

1. Crear un proyecto
2. Crear un package llamado controller
3. Crear una clase llamada ApiServicelet

Agregar el siguiente método:

```
@WebServlet("/api/data")
public class ApiServicelet extends HttpServlet{
    private static final long serialVersionUID = 1L;

    protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws IOException{
        //Configuración del JSON que responderá la petición
        response.setContentType("application/json");
        response.setCharacterEncoding("UTF-8");

        List<String> list = new ArrayList<>();

        CustomResponse cResponse = new CustomResponse(200, "Ok", list);

        String jsonResponse = new Gson().toJson(cResponse);
        response.getWriter().write(jsonResponse);
    }
}
```

4. La clase CustomResponse únicamente manejará los siguientes datos con sus getters and setters.

```
public class CustomResponse {
    private Integer httpCode;
    private String mensaje;
    private Object data;

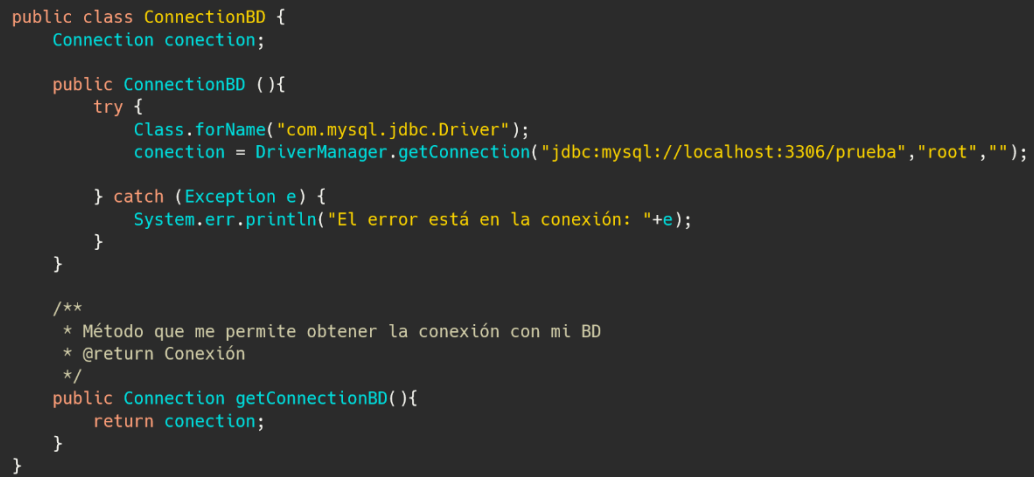
    public CustomResponse(Integer httpCode, String mensaje, Object data) {
        this.httpCode = httpCode;
        this.mensaje = mensaje;
        this.data = data;
    }
}
```

5. Crear un Standard Deployment Descriptor (web.xml) y agregar lo siguiente:

```
<servlet>
    <servlet-name>PostHandlerServlet</servlet-name>
    <servlet-class>com.example.PostHandlerServlet</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>PostHandlerServlet</servlet-name>
    <url-pattern>/postHandler</url-pattern>
</servlet-mapping> <!-- Página de bienvenida -->
<welcome-file-list>
    <welcome-file>index.jsp</welcome-file>
</welcome-file-list>
```

Implementa operaciones crud

Clase connection en el package configuration



```
public class ConnectionBD {
    Connection connection;

    public ConnectionBD () {
        try {
            Class.forName("com.mysql.jdbc.Driver");
            connection = DriverManager.getConnection("jdbc:mysql://localhost:3306/prueba","root","");
        } catch (Exception e) {
            System.err.println("El error está en la conexión: "+e);
        }
    }

    /**
     * Método que me permite obtener la conexión con mi BD
     * @return Conexión
     */
    public Connection getConnectionBD(){
        return connection;
    }
}
```

Crear un servlet Usuario dentro de controller

```

@WebServlet(
    name = "USUARIOS",
    description = "Representa las operaciones de un CRUD para los Usuarios",
    urlPatterns = {"/listar_usuarios", "/registrar_usuario",
        "/borrar_usuario", "/actualizar_usuario"})
public class Usuarios {
    ConnectionBD conexion = new ConnectionBD();
    Connection conn;
    PreparedStatement ps;
    ResultSet rs;

    protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException, SQLException {
        // Obtener los parámetros del formulario
        String nombre = request.getParameter("txt_nombre");
        String correo = request.getParameter("txt_correo");
        String matricula = request.getParameter("txt_matricula");
        String apellidos = request.getParameter("txt_apellidos");
        String celular = request.getParameter("txt_celular");
        String fecha = request.getParameter("txt_fecha_nac");
        String hora = request.getParameter("txt_hora");

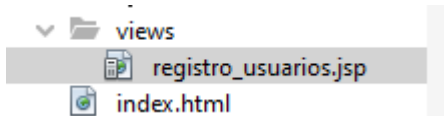
        Date fechaFinal = Date.valueOf(fecha);
        Time horaFinal = Time.valueOf(hora+":00");

        try {
            // Crear la consulta SQL para insertar el usuario
            String sql = "INSERT INTO usuario (apellidos, celular, correo, "+
                "fecha_nac, matricula, nombre, hora) VALUES (?, ?, ?, ?, ?, ?, ?)";
            conn = conexion.getConnectionBD();
            ps = conn.prepareStatement(sql);
            ps.setString(1, apellidos);
            ps.setString(2, celular);
            ps.setString(3, correo);
            ps.setDate(4, fechaFinal);
            ps.setString(5, matricula);
            ps.setString(6, nombre);
            ps.setTime(7, horaFinal);

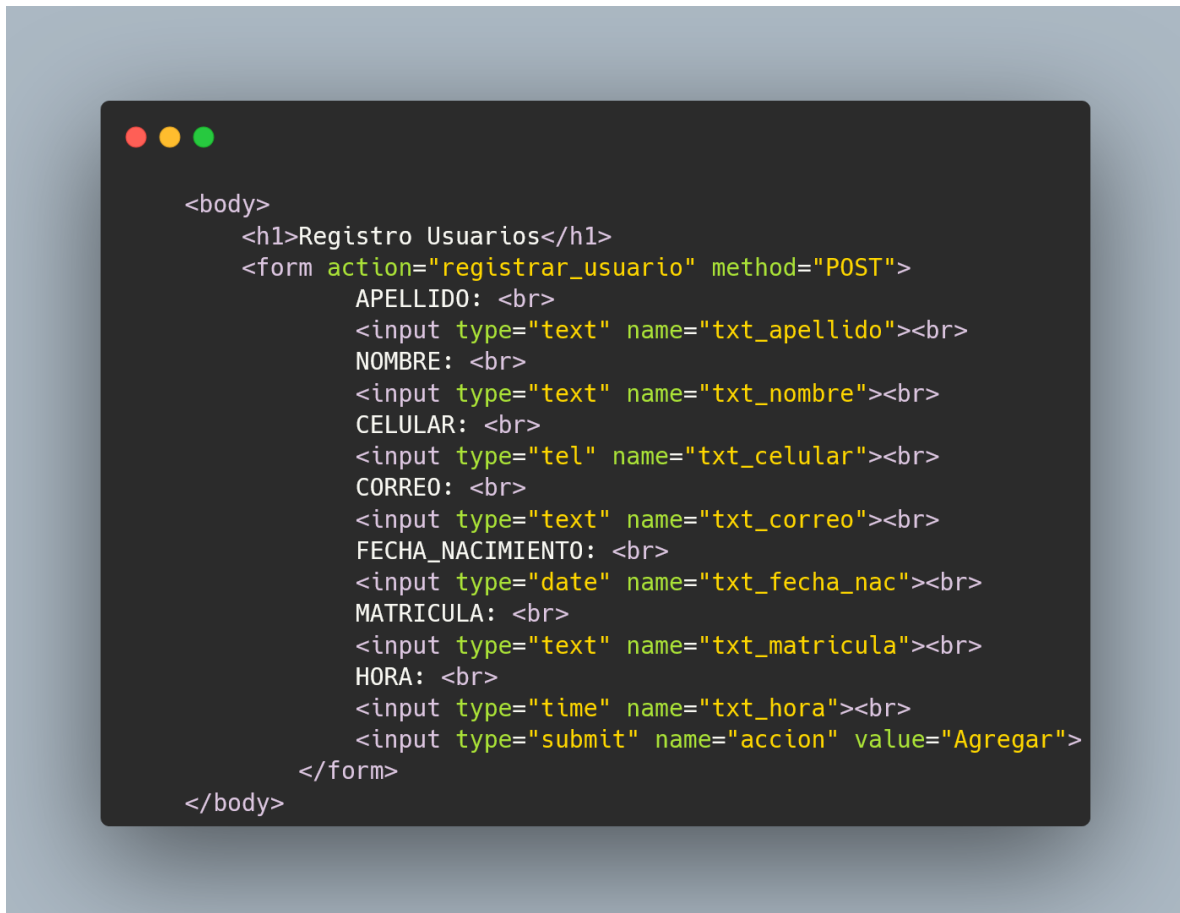
            // Ejecutar la consulta
            int filasInsertadas = ps.executeUpdate();
            if (filasInsertadas > 0) {
                // Si se insertó correctamente, redirigir al usuario a una página de éxito
                request.setAttribute("mensaje", "Usuario registrado con éxito!");
                request.getRequestDispatcher("/resultado.jsp").forward(request, response);
            } else {
                // Si falló, redirigir a una página de error
                request.setAttribute("mensaje", "Error al registrar usuario.");
                request.getRequestDispatcher("/resultado.jsp").forward(request, response);
            }
        } catch (SQLException e) {
            e.printStackTrace();
            request.setAttribute("mensaje", "Ocurrió un error: " + e.getMessage());
            request.getRequestDispatcher("/resultado.jsp").forward(request, response);
        } finally {
            // Cerrar los recursos
            try {
                if (ps != null) {
                    ps.close();
                }
                if (conn != null) {
                    conn.close();
                }
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
    }
}

```

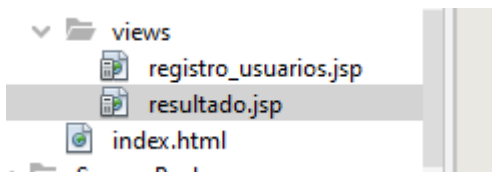
Crear una carpeta llamada: views → registro_usuarios.jsp



Agregar el siguiente código para la vista del formulario:



Crear un nuevo fichero en views



Agregar el siguiente código

```
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>Resultado del Registro</title>
  </head>
  <body>
    <h2>Resultado</h2>
    <p><%= request.getAttribute("mensaje") %></p>
    <a href="inputForm.jsp">Regresar al formulario</a>
  </body>
</html>
```

Mostrar registros

1. Crear un modelo de usuarios

```
/**
 *
 * @author Alejandro Diaz
 */
public class UsuarioModel {
    private String nombre;
    private String correo;
    private String apellidos;
    private String celular;
    private String matricula;
    private Date fecha_nac;
    private Time hora;

    public UsuarioModel() {

    }

    public UsuarioModel(String nombre, String correo, String apellidos, String celular, String
matricula, Date fecha_nac, Time hora) {
        this.nombre = nombre;
        this.correo = correo;
        this.apellidos = apellidos;
        this.celular = celular;
        this.matricula = matricula;
        this.fecha_nac = fecha_nac;
        this.hora = hora;
    }

    public Time getHora() {
        return hora;
    }

    public void setHora(Time hora) {
        this.hora = hora;
    }

    public String getNombre() {
        return nombre;
    }

    public void setNombre(String nombre) {
        this.nombre = nombre;
    }

    public String getCorreo() {
        return correo;
    }

    public void setCorreo(String correo) {
        this.correo = correo;
    }

    public String getApellidos() {
        return apellidos;
    }

    public void setApellidos(String apellidos) {
        this.apellidos = apellidos;
    }

    public String getCelular() {
        return celular;
    }

    public void setCelular(String celular) {
        this.celular = celular;
    }

    public String getMatricula() {
        return matricula;
    }

    public void setMatricula(String matricula) {
        this.matricula = matricula;
    }

    public Date getFecha_nac() {
        return fecha_nac;
    }

    public void setFecha_nac(Date fecha_nac) {
        this.fecha_nac = fecha_nac;
    }

    @Override
    public String toString() {
        return "Usuario{" + "nombre=" + nombre + ", correo=" + correo + ", apellidos=" + apellidos + ",
celular=" + celular + ", matricula=" + matricula + ", fecha_nac=" + fecha_nac + '}';
    }
}
```

2. Agrregar al método get lo siguiente

```
@Override
protected void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    System.out.println("Se ejecuta el doGet");
    List<UsuarioModel> listaUsuarios = new ArrayList<>();
    String sql = "SELECT apellidos, celular, correo, fecha_nac, matricula, nombre, hora FROM
usuario";

    try {
        conn = conexion.getConnectionBD();
        PreparedStatement ps = conn.prepareStatement(sql);
        ResultSet rs = ps.executeQuery();

        // Itera sobre los resultados y crea objetos UsuarioModel
        while (rs.next()) {
            UsuarioModel usuario = new UsuarioModel();
            usuario.setApellidos(rs.getString("apellidos"));
            usuario.setCelular(rs.getString("celular"));
            usuario.setCorreo(rs.getString("correo"));
            usuario.setFecha_nac(rs.getDate("fecha_nac"));
            usuario.setMatricula(rs.getString("matricula"));
            usuario.setNombre(rs.getString("nombre"));
            usuario.setHora(rs.getTime("hora"));
            listaUsuarios.add(usuario);
        }

        // Pasa la lista de usuarios al JSP
        request.setAttribute("usuarios", listaUsuarios);
        request.getRequestDispatcher("/views/mostrar_usuarios.jsp").forward(request, response);

    } catch (Exception e) {
        e.printStackTrace();
        response.sendError(HttpServletResponse.SC_INTERNAL_SERVER_ERROR, "Error al obtener los
usuarios");
    }
}
```

Acceder a la url

<http://localhost:8089/RequirementsAndResponsesExample/usuario>

Delete

Agregar el siguiente método

```
@Override
protected void doDelete(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    ConnectionBD conexion = new ConnectionBD();
    String matricula = request.getParameter("matricula");
    // Validate input
    if (matricula == null || matricula.trim().isEmpty()) {
        response.setStatus(HttpServletResponse.SC_BAD_REQUEST); // Invalid request
        return;
    }

    String sql = "DELETE FROM usuario WHERE matricula like ?";

    try {
        conn = conexion.getConnectionBD();
        ps = conn.prepareStatement(sql);
        ps.setString(1, matricula);

        int rowsAffected = ps.executeUpdate();
        if (rowsAffected > 0) {
            response.setStatus(HttpServletResponse.SC_OK); // Eliminar exitoso
        } else {
            response.setStatus(HttpServletResponse.SC_NOT_FOUND); // No se encontró el usuario
        }
    } catch (Exception e) {
        e.printStackTrace();
        response.setStatus(HttpServletResponse.SC_INTERNAL_SERVER_ERROR); // Error del servidor
    } finally {
        try {
            if (ps != null) {
                ps.close();
            }
            if (conn != null && !conn.isClosed()) {
                conn.close();
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Agregar en mostrar_usuario.jsp lo siguiente:


```

<script>
  function eliminarUsuario(id) {
    if (confirm("¿Estás seguro de que quieres eliminar este usuario?")) {
      fetch(`eliminarUsuario?id=${matricula}`, {
        method: 'DELETE'
      }).then(response => {
        if (response.ok) {
          alert('Usuario eliminado exitosamente');
          location.reload();
        } else {
          alert('Error al eliminar usuario');
        }
      }).catch(error => console.error('Error:', error));
    }
  }
</script>

```

Agregar el botón para eliminar:

```

<td> <button onclick="eliminarUsuario(<%= usuario.getMatricula()%>)">Eliminar</button> </td>

```

Actualizar