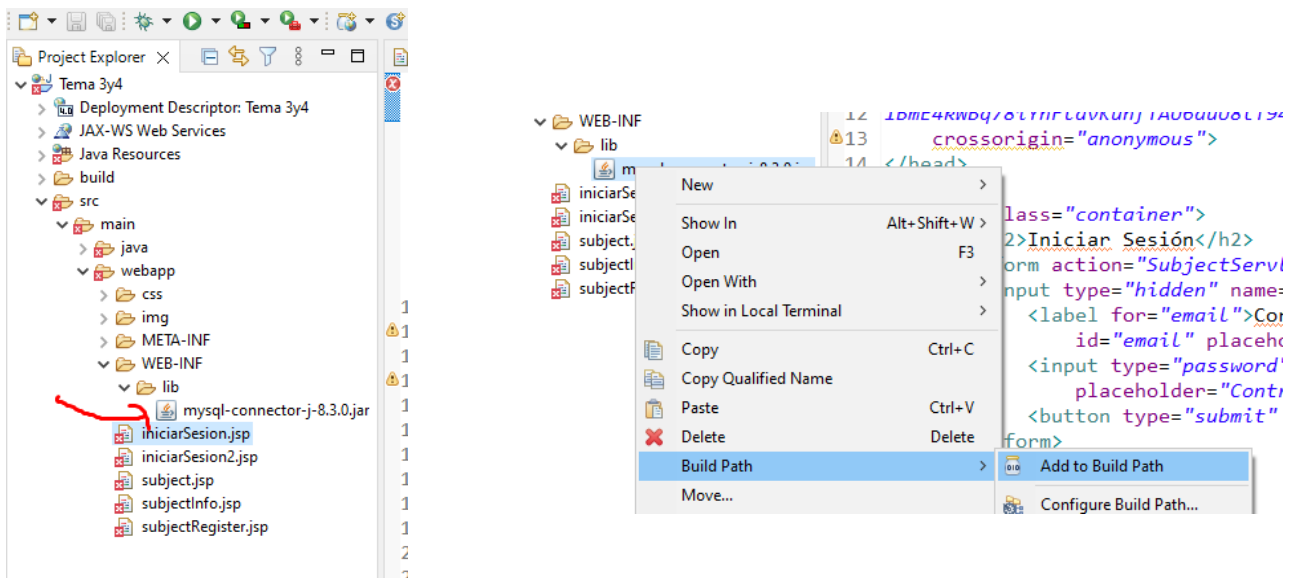
SI TE APARECEN 2 APARTADOS DE LIBRARIES



Lo colocas dentro del **Classpath**

PARA AÑADIR JAR DE SQL CONNECTION.JAR

Arrastramos el JAR a la carpeta LIB...

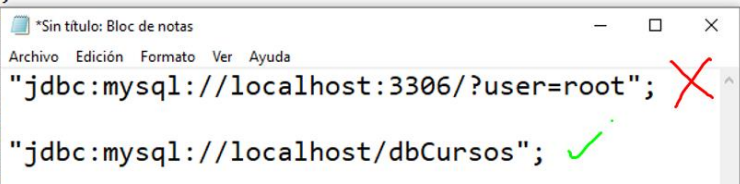


MySQLConection.java

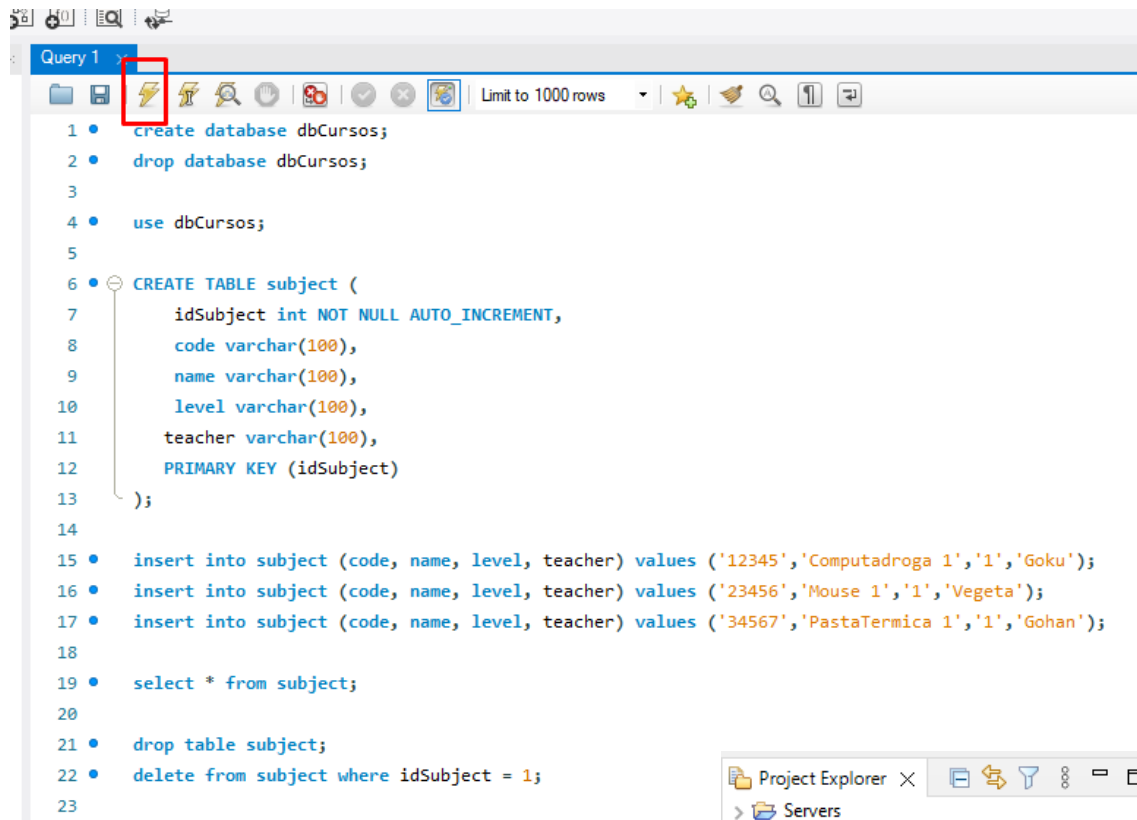
```
mysqlconex.java
1 package util;
2
3 import java.sql.Connection;
4
5
6
7 public class MySqlConexion {
8     public static Connection getConexion() {
9         Connection con = null;
10        try {
11            Class.forName("com.mysql.cj.jdbc.Driver").newInstance();
12            String url = "jdbc:mysql://localhost/dbCursos?useSSL=false&useTimezone=true&serverTimezone=UTC";
13            String usr = "root";
14            String psw = "mysql";
15            con = DriverManager.getConnection(url,usr,psw);
16        } catch (ClassNotFoundException e) {
17            // TODO: handle exception
18            System.out.println("Error >> Driver no Instalado!!"+e.getMessage());
19        } catch (SQLException e) {
20            // TODO: handle exception
21            System.out.println("Error >> de conexión con la BD"+e.getMessage());
22        } catch (Exception e) {
23            // TODO: handle exception
24            System.out.println("Error >> general: "+ e.getMessage());
25        }
26        return con;
27    }
28 }
29
```

```
MySqlConexion.java X
1 package util;
2
3 import java.sql.Connection;
4
5
6
7 public class MySqlConexion {
8     public static Connection getConexion() {
9         Connection con = null;
10        try {
11            Class.forName("com.mysql.cj.jdbc.Driver").newInstance();
12            String url = "jdbc:mysql://localhost/dbCursos?useSSL=false&useTimezone=true&serverTimezone=UTC";
13            String usr = "root";
14            String psw = "mysql";
15            con = DriverManager.getConnection(url,usr,psw);
16        } catch (ClassNotFoundException e) {
17            // TODO: handle exception
18            System.out.println("Error >> Driver no Instalado!!"+e.getMessage());
19        } catch (SQLException e) {
20            // TODO: handle exception
21            System.out.println("Error >> de conexión con la BD"+e.getMessage());
22        } catch (Exception e) {
23            // TODO: handle exception
24            System.out.println("Error >> general: "+ e.getMessage());
25        }
26        return con;
27    }
28 }
29
```

```
try {
    Class.forName("com.mysql.cj.jdbc.Driver").newInstance();
    String url = "jdbc:mysql://localhost/dbCursos?useSSL=false&useTimezone=true&serverTimezone=UTC";
    String usr = "root";
    String psw = "mysql";
    con = DriverManager.getConnection(url,usr,psw);
} catch (ClassNotFoundException e) {
    // TODO: handle exception
    System.out.println("Error >> Driver no Instalado!!"+e.getMessage());
} catch (SQLException e) {
    // TODO: handle exception
    System.out.println("Error >> de conexión con la BD"+e.getMessage());
} catch (Exception e) {
    // TODO: handle exception
    System.out.println("Error >> general: "+ e.getMessage());
}
```

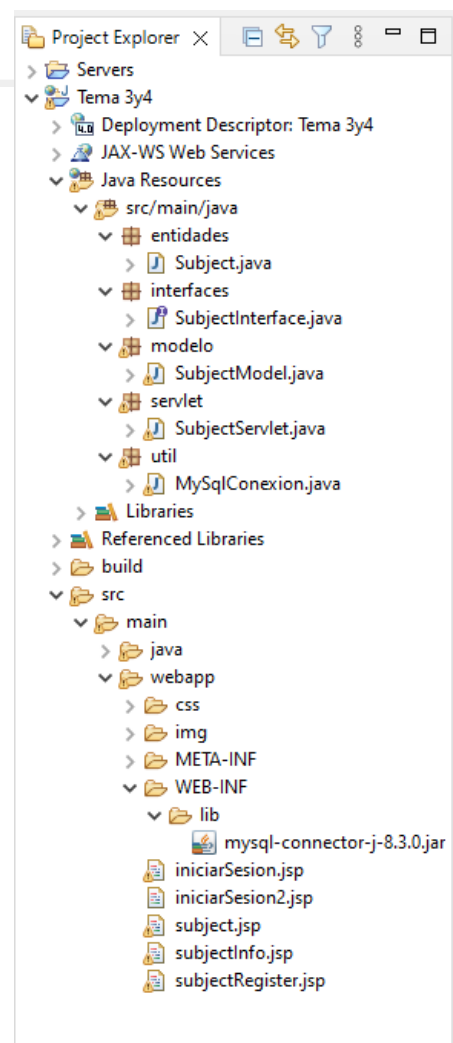


PARA CREAR LA BASE DE DATOS



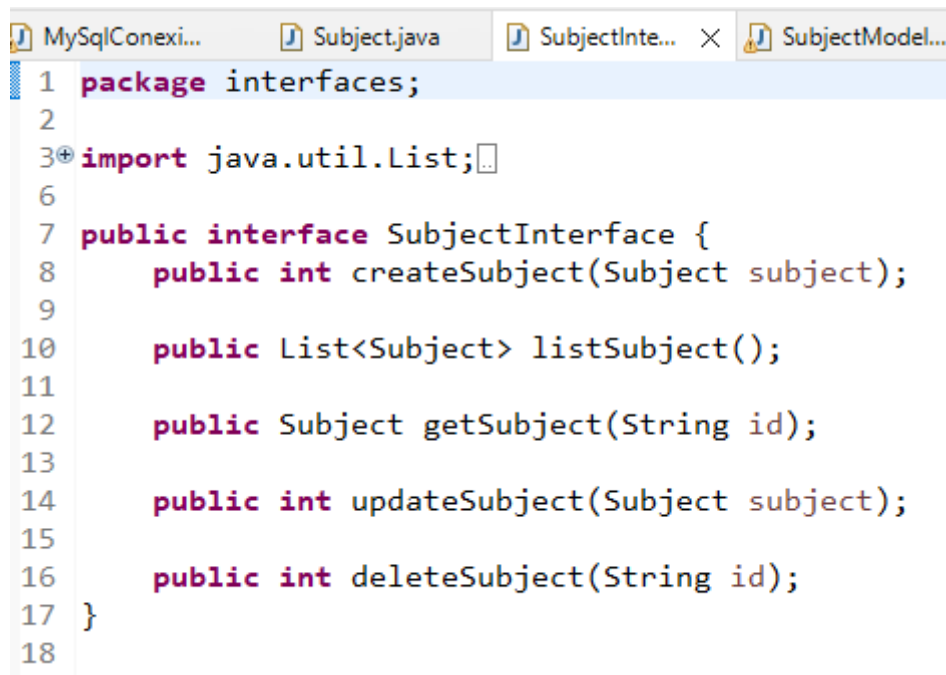
```
Query 1
1 • create database dbCursos;
2 • drop database dbCursos;
3
4 • use dbCursos;
5
6 • CREATE TABLE subject (
7     idSubject int NOT NULL AUTO_INCREMENT,
8     code varchar(100),
9     name varchar(100),
10    level varchar(100),
11    teacher varchar(100),
12    PRIMARY KEY (idSubject)
13 );
14
15 • insert into subject (code, name, level, teacher) values ('12345','Computadrogena 1','1','Goku');
16 • insert into subject (code, name, level, teacher) values ('23456','Mouse 1','1','Vegeta');
17 • insert into subject (code, name, level, teacher) values ('34567','PastaTermica 1','1','Gohan');
18
19 • select * from subject;
20
21 • drop table subject;
22 • delete from subject where idSubject = 1;
23
```

EXPLORADOR COMPLETO DEL PROYECTO



PASITOS

1. Estructurar del proyecto (carpetas, packages, implementar .JAR, solucionar error)
2. MySQL WORKBENCH
3. MySqlConnection.java
4. Entidades -> atributos + getters&Setters
5. Interfaces



```
1 package interfaces;
2
3 import java.util.List;
4
5
6
7 public interface SubjectInterface {
8     public int createSubject(Subject subject);
9
10    public List<Subject> listSubject();
11
12    public Subject getSubject(String id);
13
14    public int updateSubject(Subject subject);
15
16    public int deleteSubject(String id);
17 }
18
```

- a. método create retorna int y recibe Entidad
- b. método read(list) retorna List<Entidad>
- c. método update retorna int y recibe Entidad
- d. método delete retorna int y recibe String id
- e. método get retorna Entidad y recibe String id

6. Model

Implementa los métodos del interfaz.

PARA EJECUTAR UN

rs = psm.executeQuery();

value = psm.executeUpdate();

Primero tienes que llenar la variable PreparedStatement con

psm.setString(1,id);

psm.setString(1, subject.getCode());

a. Para método create

```
@Override
public int createSubject(Subject subject) {
    // TODO Auto-generated method stub
    int value = 0;
    Connection cn= null;
    PreparedStatement psm=null;
    try {
        cn= MySqlConnection.getConnection();
        String sql="INSERT INTO Subject VALUES (null, ?,?,?,?)";
        psm=cn.prepareStatement(sql);
        psm.setString(1, subject.getCode());
        psm.setString(2, subject.getName());
        psm.setString(3, subject.getLevel());
        psm.setString(4, subject.getTeacher());

        value=psm.executeUpdate();
    } catch (Exception e) {
        // TODO: handle exception
        e.printStackTrace();
    }finally {
        try {
            if(psm !=null) psm.close();
            if(cn != null) cn.close();
        } catch (Exception e) {
            // TODO: handle exception
            e.printStackTrace();
        }
    }
    return 0;
}
```

b. Para método list

```
@Override
public List<Subject> listSubject() {
    // TODO Auto-generated method stub
    List<Subject> listSubject = new ArrayList<Subject>();
    Connection cn=null;
    PreparedStatement psm=null;
    ResultSet rs=null;

    try {
        cn = MySqlConnection.getConnection();
        String sql="SELECT * FROM Subject";
        psm = cn.prepareStatement(sql);
        rs=psm.executeQuery();
        while (rs.next()) {
            Subject subj = new Subject();
            subj.setId(rs.getString("idSubject"));
            subj.setCode(rs.getString("code"));
            subj.setName(rs.getString("name"));
            subj.setLevel(rs.getString("level"));
            subj.setTeacher(rs.getString("teacher"));
            listSubject.add(subj);
        }
    } catch (Exception e) {
        // TODO: handle exception
        e.printStackTrace();
    } finally {
        try {
            if(rs != null) rs.close();
            if(psm != null) psm.close();
            if(cn != null) cn.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
    return listSubject;
}
```

c. Para método update

```
@Override
public int updateSubject(Subject subject) {
    // TODO Auto-generated method stub
    int value = 0;
    Connection cn= null;
    PreparedStatement psm=null;
    try {
        cn= MySqlConnection.getConnection();
        String sql="UPDATE Subject SET code=?,
            + "name=?,level=?,teacher=? where idSubject=?";
        psm=cn.prepareStatement(sql);
        psm.setString(1, subject.getCode());
        psm.setString(2, subject.getName());
        psm.setString(3, subject.getLevel());
        psm.setString(4, subject.getTeacher());
        psm.setString(5, subject.getId());

        value=psm.executeUpdate();
    } catch (Exception e) {
        // TODO: handle exception
        e.printStackTrace();
    } finally {
        try {
            if(psm !=null) psm.close();
            if(cn != null) cn.close();
        } catch (Exception e) {
            // TODO: handle exception
            e.printStackTrace();
        }
    }
    return 0;
}
```


d. Para el método delete

```
@Override
public int deleteSubject(String id) {
    // TODO Auto-generated method stub
    System.out.println("id dentro del model: "+id);
    int value = 0;
    Connection cn= null;
    PreparedStatement psm=null;
    try {
        cn= MySqlConnection.getConnection();
        String sql="DELETE FROM Subject where idSubject=?";
        psm=cn.prepareStatement(sql);
        psm.setString(1, id);

        value=psm.executeUpdate();
    } catch (Exception e) {
        // TODO: handle exception
        e.printStackTrace();
    }finally {
        try {
            if(psm !=null) psm.close();
            if(cn != null) cn.close();
        } catch (Exception e) {
            // TODO: handle exception
            e.printStackTrace();
        }
    }
    return 0;
}
```

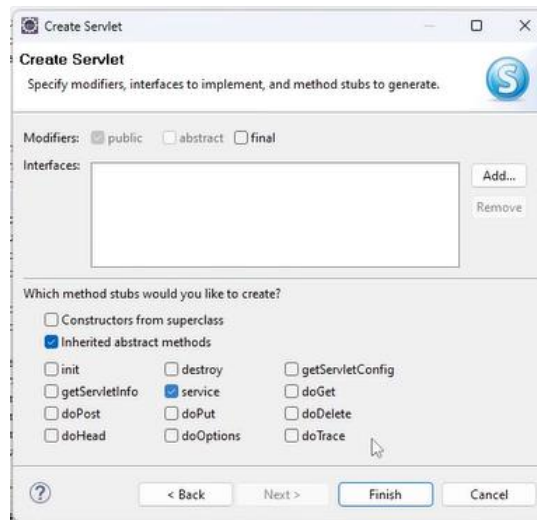
e. Para el método get

```
@Override
public Subject getSubject(String id) {
    // TODO Auto-generated method stub
    Subject subject=null;
    Connection cn=null;
    PreparedStatement psm=null;
    ResultSet rs=null;

    try {
        cn = MySqlConnection.getConnection();
        String sql="SELECT * FROM subject where idSubject=?";
        psm = cn.prepareStatement(sql);
        psm.setString(1, id);
        rs=psm.executeQuery();
        if (rs.next()) {
            subject = new Subject();
            subject.setId(rs.getString("idSubject"));
            subject.setCode(rs.getString("code"));
            subject.setName(rs.getString("name"));
            subject.setLevel(rs.getString("level"));
            subject.setTeacher(rs.getString("teacher"));
        }
    } catch (Exception e) {
        // TODO: handle exception
        e.printStackTrace();
    }finally {
        try {
            if(rs != null) rs.close();
            if(psm != null) psm.close();
            if(cn != null) cn.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
    return subject;
}
```

7. Servlet

CUANDO CREES EL SERVLET tienes q configurar unas cosas:



Así tiene que estar

El servlet interactúa entre **la base de datos (MODEL) y los .jps** mediante el poderoso **String type** que recibirá las request y redireccionará el request hacia el método CRUD adecuado.

SERVICE

```
27  */
28  protected void service(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
29      // TODO Auto-generated method stub
30      String type = request.getParameter("type");
31      System.out.println("El parámetro es: " + type);
32      switch(type) {
33          case "register": registerCurso(request,response); break;
34          case "list": listCurso(request,response); break;
35          case "edit": editCurso(request,response); break;
36          case "delete": deleteCurso(request,response); break;
37          case "info": getCurso(request,response); break;
38          default:
39              request.setAttribute("mensaje", "Ocurrió un error");
40              request.getRequestDispatcher("curso.jsp").forward(request, response);
41      }
42  }
```

Los métodos CRUD aquí se identificarán con el type de la sgte forma:

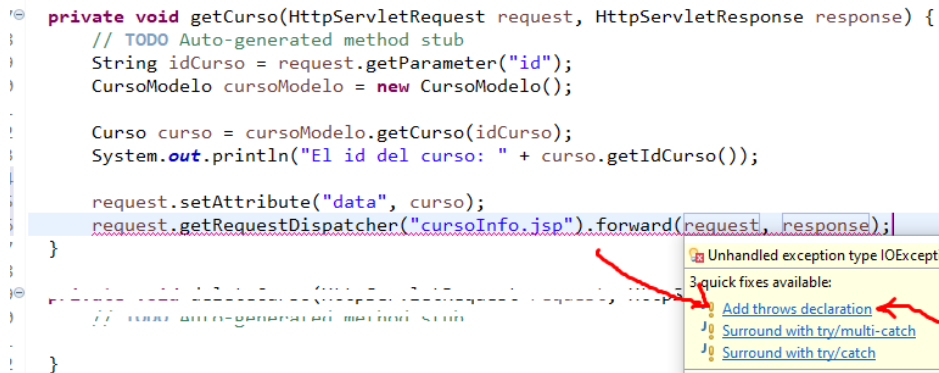
C	"register"	registerSubject
R	"list"	listSubject
U	"edit"	editSubject
D	"delete"	deleteSubject
get	"info"	getSubject

Los métodos los autogeneras con los errores que te saldrán dps de q hayas acabado con el método service.

Y luego de acabar de implementar cada método, te saldrá un error que autocompletará el nombre del método:

getCurso

```
private void getCurso(HttpServletRequest request, HttpServletResponse response) {  
    // TODO Auto-generated method stub  
    String idCurso = request.getParameter("id");  
    CursoModelo cursoModelo = new CursoModelo();  
  
    Curso curso = cursoModelo.getCurso(idCurso);  
    System.out.println("El id del curso: " + curso.getIdCurso());  
  
    request.setAttribute("data", curso);  
    request.getRequestDispatcher("cursoInfo.jsp").forward(request, response);  
}  
  
// TODO Auto-generated method stub  
}
```



deleteCurso

```
private void deleteSubject(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
    // TODO Auto-generated method stub  
    String id = request.getParameter("id");  
    System.out.println("Imprime subject : "+id);  
    SubjectModel subjectModel = new SubjectModel();  
    int value = subjectModel.deleteSubject(id);  
  
    if(value == 0) {  
        System.out.println("Entró al list subject");  
        listSubject(request, response);  
    } else {  
        request.setAttribute("mensaje", "Ocurrió un problema");  
        request.getRequestDispatcher("subject.jsp");  
    }  
}
```

editCurso

```
private void editSubject(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
    // TODO Auto-generated method stub  
    //Entradas  
    String code = request.getParameter("txtCode");  
    String name = request.getParameter("txtName");  
    String level = request.getParameter("txtLevel");  
    String teacher = request.getParameter("txtTeacher");  
    String id = request.getParameter("idSubject");  
  
    System.out.println("Imprime code : "+code);  
    System.out.println("Imprime name : "+name);  
    System.out.println("Imprime level : "+level);  
    System.out.println("Imprime teacher : "+teacher);  
    System.out.println("Imprime id : "+id);  
  
    //Creamos objeto  
    Subject subject = new Subject();  
    subject.setCode(code);  
    subject.setName(name);  
    subject.setLevel(level);  
    subject.setTeacher(teacher);  
    subject.setId(id);  
  
    //Procesos  
    SubjectModel model = new SubjectModel();  
    int value = model.updateSubject(subject);  
  
    if(value == 0) {  
        System.out.println("Entró al list subject");  
        listSubject(request, response);  
    } else {  
        request.setAttribute("mensaje", "Ocurrió un problema");  
        request.getRequestDispatcher("subject.jsp");  
    }  
}
```

listCurso

```
private void listSubject(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
    // TODO Auto-generated method stub  
    SubjectModel subjectModel = new SubjectModel();  
    List<Subject> data=subjectModel.listSubject();  
    request.setAttribute("data", data);  
    request.getRequestDispatcher("subject.jsp").forward(request, response);  
}
```

registerCurso

```
private void registerCurso(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
    // TODO Auto-generated method stub  
    String code = request.getParameter("txtCode");  
    String name = request.getParameter("txtName");  
    String level = request.getParameter("txtLevel");  
    String teacher = request.getParameter("txtTeacher");  
  
    Curso curso = new Curso();  
    curso.setCode(code);  
    curso.setName(name);  
    curso.setLevel(level);  
    curso.setTeacher(teacher);  
  
    CursoModelo cursoModelo = new CursoModelo();  
    int value = cursoModelo.createCurso(curso);  
  
    if(value==0) {  
        System.out.println("Ahora se ejecutará listCurso");  
        listCurso(request, response);  
    } else {  
        request.setAttribute("mensaje", "Ocurrió un error");  
        request.getRequestDispatcher("curso.jsp").forward(request, response);  
    }  
}
```

CDNs

JQUERY

Sus CDNs van justo debajo de la etiqueta de cierre del body

Debajo de los CDNs va la validación

```
<script  
type="text/javascript"src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1  
/jquery.min.js"></script>
```

```
<script src="https://cdnjs.cloudflare.com/ajax/libs/jquery-  
validate/1.19.1/jquery.validate.min.js"></script>
```

```
<script type="text/javascript"> //VALIDACI00000000000N </script>
```

J VALIDATE

```
30 //Mediante jquery y la librería validate se valida por medio de mensajes los campos requeridos
31 $(function() {
32     $("form[name='registration']").validate({
33         rules : {
34             email : {
35                 required : true,
36                 email : true
37             },
38             password : {
39                 required : true,
40                 minlength : 8
41             }
42         },
43         messages : {
44             password : {
45                 required : "Ingresa su contraseña",
46                 minlength : "Tu contraseña debe tener al menos 5 caracteres"
47             },
48             email : "Ingresa su correo"
49         },
50         submitHandler : function(form) {
51             form.submit();
52         }
53     });
54 });
55 </script>
56 </html>
```

Y para el mensaje en pantalla...

```
94     },
95     submitHandler:function(form){
96         form.submit();
97     }
98 }); //Aquí acaba la validación
99
100
101 $("form[name='registroNuevo']").submit(function(event){
102     event.preventDefault();
103     if($(this).valid() {
104         alert("Registrado con éxito.");
105         this.submit();
106     } else {
107         alert("Error al registrar.");
108     }
109 });
110
111 }); //Aquí acaba la función
112
113 </script>
114
115 </html>
```

Validación extra al querer REGISTRAR un CURSO

```
<body>
  <div class="container">
    <form action="ServletObjeto" method="post" name="creacion">
      <input type="hidden" value="registrar" name="type">
      <div class="text-center">
        <h3>Registrar Curso</h3>
      </div>
      <div class="mb-3">
        <label>Codigo</label> <input name="txtCodigo" type="text"
          class="form-control">
      </div>
      <div class="mb-3">
        <label>Nombre</label> <input name="txtNombre" type="text"
          class="form-control">
      </div>
      <div class="mb-3">
        <label>Nivel</label> <input name="txtNivel" type="text"
          class="form-control">
      </div>
      <div class="mb-3">
        <label>Profesor</label> <input name="txtProfesor" type="text"
          class="form-control">
      </div>
      <button type="submit" class="btn btn-primary">Enviar Datos</button>
      <button type="button" class="btn btn-primary"
        onclick="location.href='ServletObjeto?type=listar'">Regresar</button>
    </form>
  </div>
</body>
```

```
$(function() {
  $("form[name='creacion']").validate({
    rules: {
      txtCodigo: {
        required: true,
        minlength: 8
      },
      txtNombre: {
        required: true
      },
      txtNivel: {
        required: true
      },
      txtProfesor: {
        required: true
      }
    },
    messages: {
      txtCodigo: {
        required: "Ingrese su codigo",
        minlength: "Su codigo debe tener como minimo 8 caracteres"
      },
      txtNombre: "Ingrese su nombre",
      txtNivel: "ingrese su nivel",
      txtProfesor: "Ingrese un docente"
    },
    submitHandler : function(form) {
      form.submit();
    }
  });
});
</script>
```

BOOTSTRAP

Esto va debajo del TITTLE

```
<link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
1BmE4kWbQ78iYhFLdvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqYL2QvZ6jIW3" crossorigin=
"anonymous">
```

8. .JSP

iniciarSesion.jsp

```
<form name="inicioSesion" action="SubjectServlet" method="post">
  <input type="hidden" name="type" value="List">

  <!-- Email input -->
  <div class="form-outline mb-4">
    <input type="email" name="email" id="email" class="form-control" />
    <label class="form-label" for="form2Example1">Correo</label>
  </div>

  <!-- Password input -->
  <div class="form-outline mb-4">
    <input type="password" name="password" id="password" class="form-control" />
    <label class="form-label" for="form2Example2">Contraseña</label>
  </div>

  <!-- 2 column grid layout for inline styling -->
  <div class="row mb-4">
    <div class="col d-flex justify-content-center">
      <!-- Checkbox -->
      <div class="form-check">
        <input class="form-check-input" type="checkbox" value="" id="form2Example31" checked />
        <label class="form-check-label" for="form2Example31">Recordar mi cuenta </label>
      </div>
    </div>
    <div class="col">
      <!-- Simple link -->
      <a href="#">¿Recuperar contraseña?</a>
    </div>
  </div>

  <!-- Submit button -->
  <button type="submit" class="btn btn-primary btn-block mb-4">Iniciar Sesión</button>
</form>
```

Y para la validación...

```
<!-- jQuery -->
<script type="text/javascript" src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
<!-- jQuery Validation Plugin -->
<script src="https://cdnjs.cloudflare.com/ajax/libs/jquery-validate/1.19.1/jquery.validate.min.js"></script>
<!-- Validación de formulario -->
<script type="text/javascript">
$(function() {
    $("form[name='inicioSesion']").validate({
        rules: {
            email: {
                required: true,
                email: true
            },
            password: {
                required: true,
                minlength: 5
            }
        },
        messages: {
            password: {
                required: "Ingresa su contraseña",
                minlength: "Tu contraseña debe tener al menos 5 caracteres"
            },
            email: "Ingresa su correo"
        },
        submitHandler: function(form) {
            form.submit();
        }
    });
});
</script>
</html>
```

Y para le mensaje en pantalla...

```
94         },
95         submitHandler: function(form){
96             form.submit();
97         }
98     }); //Aquí acaba la validación
99
100     $("form[name='registroNuevo']").submit(function(event){
101         event.preventDefault();
102         if($(this).valid()) {
103             alert("Registrado con éxito.");
104             this.submit();
105         } else {
106             alert("Error al registrar.");
107         }
108     });
109
110     }); //Aquí acaba la función
111
112 </script>
113
114 </html>
115 <
```


subject.jsp

```
1 <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2   pageEncoding="ISO-8859-1"%>
3 <%@ page import="java.util.ArrayList"%>
4 <%@ page import="java.util.List"%>
5 <%@ page import="entidades.Subject"%>
6 <!DOCTYPE html>
7 <html>
8 <head>
9   <meta charset="ISO-8859-1"%>
10  <title>Insert title here</title>
11  <!-- Procurar que en este apartado siempre vayan los css -->
12  <link
13    href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css"
14    rel="stylesheet"
15    integrity="sha384-1BmE4kWBq78iYhFtdvKuhfTAU6auU8tT94WrfhtjDbrCEXSU1oBoqyl2QvZ6jIN3"
16    crossorigin="anonymous">
17 </head>
18 <body>
19   <div class="container">
20     <div class="row">
21       <div class="col-12 text-center">
22         <h3>Mantenimiento de Cursos</h3>
23       </div>
24       <div class="col-12">
25         <button type="button" class="btn btn-primary" onclick="window.location='subjectRegister.jsp'">Registrar</button>
26       </div>
27       <br>
28       <div class="col-12">
29         <table class="table">
30           <thead>
31             <tr>
32               <th>Id</th>
33               <th>Codigo</th>
34               <th>Nombre</th>
35               <th>Nivel</th>
36               <th>Profesor</th>
37               <th>Acciones</th>
38             </tr>
39           </thead>
40           <tbody>
41             <tr>
42               <th>Acciones</th>
43             </tr>
44           </tbody>
45         </table>
46       </div>
47       <div class="col-12">
48         <table>
49           <thead>
50             <tr>
51               <th>Id</th>
52               <th>Codigo</th>
53               <th>Nombre</th>
54               <th>Nivel</th>
55               <th>Profesor</th>
56               <th>Acciones</th>
57             </tr>
58           </thead>
59           <tbody>
60             <tr>
61               <td><%=item.getId()%></td>
62               <td><%=item.getCodigo()%></td>
63               <td><%=item.getNombre()%></td>
64               <td><%=item.getNivel()%></td>
65               <td><%=item.getProfesor()%></td>
66               <td>
67                 <button type="button" onclick="window.location.href='SubjectServlet?type=info&id=<%=item.getId()%>'>Editar</button>
68                 <button type="button" onclick="window.location.href='SubjectServlet?type=delete&id=<%=item.getId()%>'>Eliminar</button>
69               </td>
70             </tr>
71           </tbody>
72         </table>
73       </div>
74     </div>
75   </div>
76 </body>
77 </html>
```

LAS LINEAS DE LAS ETIQUETAS BUTTON (líneas 53-54) DICE:

```
<button type="button" onclick="window.location.href='SubjectServlet?type=info&id=<%=item.getId()%>'>Editar</button>
<button type="button" onclick="window.location.href='SubjectServlet?type=delete&id=<%=item.getId()%>'>Eliminar</button>
```

TAMBIEN PUEDEN SER:

```
<button type="button" class="btn btn-primary" onclick="window.location='LibroServlet?type=info&id=<%= libro.getIdLibro() %>'>Editar</button>
<button type="button" class="btn btn-danger" onclick="window.location='LibroServlet?type=delete&id=<%= libro.getIdLibro() %>'>Eliminar</button>
```

subjectInfo.jsp

```
listaRosas.jsp registroRosas.jsp LibroServlet.java LibroModel.java LibroInterface.java MySqlConnection.java in
1 <%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
2 <%@ page import="entidades.Libro"%>
3 <!DOCTYPE html>
4 <html>
5 <head>
6 <meta charset="UTF-8"%>
7 <title>subjectInfo.jsp</title>
8 </head>
9 <body>
10 <div class="container min-vh-100 d-flex justify-content-center align-items-center">
11 <div class="col-6">
12 <form action="SubjectServlet" method="post">
13 <div class="form-group">
14 <input type="hidden" name="type" value="edit">
15 <input type="hidden" name="idSubject" value="<%=subject.getId()%>">
16 <div class="form-group">
17 <label class="text-secondary">Codigo</label>
18 <input class="form-control" type="text" name="txtCode" value="<%= (subject!=null)? subject.getCode(): "" %>">
19 </div>
20 <div class="form-group">
21 <label class="text-secondary">Nombre</label>
22 <input class="form-control" type="text" name="txtName" value="<%= (subject!=null)? subject.getName(): "" %>">
23 </div>
24 <div class="form-group">
25 <label class="text-secondary">Nivel</label>
26 <input class="form-control" type="text" name="txtLevel" value="<%= (subject!=null)? subject.getLevel(): "" %>">
27 </div>
28 <div class="form-group">
29 <label class="text-secondary">Profesor</label>
30 <input class="form-control" type="text" name="txtTeacher" value="<%= (subject!=null)? subject.getTeacher(): "" %>">
31 </div>
32 <div class="form-group">
33 <input type="submit" class="btn btn-primary" value="Actualizar datos">
34 <input type="button" class="btn btn-primary" value="Regresar" value="Regresar" onclick="location.href='SubjectServlet?type=list'">
35 </div>
36 </div>
37 </div>
38 </body>
39 </html>
```

Esto es importante (lo de aquí abajo)

```
42 <input class="form-control" type="text" name="txtTeacher" value="<%= (subject!=null)? subject.getTeacher(): "" %>">
43 </div>
44 <div class="form-group">
45 <input type="submit" class="btn btn-primary" value="Actualizar datos">
46 <input type="button" class="btn btn-primary" value="Regresar" value="Regresar" onclick="location.href='SubjectServlet?type=list'">
47 </div>
48 </div>
49 </body>
50 </html>
```

BOTONES TAMBIEN PUEDEN SER:

```
<button type="submit" class="btn btn-primary">Actualizar</button>
<button type="button" class="btn btn-primary" onclick="window.location='LibroServlet?type=list'">Volver al listado</button>
```

CADA PARTE DEL FORMULARIO de info TAMBIÉN PUEDE SER:

```
</head>
<body>
  <br>
  <div class="container">
    <h3 class="text-center">Actualizar registro</h3>
    <form action="LibroServlet" method="post">
      <%
        Libro libro = (Libro) request.getAttribute("data");
      %>

      <input type="hidden" name="type" value="edit">

      <div>
        <input type="hidden" name="idLibro" value="<%= libro.getIdLibro() %>">
      </div>
      <div class="mb-3">
        <label>Código</label>
        <input class="form-control" type="text" name="txtCodigo" value="<%= libro.getCodigo() %>">
      </div>
      <div class="mb-3">
        <label>Nombre del libro</label>
        <input class="form-control" type="text" name="txtNombreLibro" value="<%= libro.getNombreLibro() %>">
      </div>
      <div class="mb-3">
        <label>Edición</label>
        <input class="form-control" type="text" name="txtEdicion" value="<%= libro.getEdicion() %>">
      </div>
      <div class="mb-3">
        <label>Tipo</label>
        <input class="form-control" type="text" name="txtTipo" value="<%= libro.getTipo() %>">
      </div>
      <div class="mb-3">
        <label>Precio($/.)</label>
        <input class="form-control" type="text" name="txtPrecio" value="<%= libro.getPrecio() %>">
      </div>
      <div class="mb-3">
        <label>Stock</label>
        <input class="form-control" type="text" name="txtStock" value="<%= libro.getStock() %>">
      </div>
      <div class="mb-3">
        <label>Fecha de compra</label>
        <input class="form-control" type="date" name="txtFechaCompra" value="<%= libro.getFechaCompra() %>">
      </div>

      <button type="submit" class="btn btn-primary">Actualizar</button>
      <button type="button" class="btn btn-primary" onclick="window.location='LibroServlet?type=list'">Volver al listado</button>
    </form>
```

subjectRegister.jsp

```
MySqlConexion.java Subject.java SubjectInterface.java SubjectModel.java SubjectServlet.java iniciarSesion2.jsp subject.jsp subjectInfo.jsp subjectRegister.jsp X
6 <meta charset="ISO-8859-1">
7 <title>Insert title here</title>
8 <link
9 href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css"
10 rel="stylesheet"
11 integrity="sha384-1BmE4kWBq78iYhFLdvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIN3"
12 crossorigin="anonymous">
13 </head>
14 <body>
15 <div class="container min-vh-100 d-flex justify-content-center align-items-center">
16 <div class="col-6">
17 <h3>Registrar Curso</h3>
18 <form action="SubjectServlet" method="post">
19 <input type="hidden" name="type" value="register">
20 <div class="form-group">
21 <label>Código</label> <input class="form-control" type="text"
22 name="txtCode">
23 </div>
24 <div class="form-group">
25 <label>Nombre</label> <input class="form-control" type="text"
26 name="txtName">
27 </div>
28 <div class="form-group">
29 <label>Nivel</label> <input class="form-control" type="text"
30 name="txtLevel">
31 </div>
32 <div class="form-group">
33 <label>Profesor</label> <input class="form-control" type="text"
34 name="txtTeacher">
35 </div>
36 <br>
37 <button type="submit" class="btn btn-primary">Enviar datos</button>
38 <button type="button" class="btn btn-primary" onclick="location.href='SubjectServlet?type=list'">Regresar</button>
39 </form>
40 </div>
41 </div>
42
43 </body>
44 </html>
```

OTRA MANERA DE HACER ESOS ULTIMOS 2 BOTONES:

```
<input type="date" class="form-control" name="txtFechaCompra">
</div>
<button type="submit" class="btn btn-primary">Registrar</button>
<button type="button" class="btn btn-primary" onclick="window.location='LibroServlet?type=list'">Volver al Historial</button>
</form>
```

OTRA MANERA DE HACER EL REGISTER:

```
registroNuevos.jsp X
10 <body>
11 <br>
12 <div class="container">
13 <h3>Nuevo registro</h3>
14 <form action="LibroServlet" method="post" name="registroNuevo">
15 <input type="hidden" name="type" value="register">
16 <div class="mb-3">
17 <label>Código</label>
18 <input type="text" class="form-control" name="txtCodigo">
19 </div>
20 <div class="mb-3">
21 <label>Nombre del Libro</label>
22 <input type="text" class="form-control" name="txtNombreLibro">
23 </div>
24 <div class="mb-3">
25 <label>Edición</label>
26 <input type="text" class="form-control" name="txtEdicion">
27 </div>
28 <div class="mb-3">
29 <label>Tipo</label>
30 <input type="text" class="form-control" name="txtTipo">
31 </div>
32 <div class="mb-3">
33 <label>Precio(S/.)</label>
34 <input type="text" class="form-control" name="txtPrecio" placeholder="Ejemplo: 56.80">
35 </div>
36 <div class="mb-3">
37 <label>Stock</label>
38 <input type="text" class="form-control" name="txtStock">
39 </div>
40 <div class="mb-3">
41 <label>Fecha de compra</label>
42 <input type="date" class="form-control" name="txtFechaCompra">
43 </div>
44 <button type="submit" class="btn btn-primary">Registrar</button>
45 <button type="button" class="btn btn-primary" onclick="window.location='LibroServlet?type=list'">Volver al Historial</button>
46 </form>
47
48 </div>
49
50 </body>
```

J VALIDATE aquí:

```
registroRosas.jsp X
50 </body>
51
52 <script type="text/javascript" src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
53 <script src="https://cdnjs.cloudflare.com/ajax/libs/jquery-validate/1.19.1/jquery.validate.min.js"></script>
54 <script type="text/javascript"> //
55
56     $(function(){
57         $("form[name='registroNuevo']").validate({
58             rules:{
59                 txtCodigo:{
60                     required:true,
61                     minlength:5
62                 },
63                 txtNombreLibro:{
64                     required:true
65                 },
66
67                 txtEdicion:{
68                     required:true
69                 },
70                 txtTipo:{
71                     required:true
72                 },
73                 txtPrecio:{
74                     required:true
75                 },
76                 txtStock:{
77                     required:true
78                 },
79                 txtFechaCompra:{
80                     required:true
81                 }
82             },
83             messages:{
84                 txtCodigo:{
85                     required:"Ingrese código.",
86                     minlength:"Código debe tener al menos 5 números."
87                 },
88                 txtNombreLibro:"Ingrese nombre del libro.",
89                 txtEdicion:"Ingrese número de edición.",
90                 txtTipo:"Ingrese tipo de libro.",
91                 txtPrecio:"Ingrese precio",
92                 txtStock:"Ingrese stock.",
93                 txtFechaCompra:"Seleccione fecha de compra."
94             },
95             submitHandler:function(form){
96                 form.submit();
97             }
98         }); //Aquí acaba la validación
99
100
```

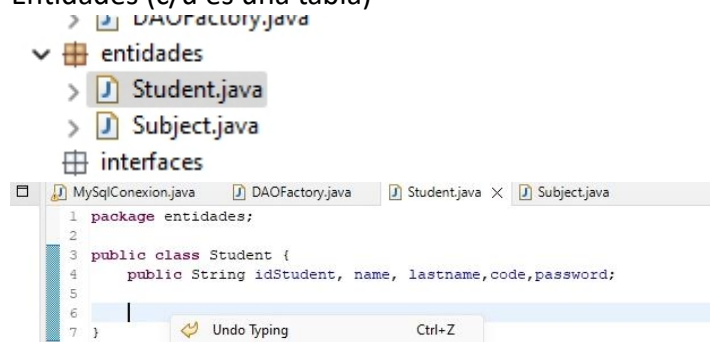
Con mensaje de confirmación en pantalla:

```
79         txtFechaCompra:{
80             required:true
81         }
82     },
83     messages:{
84         txtCodigo:{
85             required:"Ingrese código.",
86             minlength:"Código debe tener al menos 5 números."
87         },
88         txtNombreLibro:"Ingrese nombre del libro.",
89         txtEdicion:"Ingrese número de edición.",
90         txtTipo:"Ingrese tipo de libro.",
91         txtPrecio:"Ingrese precio",
92         txtStock:"Ingrese stock.",
93         txtFechaCompra:"Seleccione fecha de compra."
94     },
95     submitHandler:function(form){
96         form.submit();
97     }
98
99 }); //Aquí acaba la validación
100
101 $("form[name='registroNuevo']").submit(function(event){
102     event.preventDefault();
103     if($(this).valid()) {
104         alert("Registrado con éxito.");
105         this.submit();
106     } else {
107         alert("Error al registrar.");
108     }
109 });
110
111 }); //Aquí acaba la función
112
113 </script>
114
115 </html>
```

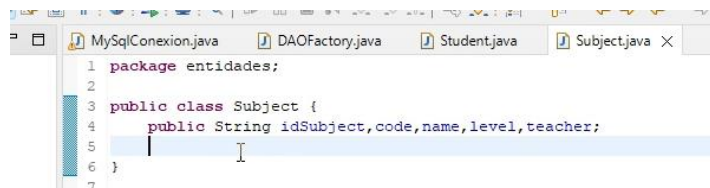
L2: APUNTES - DAO & SESIONES

& CRUD

1. Creación proyecto / Estructuración proyecto
2. Workbench *PLANTILLA COPY/PASTE*
3. MySqlConnection *PLANTILLA COPY/PASTE*
4. Entidades (c/u es una tabla)



+ getters&setters

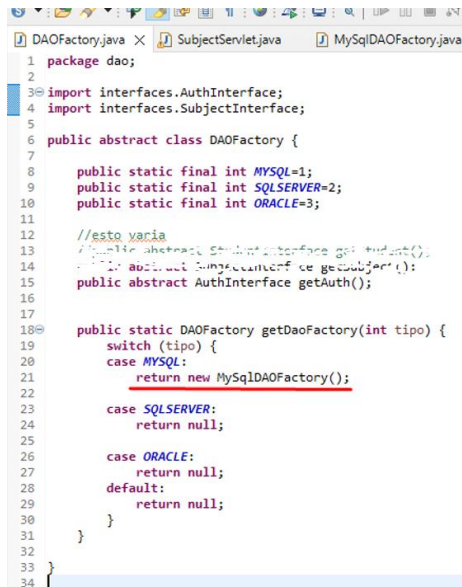


+ getters&setters

DAO & SESIÓN

5. DAO Factory (abstracto) **PLANTILLA COPY/PASTE**

De aquí salen ABSTRACT INTERFACE y te manda al xxxDAOFactory del que corresponda:

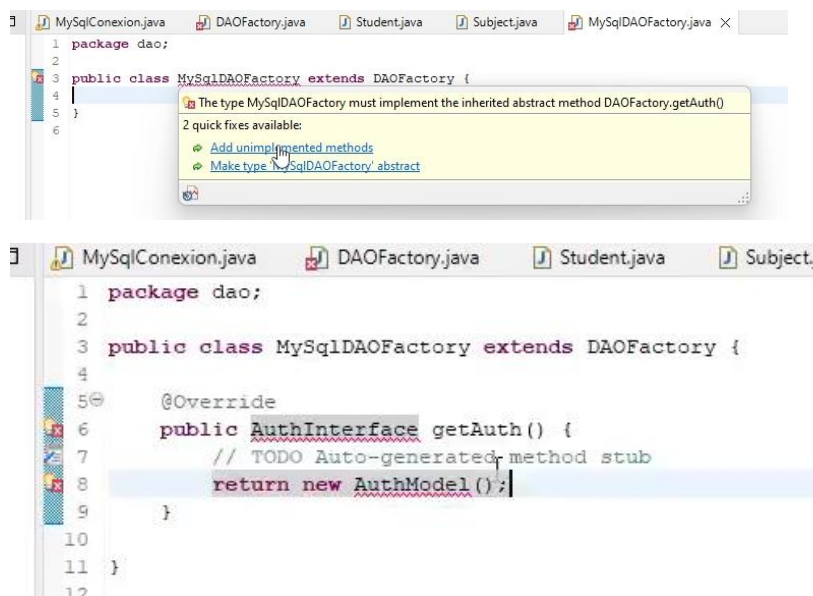


```
1 package dao;
2
3 import interfaces.AuthInterface;
4 import interfaces.SubjectInterface;
5
6 public abstract class DAOFactory {
7
8     public static final int MYSQL=1;
9     public static final int SQLSERVER=2;
10    public static final int ORACLE=3;
11
12    //esto varia
13    //public abstract StudentInterface getStudent();
14    //public abstract ConnectionInterface getConnection();
15    public abstract AuthInterface getAuth();
16
17
18    public static DAOFactory getDaoFactory(int tipo) {
19        switch (tipo) {
20            case MYSQL:
21                return new MySqlDAOFactory();
22            case SQLSERVER:
23                return null;
24            case ORACLE:
25                return null;
26            default:
27                return null;
28        }
29    }
30
31 }
32
33
34 }
```

Pasamos mouse y creamos clase xxxDAOFactory que corresponda.

6. MySqlDAOFactory (creado aplicando métodos no implementados) **AQUÍ VIENE LO MEDIO COMPLICADO**

Este tiene métodos TIPO INTERFAZ que te mandan al MODELO.



```
1 package dao;
2
3 public class MySqlDAOFactory extends DAOFactory {
4
5 }
6
```

The type MySqlDAOFactory must implement the inherited abstract method DAOFactory.getAuth()

2 quick fixes available:

- Add unimplemented methods
- Make type 'MySqlDAOFactory' abstract

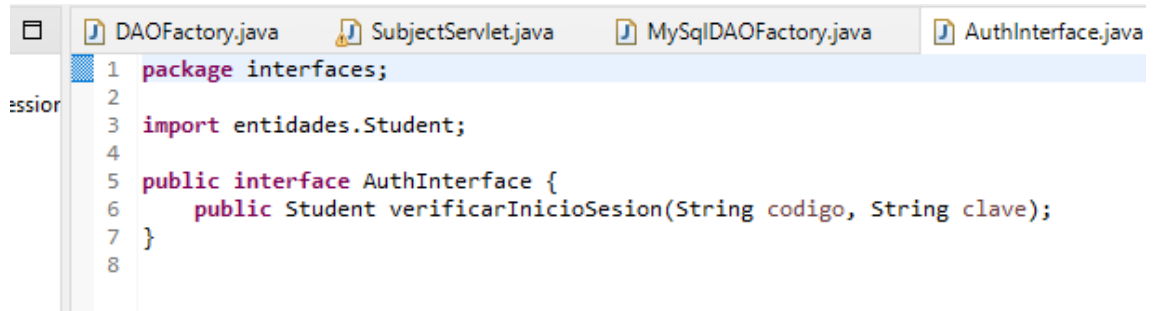
```
1 package dao;
2
3 public class MySqlDAOFactory extends DAOFactory {
4
5     @Override
6     public AuthInterface getAuth() {
7         // TODO Auto-generated method stub
8         return new AuthModel();
9     }
10
11 }
12
```

PRIMERO NOS ENCARGAMOS DE LA **INTERFAZ** Y LUEGO DEL **MODELO**

Pasamos mouse encima a AuthInterface y creamos su INTERFAZ

7. Interfaz

RECUERDA, INTERFAZ TIENE DECLARADO METODOS QUE USARA EL MODELO (implements interface).

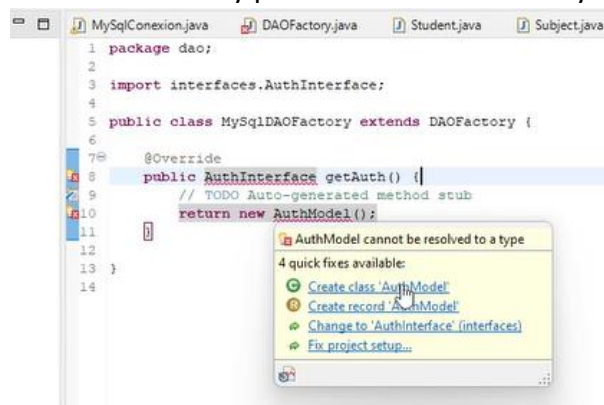


```
1 package interfaces;
2
3 import entidades.Student;
4
5 public interface AuthInterface {
6     public Student verificarInicioSesion(String codigo, String clave);
7 }
8
```

8. MODELO

PLANTILLA COPY/PASTE

En xxxDAOFactory pasamos Mouse encima y creamos modelo



```
1 package dao;
2
3 import interfaces.AuthInterface;
4
5 public class MySQLDAOFactory extends DAOFactory {
6
7     @Override
8     public AuthInterface getAuth() {
9         // TODO Auto-generated method stub
10        return new AuthModel();
11    }
12
13 }
14
```

parecido al modelo de toda la vida.
pero no es igual.

FUNCIÓN DE MODELO INICIAR SESION:

Identifica el objeto estudiante que esta logeandose y lo retorna con 3 para atributos para poder mostrarlos en el jsp

```
DAOFactory.java SubjectServlet.java MySQLDAOFactory.java AuthInterface.java
1 package modelo;
2
3 import java.sql.Connection;
10
11 public class AuthModel implements AuthInterface {
12
13     @Override
14     public Student verificarInicioSesion(String codigo, String clave) {
15         // TODO Auto-generated method stub
16
17         Student student= null;
18         PreparedStatement psmt = null;
19         Connection cn = null;
20         ResultSet rs = null;
21
22         try {
23             cn = MySQLConexion.getConnection();
24             String mysql = "SELECT * FROM STUDENT WHERE CODE=? AND PASSWORD=?";
25             psmt = cn.prepareStatement(mysql);
26             psmt.setString(1, codigo);
27             psmt.setString(2, clave);
28             rs = psmt.executeQuery();
29             if(rs.next()) {
30                 student = new Student();
31                 student.setName(rs.getString("name"));
32                 student.setLastname(rs.getString("lastname"));
33                 student.setCode(rs.getString("code"));
34             }
35         } catch (Exception e) {
36             // TODO: handle exception
37             e.printStackTrace();
38             e.printStackTrace();
39
40         } finally {
41             try {
42                 if (rs != null) rs.close();
43                 if (psmt != null) rs.close();
44                 if (cn != null) rs.close();
45             } catch (Exception e) {
46                 // TODO: handle exception
47                 e.printStackTrace();
48             }
49         }
50         return student;
51     }
52 }
53 }
```

9. SERVLET

(AUTH_SERVLET)

PLANTILLA COPY/PASTE

RECUERDA QUE SERVLET recupera o envía información a los JSP.

RECUERDA QUE SERVLET utiliza a los MODEL.

ahora antes de utilizar MODEL, tenemos que pasar por el patrón DAO.

FUNCIÓN DE SERVLET INICIAR SESION:

Agarrar petición TYPE LOGIN del JSP iniciar sesión, agarra parámetros

Pasa por el DAO, luego Interfaz y luego Modelo

Obteniendo al estudiante q se logea.

CREA LA SESIÓN E INICIA CON HTTPS SESSION y true

Obtiene 3 datos del estudiante y los almacena en el servlet (setAttribute) para dsp usarlos...

Ahora viene Subject

DAO & SUBJECT para mostrar listado al iniciar Sesion

Ahora agregamos todo desde el DAO pero de parte del Subject.

1. Agregamos ese Abstract class en DAO Factory

PLANTILLA COPY/PASTE

```
1 package dao;
2
3 import interfaces.AuthInterface;
4
5 public abstract class DAOFactory {
6
7     public static final int MYSQL=1;
8     public static final int SQLSERVER=2;
9     public static final int ORACLE=3;
10
11
12     public abstract AuthInterface getAuth();
13     public abstract SubjectInterface getSubject();
14
15
16     public static DAOFactory getDaoFactory(int tipo) {
17         switch (tipo) {
18             case MYSQL:
19                 return new MySqlDAOFactory();
20
21             case SQLSERVER:
22                 return null;
23
24             case ORACLE:
25                 return null;
26             default:
27                 return null;
28         }
29     }
30 }
31 }
32 }
```

Para luego en la clase
xxxDaoFactory CREAR EL
METODO FALTANTE

2. Aumentamos el método para Subject en xxxDAOFactory

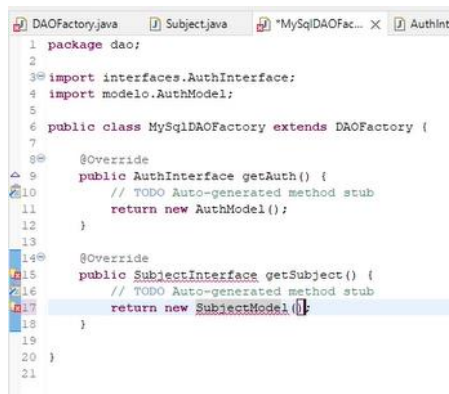
```
1 package dao;
2
3 import interfaces.AuthInterface;
4 import modelo.AuthModel;
5
6 public class MySqlDAOFactory extends DAOFactory {
7
8     @Override
9     public AuthInterface getAuth() {
10         // TODO
11         return null;
12     }
13
14 }
15 }
```

The type MySqlDAOFactory must implement the inherited abstract method DAOFactory.getSubject()

2 quick fixes available:

- Add unimplemented methods
- Make type 'MySqlDAOFactory' abstract

Press F2 for focus



```

1 package dao;
2
3 import interfaces.AuthInterface;
4 import modelo.AuthModel;
5
6 public class MySQLDAOFactory extends DAOFactory {
7
8     @Override
9     public AuthInterface getAuth() {
10         // TODO Auto-generated method stub
11         return new AuthModel();
12     }
13
14     @Override
15     public SubjectInterface getSubject() {
16         // TODO Auto-generated method stub
17         return new SubjectModel();
18     }
19 }
20
21

```

3.**INTERFAZ**: Creamos **interfaz** subject y le metemos métodos CRUD



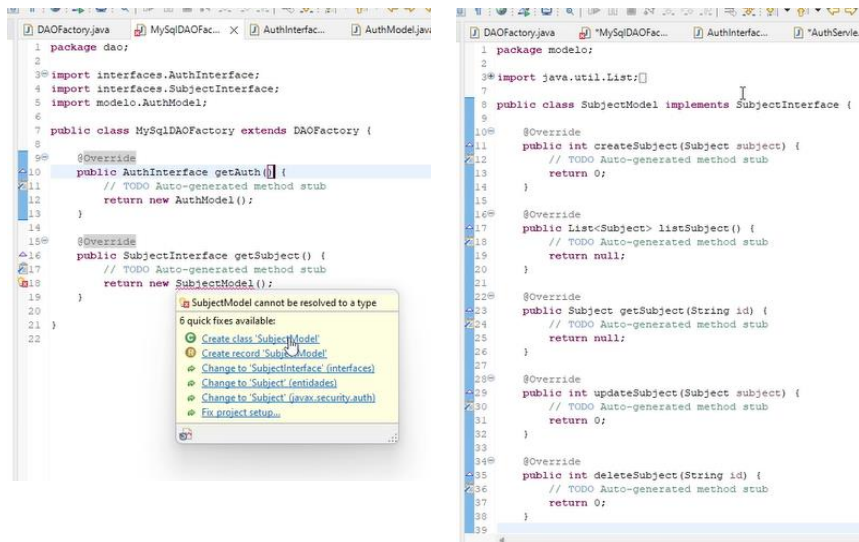
```

1 package interfaces;
2
3 import java.util.List;
4
5 import entidades.Subject;
6
7 public interface SubjectInterface {
8
9     public int createSubject(Subject subject);
10    public List<Subject> listSubject();
11    public Subject getSubject(String id);
12    public int updateSubject(Subject subject);
13    public int deleteSubject(String id);
14 }
15

```

Terminamos de importar lo que falte importar en otras clases.

4.**MODELO**: Desde **xxxDAOFactory** creamos **MODELO** subject – implements **INTERFAZ**



IGUALITO QUE EL MODELO PARA CRUD (L1)

Para listar cursos y verlos después de logearse, solo se necesita METODO LISTAR

VOLVEMOS A SERVLET

SEVLET:

(AuthServlet)

PLANTILLA COPY/PASTE

RECUERDA QUE SERVLET recupera o envía información a los JSP.

RECUERDA QUE SERVLET utiliza a los MODEL.

ahora antes de utilizar MODEL, tenemos que pasar por el patrón DAO.

Completamos la parte para **LISTAR** de servletAuth

FUNCIÓN DE SERVLET INICIAR SESION (PARTE DE LISTADO SUBJECT):

Pasamos por el DAO para utilizar **MODELO** listar y así obtener todos los subject en un listado.

Lo pasamos al servlet con (setAttribute).

Pasamos TODO tanto datos del estudiante haciendo LOGIN & datos del listado con (getRequestDispatcher)

HACEMOS ELSE IF para “logout” en donde matamos la sesión y todo

HASTA AHÍ TEEMOS TODO EL **BACK NECESARIO PARA LISTAR**

Ahora viene FRONT

JSP

login

```
login.jsp x listSubject.jsp
1 <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2   pageEncoding="ISO-8859-1"%>
3 <!DOCTYPE html>
4 <html>
5 <head>
6 <meta charset="ISO-8859-1">
7 <title>Insert title here</title>
8 <link
9   href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css"
10  rel="stylesheet"
11  integrity="sha384-
12  1BmE4kWBq78iYhFtdvKuhfTAU6auU8tT94WtHftjDbrCEXSU1oBoqyl2QvZ6jIW3"
13  crossorigin="anonymous">
14 </head>
15 <body>
16 <div class="container min-vh-100 d-flex justify-content-center align-items-center">
17 <form action="AuthServlet">
18   <input type="hidden" name="type" value="Login">
19   <div class="mb-3">
20     <label for="exampleInputEmail1" class="form-label">Codigo de Alumno</label>
21     <input type="text" name="txtCode" class="form-control" id="exampleInputEmail1" aria-describedby="emailHelp">
22     <div id="emailHelp" class="form-text">We'll never share your email with anyone else.</div>
23   </div>
24   <div class="mb-3">
25     <label for="exampleInputPassword1" class="form-label">Contraseña</label>
26     <input type="password" name="txtPass" class="form-control" id="exampleInputPassword1">
27   </div>
28   <div class="mb-3 form-check">
29     <input type="checkbox" class="form-check-input" id="exampleCheck1">
30     <label class="form-check-label" for="exampleCheck1">Check me out</label>
31   </div>
32   <button type="submit" class="btn btn-primary">Submit</button>
33 </form>
34 </div>
35 </body>
36 </html>
```

Acercamiento a lo importante

```
<div class="container min-vh-100 d-flex justify-content-center align-items-center">
<form action="AuthServlet">
  <input type="hidden" name="type" value="Login">
  <div class="mb-3">
    <label for="exampleInputEmail1" class="form-label">Codigo de Alumno</label>
    <input type="text" name="txtCode" class="form-control" id="exampleInputEmail1" aria-describedby="emailHelp">
    <div id="emailHelp" class="form-text">We'll never share your email with anyone else.</div>
  </div>
  <div class="mb-3">
    <label for="exampleInputPassword1" class="form-label">Contraseña</label>
    <input type="password" name="txtPass" class="form-control" id="exampleInputPassword1">
  </div>
  <div class="mb-3 form-check">
    <input type="checkbox" class="form-check-input" id="exampleCheck1">
    <label class="form-check-label" for="exampleCheck1">Check me out</label>
  </div>
  <button type="submit" class="btn btn-primary">Submit</button>
</form>
```

Saludo&listado

```
login.jsp listSubject.jsp X
1 <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2   pageEncoding="ISO-8859-1"%>
3 <%@ page import="java.util.ArrayList"%>
4 <%@ page import="java.util.List"%>
5 <%@ page import="entidades.Subject"%>
6 <!DOCTYPE html>
7 <html>
8 <head>
9   <meta charset="ISO-8859-1">
10  <title>Insert title here</title>
11 </head>
12 <link
13   href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css"
14   rel="stylesheet"
15   integrity="sha384-
16 1BmE4kWBq78iYhFLdvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3"
17   crossorigin="anonymous">
18 <body>
19 <nav class="navbar navbar-expand-lg navbar-dark bg-primary">
```

DENTRO DE BODY

Saludo h3 & normal h3 de titulo

```
<h3>Usuario logueado: ${sessionScope.nombre} ${sessionScope.apellido} - ${sessionScope.codigo}</h3>
<div class="col-12 text-center">
  <h3>Mantenimiento de Cursos</h3>
</div>
```

botón de LogOut

```
<form action="AuthServlet" class="d-flex">
  <input type="hidden" name="type" value="Logout">
  <button class="btn btn-outline-light" type="submit">Cerrar Sesión</button>
</form>
...
```

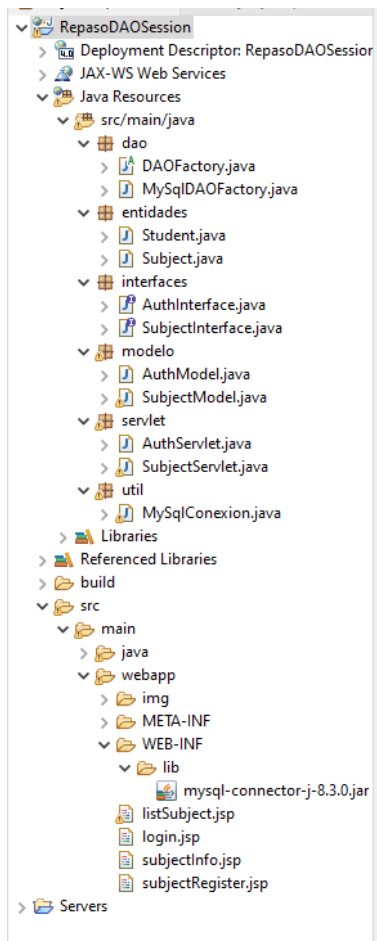
Listado

```

<br>
<div class="col-12">
    <table class="table">
        <thead>
            <tr>
                <th>Id</th>
                <th>Código</th>
                <th>Nombre</th>
                <th>Nivel</th>
                <th>Profesor</th>
                <th>Acciones</th>
            </tr>
        </thead>
        <tbody>
            <%
                List<Subject> listSubject = (List<Subject>) request.getAttribute("data");
                if (listSubject != null) {
                    for (Subject item : listSubject) {
            %>
            <tr>
                <td><%=item.getIdSubject()%></td>
                <td><%=item.getCode()%></td>
                <td><%=item.getName()%></td>
                <td><%=item.getLevel()%></td>
                <td><%=item.getTeacher()%></td>
                <td><a
                    href="SubjectServlet?type=info&id=<%=item.getIdSubject()%>"> 
                </a> <a href="SubjectServlet?type=delete&id=<%=item.getIdSubject()%>">
                    
                </a></td>
            </tr>
            <%
                }
            %>
        </tbody>
    </table>
</div>
</div>

```


Explorador



RESUMEN

MODEL

authModel:

para trackear que estudiante se esta logeando.

Devuelve un objeto

similar pero no igual al model de subject(L1)

HAY APUNTES NOTEPAD

subjectModel:

igual al model de subject(L1)

HAY SCREENSHOTS ARRIBA

SERVLET

authServlet: (full .JSP de saludo y listado)

Agarrar petición TYPE LOGIN del JSP iniciar sesión, agarra parámetros

Pasa por el DAO, luego Interfaz y luego Modelo **para obtener al estudiante que se logea**

CREA LA SESIÓN E INICIA CON HTTPS SESSION y true

Obtiene 3 datos del estudiante y los almacena en el servlet (setAttribute) para dsps usarlos...
subjectServlet

Pasamos por el DAO para utilizar **MODELO listar** y asi obtener todos los subject en un listado.

Lo pasamos al servlet con (setAttribute).

Pasamos TODO tanto datos del estudiante haciendo LOGIN & datos del listado con
(getRequestDispatcher)

HACEMOS ELSE IF para “logout” en donde matamos la sesión y todo

HAY APUNTES NOTEPAD

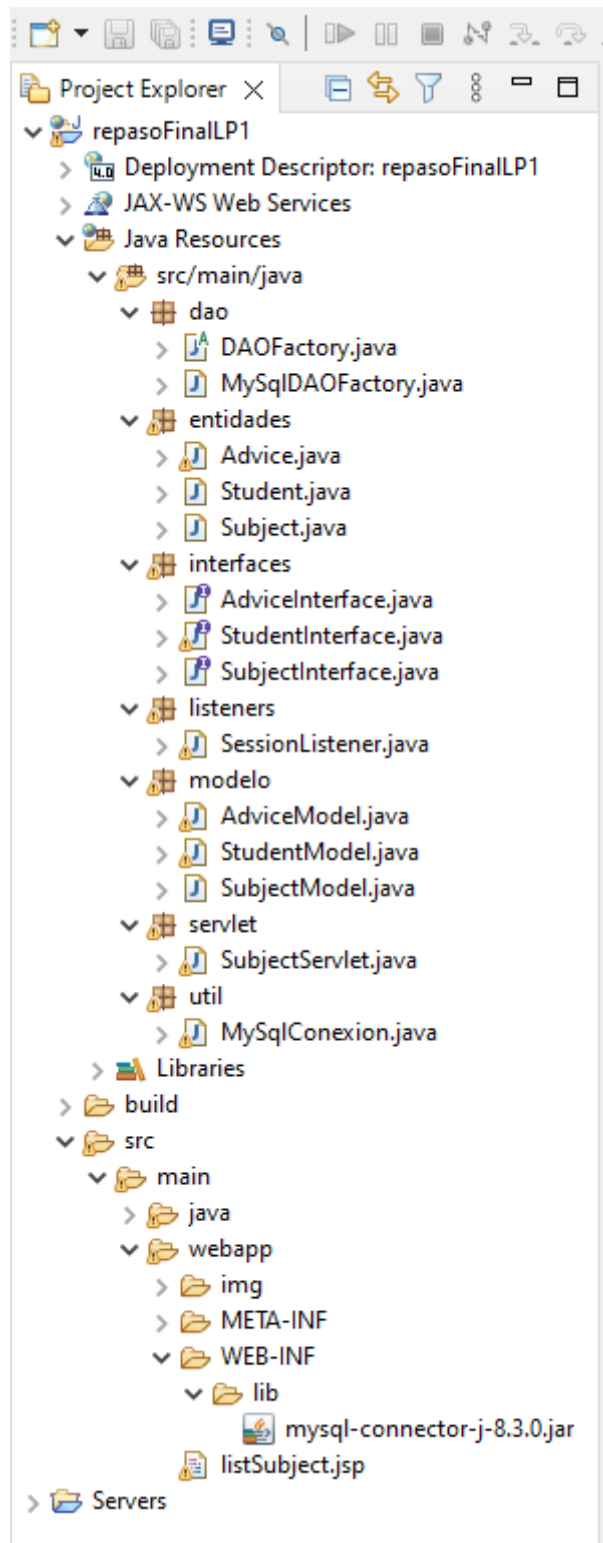
subjectServlet:

Es lo *mismo que el servlet en la L1, solo que* se le aumentan las primeras 2 lineas donde **PASA POR DAO PRIMERO**

```
private void listSubject(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    // TODO Auto-generated method stub

    DAOFactory daoFactory=DAOFactory.getDaoFactory(DAOFactory.MYSQL);
    SubjectInterface subjectInterface=daoFactory.getSubject (0);
    List<Subject> data=subjectInterface.listSubject ();
    request.setAttribute("data", data);
    request.getRequestDispatcher("listSubject.jsp").forward(request, response);
}
```

APUNTES EF



1. WORKBENCH MYSQL ----- **BLOC DE NOTAS**
2. CONEXIÓN A BASE DE DATOS (UTIL) MySqlConnection-**BLOC DE NOTAS** pero **los apuntes pasados**
3. Entidades

```

6 public class Advice {
7     public String idAdvice,codeStudent,codeSubject;
8 }

```

```

1 package entidades;
2
3 public class Student {
4     public String idStudent,name,lastName,code;|
5 }

```

```

2
3 public class Subject {
4     public String id, code, name, level, teacher;
5     public int available;
6 }

```

+ gettes & setters

4. DAO Factory abstracto ----- **BLOC DE NOTAS**
5. MySqlDAOFactory

Autogenerado desde el DAOFactory abstracto

```

DAOFactory.java x  MySqlDAOFactory.java x
1 package dao;
2
3 import interfaces.AdviceInterface;
4
5
6
7
8
9
10 public class MySqlDAOFactory extends DAOFactory{
11
12     @Override
13     public SubjectInterface getSubject() {
14         // TODO Auto-generated method stub
15         return new SubjectModel();
16     }
17
18     @Override
19     public StudentInterface getStudent() {
20         // TODO Auto-generated method stub
21         return new StudentModel();
22     }
23
24     @Override
25     public AdviceInterface getAdvice() {
26         // TODO Auto-generated method stub
27         return new AdviceModel();
28     }
29
30 }
31

```

Aumentas los retornos MODEL DE CADA ENTIDAD en cada método como se muestra en la imagen

6. Interfaces

Autogenerados desde **MySqlDAOFactory**

```
AdviceInterface.java × StudentInterface.java SubjectInterface.java
1 package interfaces;
2
3 import entidades.Advice;
4
5 public interface AdviceInterface {
6     public int createAdvice(Advice advice);
7 }
8
```

```
StudentInterface.java × SubjectInterface.java
1 package interfaces;
2
3 import java.util.List;
4
5 public interface StudentInterface {
6     public List<Student> listStudents();
7 }
8
```

```
SubjectInterface.java ×
1 package interfaces;
2
3 import java.util.List;
4
5 public interface SubjectInterface {
6     public List<Subject> listSubjects();
7 }
8
```

7. Modelos ----- Autogenerados desde **MySqlDAOFactory**

Completar con **BLOC DE NOTAS**

8. Listeners ----- SessionListener **BLOC DE NOTAS**

9. Servlet ----- SubjectServlet **BLOC DE NOTAS**

10. JSP ----- listSubject **BLOC DE NOTAS**