INSTITUTO SUPERIOR TECNOLÓGICO PRIVADO CIBERTEC DIRECCIÓN ACADÉMICA CARRERAS PROFESIONALES



NOTA

CURSO : LENGUAJE DE PROGRAMACION II PROFESOR : Ing. Manuel Huertas Honores

SEMESTRE : 2023-2 CICLO : CUARTO SECCION : T4ET-01

FECHA: LUNES 25 DE SEPTIEMBRE DE 2023

DURACION: 90 MINUTOS

ALUMNO (A): CARLOS GABRIEL BACA MANRIQUE

EVALUACIÓN DE LABORATORIO 1

Tenga presente que el logro es que usted, implemente algunas opciones de una aplicación web que utilice **Maven, JPA y MySQL** en la implementación de CRUD utilizando el patrón MVC.

Consideraciones generales:

- El desarrollo de la práctica es estrictamente individual.
- Está permitido el uso de apuntes, libros, materiales o recursos informáticos adicionales.
- Las preguntas deben responderse de acuerdo con los conceptos discutidos o analizados en clase y aplicando su propio criterio para dar una solución a los problemas planteados.

Consolidado

Pregunta	Puntaje		Llenar solo en caso de Recalificación justificada		
	Máximo	Obtenido	Sustento	Puntaje	
1	20				
			Nota Recalificada		

PREGUNTA 1

Usted como desarrollador se le ha asignado la funcionalidad de registro de empleado, para ello debe implementar un CRUD-MVC empleando JPA y MySQL

TABLA : Empleado

IdEmpleado

Apellidos

Nombres

Edad (>=18)

Sexo ('M' o 'F')

Salario (1500 a 2500)

REGISTRO DE EMPLEADOS								
IdEmpleado	Apellidos	Nombres	Edad	Sexo	Salario			
EMP0001	TORRES	JOSE	22	M	1500			
EMP0002	GARCIA	MARIA	21	F	1750			
EMP0003	FLORES	JUAN	22	M	1850			
EMP0004	CASAS	ANA	23	F	2000			

TOTAL DE REGISTROS: 5

Registrar Empleado

NOTA: Una vez terminado el proyecto subirlo a su cuenta de Git y enviar el link correspondiente.

REPOSITORIO: https://github.com/BlackTail2000/CL1_BacaManrique

PROPIEDADES:

APP:

```
package com.baca;

package com.baca;

import org.springframework.boot.SpringApplication;

GspringBootApplication

public class Cl1BacaManriqueApplication {

public static void main(String[] args) {
    SpringApplication.run(Cl1BacaManriqueApplication.class, args);
}

SpringApplication.run(Cl1BacaManriqueApplication.class, args);
}
```

MODEL:

```
1 package com.baca.model;
 3@ import java.math.BigDecimal;
 5 import jakarta.persistence.*;
 6 import lombok.*;
8 @Getter @Setter
9 @NoArgsConstructor
10 @AllArgsConstructor
11 @Entity
12 public class Empleado {
13
14⊖
       @Id
15
       @Column(columnDefinition = "CHAR(7)", name = "Id Empleado")
16
       private String IdEmpleado;
17⊝
      @Column(length = 45)
18
       private String Apellidos;
19⊜
       @Column(length = 45)
20
       private String Nombres;
21
       private int Edad;
      @Column(columnDefinition = "CHAR(1)")
22⊖
23
       private char Sexo;
2.4⊖
      @Column(columnDefinition = "DECIMAL(8,2)")
25
       private BigDecimal Salario;
26 }
27
```

REPOSITORY:

```
package com.baca.repository;

import org.springframework.data.jpa.repository.JpaRepository;

Repository
public interface EmpleadoRepository extends JpaRepository<Empleado, String>{

10
11 }
12
```

SERVICE INTERFACE:

```
1 package com.baca.service;
 2
 3⊖ import java.util.List;
 5
  import com.baca.model.Empleado;
 6
 7
   public interface IEmpleadoService {
 8
 9
       List<Empleado> findAllEmpleados();
10
       Empleado findEmpleadoById(String IdEmpleado);
       void saveEmpleado(Empleado empleado);
11
12
       void removeEmpleado(Empleado empleado);
13 }
14
```

SERVICE IMPLEMENTACIÓN:

```
1 package com.baca.service.impl;
 3⊕ import java.util.List; ...
12
13 @Service
14 public class EmpleadoService implements IEmpleadoService {
15
16⊝
       @Autowired
17
       private EmpleadoRepository empleadoRepository;
18
190
       @Override
      public List<Empleado> findAllEmpleados() {
≥20
21
           return empleadoRepository.findAll();
22
23
24⊖
       @Override
≥25
      public Empleado findEmpleadoById(String IdEmpleado) {
26
           Empleado empleado = null;
27
           Optional < Empleado > optional = empleadoRepository.findById(IdEmpleado);
28
           if(optional.isPresent())
29
               empleado = optional.get();
30
31
           return empleado;
32
      }
33
34⊖
       @Override
       public void saveEmpleado(Empleado empleado) {
35ء
36
           empleadoRepository.save(empleado);
37
38
39⊜
       @Override
40
       public void removeEmpleado(Empleado empleado) {
41
           empleadoRepository.delete(empleado);
42
43
44 }
```

CONTROLLER:

```
1 package com.baca.controller;
 3⊕ import java.math.BigDecimal; ...
17 @Controller
18 public class EmpleadoController {
19
20⊝
       @Autowired
       private IEmpleadoService empleadoService;
21
22
23⊜
       @GetMapping("/Empleado/Listado")
24
       public String Listar(Model model) {
2.5
           List<Empleado> empleados = empleadoService.findAllEmpleados();
26
           model.addAttribute("empleados", empleados);
27
28
            return "ListadoEmpleados";
29
       }
30
31⊜
       @GetMapping("/Empleado/Registro")
32
       public String Registro(Model model) {
33
            Empleado empleado = new Empleado();
34
            empleado.setEdad(18);
35
            empleado.setSalario(new BigDecimal(1500));
36
           model.addAttribute("empleado", empleado);
37
           return "RegistrarEmpleado";
38
       }
39
       @PostMapping("/Empleado/Registrar")
40⊜
41
       public String Registrar(@ModelAttribute("empleado") Empleado empleado) {
            empleadoService.saveEmpleado(empleado);
42
43
            return "redirect:/Empleado/Listado";
44
      }
45
      @GetMapping("/Empleado/Eliminar")
46⊖
      public String Eliminar(@RequestParam("id") String IdEmpleado) {
47
48
          Empleado empleado = empleadoService.findEmpleadoById(IdEmpleado);
49
          empleadoService.removeEmpleado(empleado);
50
51
          return "redirect:/Empleado/Listado";
52
53
54⊖
      @GetMapping("/Empleado/Actualizacion")
55
      public String Actualizacion(@RequestParam("id") String IdEmpleado, Model model) {
          Empleado empleado = empleadoService.findEmpleadoById(IdEmpleado);
56
57
          model.addAttribute("empleado", empleado);
58
          return "ActualizarEmpleado";
59
      @PostMapping("/Empleado/Actualizar")
61⊖
62
      public String Actualizar(@ModelAttribute("empleado") Empleado empleado) {
63
          empleadoService.saveEmpleado(empleado);
          return "redirect:/Empleado/Listado";
64
65
66 }
```

REGISTRAR:

LISTAR:

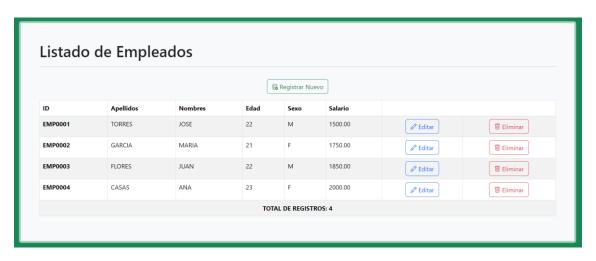
```
| Section | Sect
```

ACTUALIZAR:

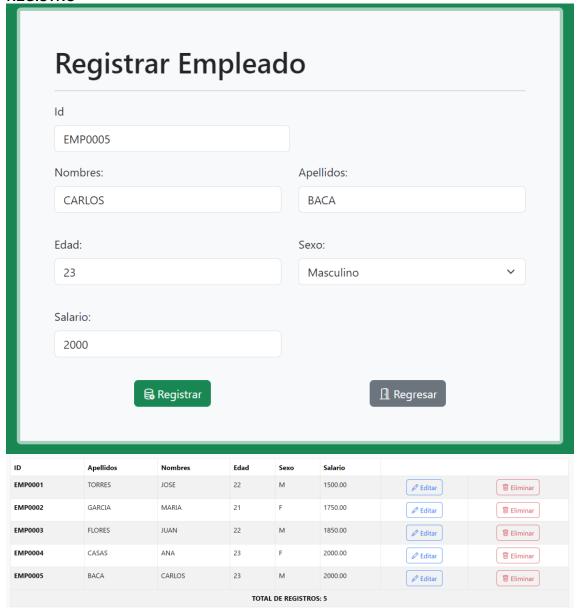
```
| Section | Sect
```

EVIDENCIA:

LISTADO



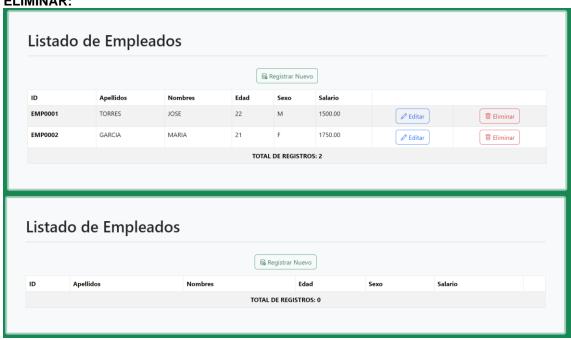
REGISTRO



EDITAR:



ELIMINAR:



	RÚBRICA PARA EL EXAMEN FINAL								
PUNTOS	EXCELENTE 20 PUNTOS	BUENO 16 PUNTOS	REGULAR 12 PUNTOS	REGULAR 8 PUNTOS	DEFICIENTE 0 PUNTOS				
20 Puntos	Crear correctamente el proyecto Web Dinámico con Maven (carga de dependencias en su archivo de configuración), implementa correctamente el patrón MVC para las clases correspondiente s, configura correctamente el archivo de configuración para la conexión con MySQL Hace uso del repositorio de JPA para realizar los procesos de listado y registro. Carga el proyecto y lo comparte a través de su cuenta de Git.	Crear correctamente el proyecto Web Dinámico con Maven (carga de dependencias en su archivo de configuración), implementa parcialmente el patrón MVC para las clases correspondiente s, configura correctamente el archivo de configuración para la conexión con MySQL Hace uso del repositorio de JPA para realizar los procesos de listado y registro. Carga el proyecto y lo comparte a través de su cuenta de Git.	Crear correctamente el proyecto Web Dinámico con Maven (carga de dependencias en su archivo de configuración) , implementa parcialmente el patrón MVC para las clases correspondien tes, configura correctamente el archivo de configuración para la conexión con MySQL Hace uso del repositorio de JPA para realizar el proceso de listado-	Crear correctamente el proyecto Web Dinámico con Maven (carga de dependencias en su archivo de configuración) , implementa parcialmente el patrón MVC para las clases correspondien tes, configura correctamente el archivo de configuración para la conexión con MySQL	No implementa Ninguna funcionalidad.				