



Reporte final

Datos eliminados: matrícula, correo, celular, teléfono; con fundamento en lo establecido en los artículos 65, fracción II, 98, fracción III, 113, fracción I y último párrafo de la Ley Federal de Transparencia y Acceso a la Información Pública, 44, fracción II, 106, fracción III y 116 de la Ley General de Transparencia y Acceso a la Información Pública, así como a lo establecido en los Lineamientos Generales en Materia de Clasificación y Desclasificación de la Información, así como para la elaboración de versiones públicas, por tratarse de datos personales concernientes a una persona física identificada, en la modalidad de confidencial. Área Dirección Adjunta de Innovación y Conocimiento. Periodo indefinido, toda vez que no está sujeta a temporalidad alguna y solo podrán tener acceso a ella los titulares de la misma. Aprobada en la Primera Sesión Extraordinaria del Comité de Transparencia de INFOTEC ejercicio 2025.

Índice

Introducción	3
Desarrollo	3
Práctica profesional	36
Estructura organizacional	36
• El sector de actividad	36
• Presentación de la empresa e historia de la empresa	37
Informe de la práctica profesional	38
• Descripción de las funciones	38
• Desglose de actividades	38
• Bitácora de prácticas	39
Conclusiones	41
Cuadro CQA de mi estancia en la organización	41
Evaluación de desempeño	42
Referencias	42

Introducción

Durante la estancia en el programa de TSU en Desarrollo de Software, aplicamos conocimientos esenciales como la programación orientada a objetos y utilizamos diversos lenguajes y herramientas para el desarrollo de proyectos. Entre los lenguajes empleados tenemos HTML, CSS, JavaScript, y framework como Bootstrap y Angular. También hicimos uso de tecnologías como SQL, junto con herramientas como Visual Studio Code, Spring Boot Tools, PostgreSQL, Postman y la consola, lo que nos permitió plasmar nuestras habilidades y llevar a cabo desarrollos funcionales.

En conjunto con los lenguajes y herramientas mencionados, desarrollaremos un 'Sistema Gestor de Tareas' basado en una base de datos relacional creada con PostgreSQL. Esta base de datos incluirá cinco entidades y estará conectada a nuestro proyecto frontend desarrollado en Angular. A través de esta integración, implementaremos funcionalidades como consultas, adiciones, actualizaciones y eliminación de tareas, siguiendo el modelo CRUD e incorporando validaciones para garantizar la calidad y consistencia del sistema.

Este proyecto no solo nos permitirá consolidar los conocimientos adquiridos previamente, sino también aprender nuevas habilidades que contribuirán a desarrollar un sistema más robusto y funcional.

Desarrollo

Instalación de Herramientas de trabajo

Para el manejo del código que estaremos utilizando para el frontend, instalaremos Visual Studio Code.

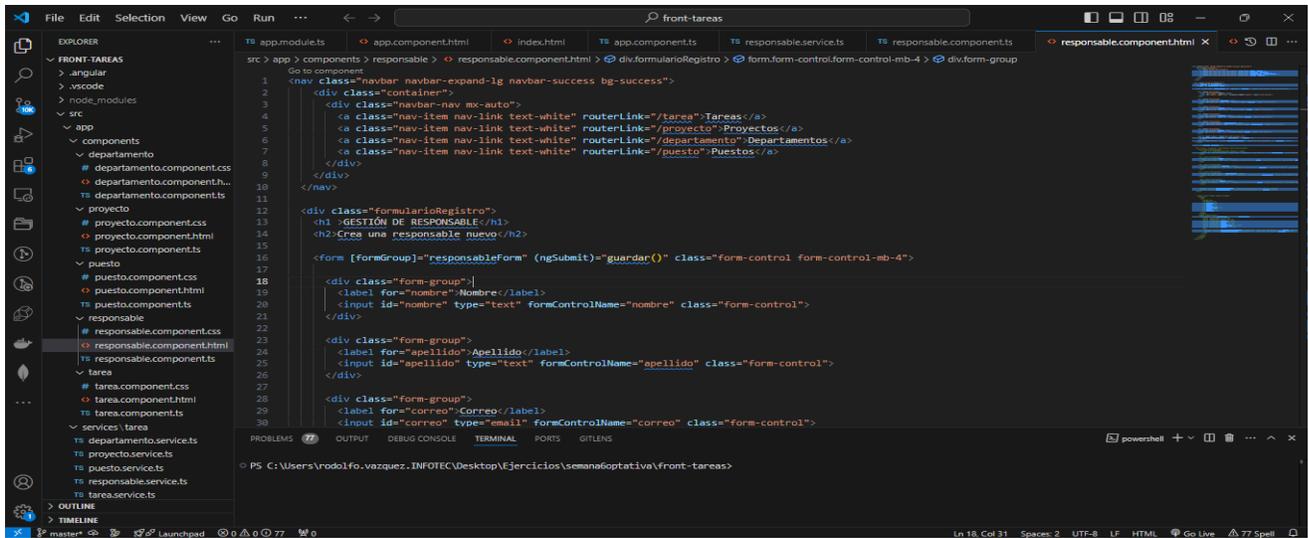


Imagen de elaboración propia.

Para el manejo del código del que estaremos utilizando para el backend, instalaremos Spring Tools.

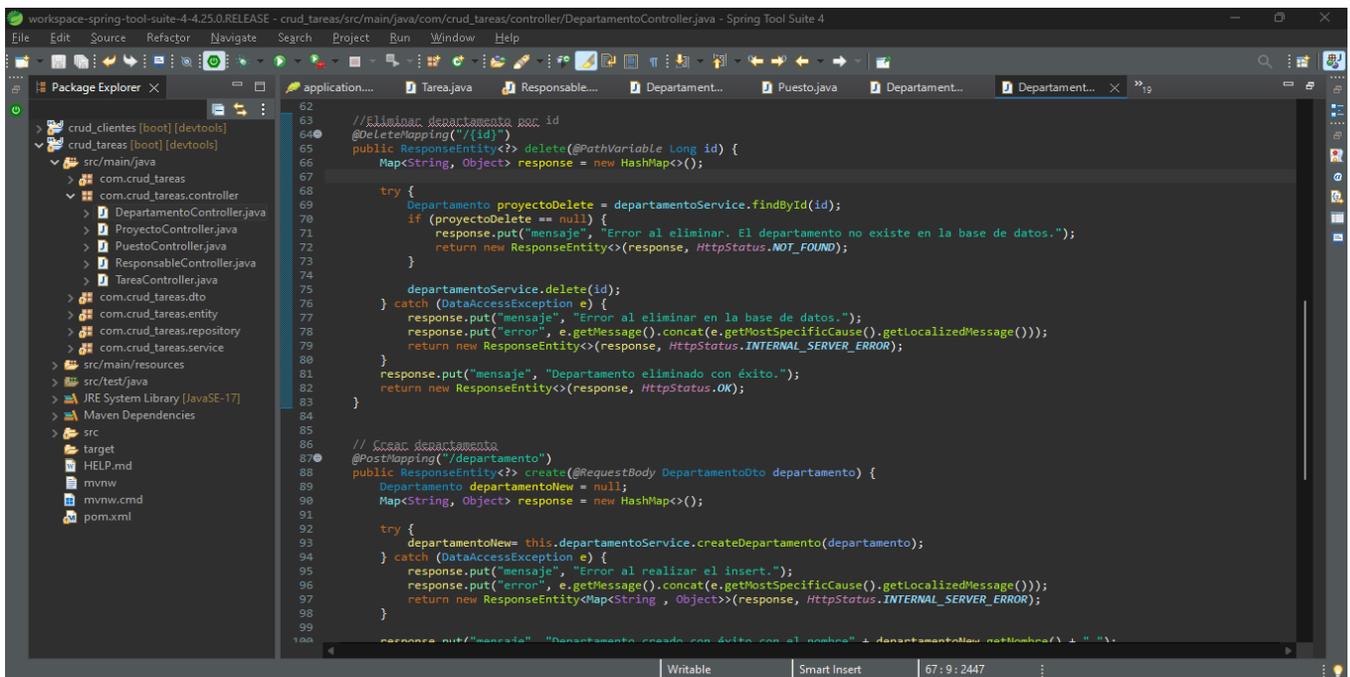


Imagen de elaboración propia.

Para la administración de la base de datos que estaremos utilizando en conjunto con lo generado en el backend, instalaremos pg Admin de PostgreSQL.



Imagen de elaboración propia.

Para realizar las pruebas de la base de datos que estaremos utilizando Postman.

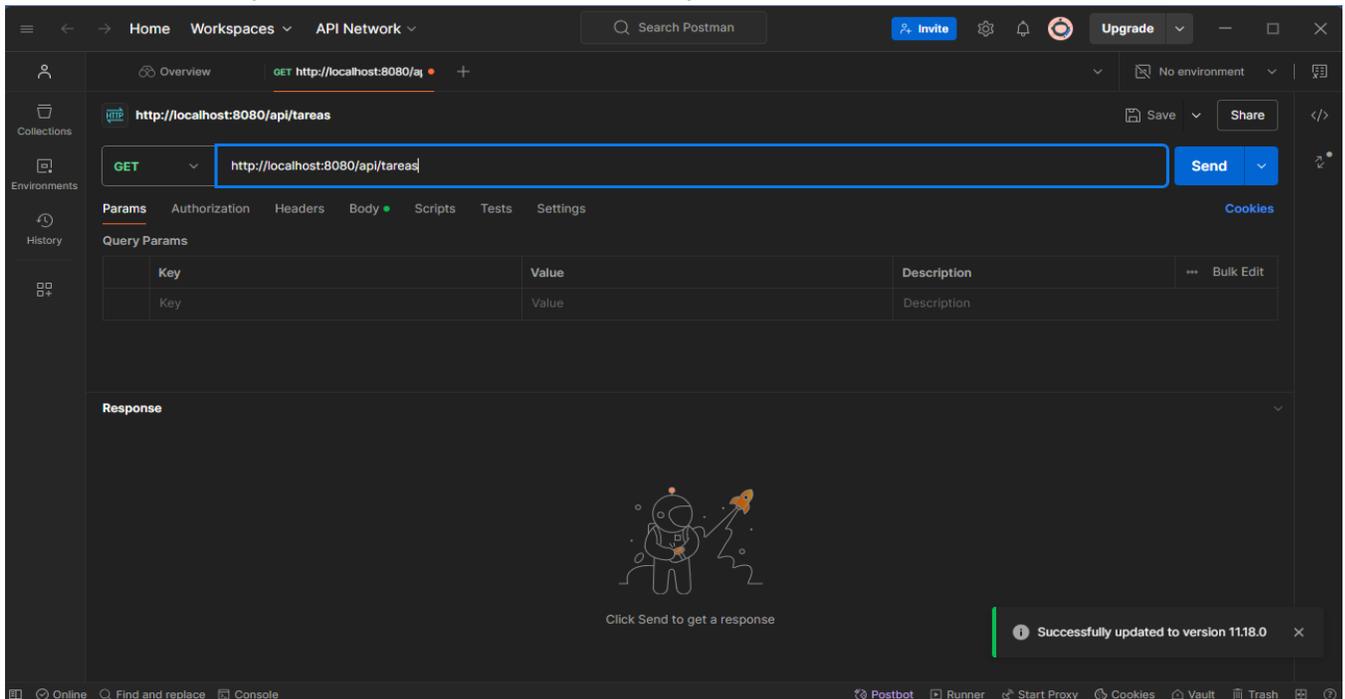


Imagen de elaboración propia.

Iniciaremos con la creación de nuestra base de datos con Spring Tools, en donde inicialmente crearemos nuestro proyecto, dando clic en **File, New y Spring Starter Project**.

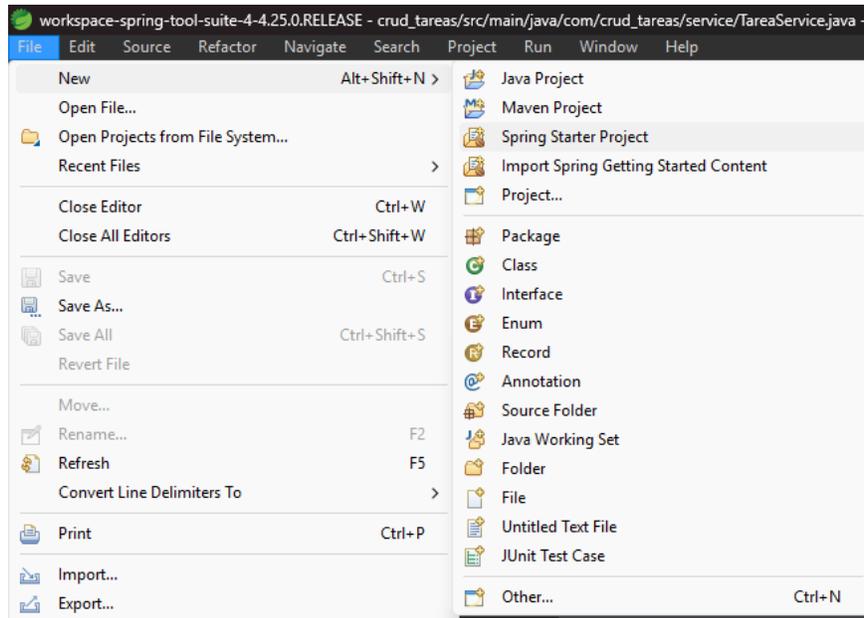


Imagen de elaboración propia.

Ahora le daremos un nombre a nuestro proyecto y daremos clic en **Next**.

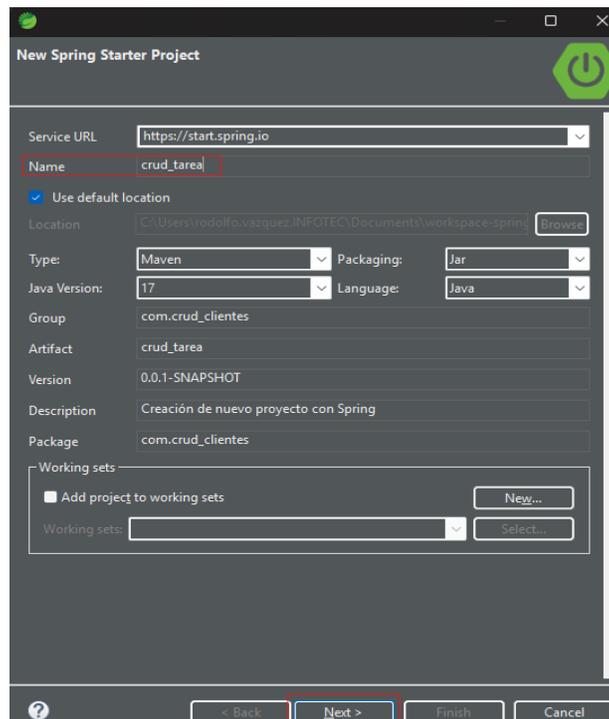


Imagen de elaboración propia.

Seleccionaremos las dependencias que estaremos usando y damos clic en **Finish**.

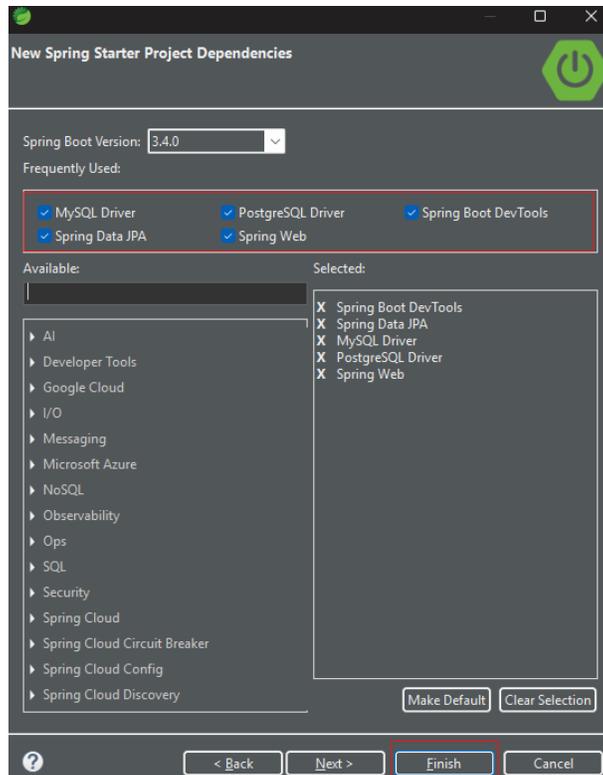


Imagen de elaboración propia.

Una vez creado nuestro proyecto, iniciaremos agregando nuestros paquetes de nuestra base de datos, los cuales serán nuestro paquete **Controller, Dto, Entity, Respository y Service**.

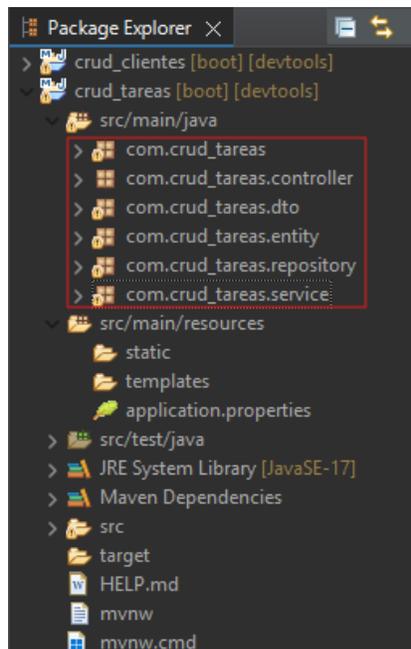


Imagen de elaboración propia.

En este caso estaremos usando 5 entidades (Tarea, Responsable, Proyecto, Departamento y Puesto), por lo que dentro de cada paquete crearemos un archivo por cada entidad correspondiente a cada paquete que generamos.

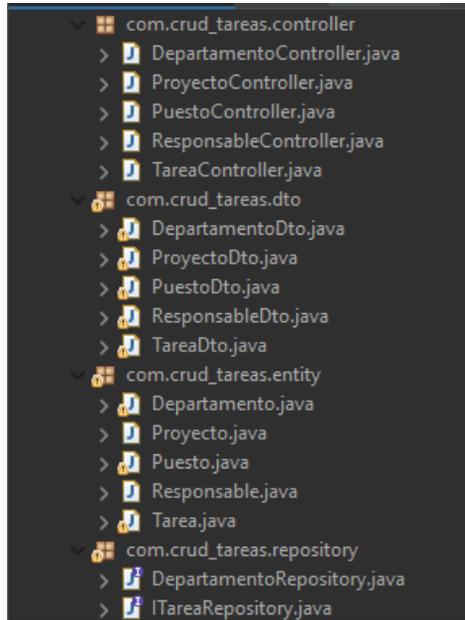


Imagen de elaboración propia.

Bien, una vez que tengamos creados nuestros paquetes y nuestros archivos, iniciaremos con la generación del código de nuestra base de datos.

Package crud_tareas.controller

Tarea.controller

Aquí vamos a declarar la ruta en la que vamos a tener mapeado nuestra página, y en donde declararemos nuestros métodos de búsqueda, creación, actualización y eliminación del CRUD de nuestra base de datos.

```

TareaDto.java Tarea.java TareaServic... Responsable... TareaControl...
1 package com.crud_tareas.controller;
2
3 import java.util.List;
31
32
33 @CrossOrigin(origins= {"http://localhost:4200"})
34 @RestController
35 @RequestMapping("/api/tareas")
36 public class TareaController {
37
38
39     @Autowired
40     private TareaService tareaService;
41
42     @Autowired
43     private ProyectoService proyectoService;
44
45     @Autowired
46     private ResponsableService responsableService;
47
48     @GetMapping
49     @ResponseStatus(HttpStatus.OK)
50
51     public List<Tarea> consulta(){
52     return tareaService.findAllWithResponsable();
53     }
54
55     //Realizar búsqueda por ID
56     @GetMapping("/{id}")
57     public ResponseEntity<?> consultaPorID(@PathVariable Long id) {
58
59         Tarea tarea=null;
60         String response="";
61         try {

```

Imagen de elaboración propia.

Package crud_tareas.dto

Tarea.dto

Aquí es en donde vamos a declarar las variables para los campos de nuestras entidades, así como sus correspondientes "getters y setters".

```

1 package com.crud_tareas.dto;
2
3 import java.io.Serializable;
11
12 public class TareaDto implements Serializable {
13
14     private String nombre;
15     private String prioridad;
16     private String estado;
17     private Long proyectoId;
18     private Long responsableId;
19     private LocalDateTime createAt;
20     private LocalDateTime createAtc;
21
22     // Getters y Setters
23     public String getNombre() { return nombre; }
24     public void setNombre(String nombre) { this.nombre = nombre; }
25
26     public String getPrioridad() { return prioridad; }
27     public void setPrioridad(String prioridad) { this.prioridad = prioridad; }
28
29     public String getEstado() { return estado; }
30     public void setEstado(String estado) { this.estado = estado; }
31
32     public Long getProyectoId() { return proyectoId; }
33     public void setProyectoId(Long proyectoId) { this.proyectoId = proyectoId; }
34
35     public Long getResponsableId() { return responsableId; }
36     public void setResponsableId(Long responsableId) { this.responsableId = responsableId; }
37
38     public LocalDateTime getCreateAt() { return createAt; }
39     public void setCreateAt(LocalDateTime createAt) { this.createAt = createAt; }
40
41     public LocalDateTime getCreateAtc() { return createAtc; }

```

Imagen de elaboración propia.

Package crud_tareas.entity

Tarea.entity

Aquí vamos a definir las columnas de nuestras tablas y los valores que va a estar tomando de acuerdo a nuestros campos declarados, así como las relaciones que se van a tener con otras tablas y que tipo de relación van a tener estas.

```

1 package com.crud_tareas.entity;
2
3 import java.time.LocalDateTime;
18
19 @Entity
20 @Table(name = "tarea")
21 public class Tarea {
22
23     @Id
24     @GeneratedValue(strategy = GenerationType.IDENTITY)
25     private Long id;
26
27     @Column(name="nombreTarea")
28     private String nombre;
29     private String prioridad;
30     private String estado;
31
32     @Column(name="fecha_registro")
33     private LocalDateTime createAt;
34
35     @Column(name="fecha_cierre")
36     private LocalDateTime createAtc;
37
38     @ManyToOne(fetch = FetchType.LAZY) //Relación de muchas tareas a un responsable
39     @JoinColumn(name = "responsable_id")
40     @JsonIgnoreProperties({"hibernateLazyInitializer", "handler"})
41     private Responsable responsable;
42
43     public Responsable getResponsable() {
44         return responsable;
45     }
46
47     public void setResponsable(Responsable responsable) {
48         this.responsable = responsable;

```

Imagen de elaboración propia.

Package crud_tareas.repository

Tarea.repository

Aquí estamos creando una interfaz como un componente que interactúa con la base de datos, permitiendo que Spring gestione la inyección de dependencias de este repositorio en otras partes de tu aplicación, y permitiéndonos agregar métodos para el manejo de la entidad Tarea (Tarea.entity).

```

1 package com.crud_tareas.repository;
2
3 import java.util.List;
4
5
6
7
8
9
10 @Repository
11 public interface ITareaRepository extends JpaRepository <Tarea, Long>{
12     @Query("SELECT t FROM Tarea t JOIN FETCH t.responsable")
13     List<Tarea> findAllWithResponsable();
14 }
15

```

Imagen de elaboración propia.

Package crud_tareas.service

Tarea.service

Aquí nos ayudan a interactuar con el controlador quien es quien maneja las solicitudes y respuestas (solicitudes del cliente), y con el repositorio que es el que interactúa directamente con la base de datos.

```

1 package com.crud_tareas.service;
2
3 import java.time.LocalDate;
4
5
6
7
8
9
10
11
12
13
14
15
16 @Service
17 public class TareaService {
18
19     @Autowired
20     private ITareaRepository tareaRepository;
21
22     @Autowired
23     private ResponsableService responsableService;
24
25     @Autowired
26     private ProyectoService proyectoService;
27
28     public List<Tarea> findAllWithResponsable() {
29         return tareaRepository.findAllWithResponsable();
30     }
31
32     //Consulta de todas las tareas
33     @Transactional(readOnly = true)
34     public List<Tarea> findAll() {
35         return (List<Tarea>)tareaRepository.findAll();
36     }
37
38
39     //Consulta por ID
40     @Transactional(readOnly = true)
41     public Tarea findById(Long id) {
42         return (Tarea) tareaRepository.findById(id).orElse(null);
43     }
44

```

Imagen de elaboración propia.

A continuación, dejare la evidencia del resto de los archivos generados.

Responsable.controller

```
TareaDto.java  Tarea.java  TareaServic...  Responsable...  Depart...
1 package com.crud_tareas.dto;
2
3 import java.io.Serializable;
4
5
6 public class ResponsableDto implements Serializable {
7     private String nombre;
8     private String apellido;
9     private String correo;
10    private String celular;
11    private DepartamentoDto departamento;
12    private PuestoDto puesto;
13    public String getNombre() {
14        return nombre;
15    }
16    public void setNombre(String nombre) {
17        this.nombre = nombre;
18    }
19    public String getApellido() {
20        return apellido;
21    }
22    public void setApellido(String apellido) {
23        this.apellido = apellido;
24    }
25    public String getCorreo() {
26        return correo;
27    }
28    public void setCorreo(String correo) {
29        this.correo = correo;
30    }
31    public String getCelular() {
32        return celular;
33    }
34    public void setCelular(String celular) {
35        this.celular = celular;
36    }
37    public DepartamentoDto getDepartamento() {
38        return departamento;
39    }
40    public void setDepartamento(DepartamentoDto departamento) {
41        this.departamento = departamento;
42    }
43    public PuestoDto getPuesto() {
44        return puesto;
45    }
46    public void setPuesto(PuestoDto puesto) {
```

Imagen de elaboración propia.

Proyecto.controller

```

TareaDto.java | Tarea.java | TareaServic... | Responsable... | PuestoContr... | TareaControl... | ResponsableC...
1 package com.crud_tareas.controller;
2
3 import java.util.HashMap;
4
5
6 @CrossOrigin(origins= {"http://localhost:4200"})
7 @RestController
8 @RequestMapping("/api/proyecto")
9 public class ProyectoController {
10
11     @Autowired
12     private ProyectoService proyectoService;
13
14
15     //consulta de todas las proyectos
16     @GetMapping("")
17     @ResponseStatus(HttpStatus.OK)
18
19     public List<Proyecto> consulta(){
20     return proyectoService.findAll();
21     }
22
23     //consulta de proyecto por id
24     @GetMapping("/{id}")
25     public ResponseEntity<?> consultaPorId(@PathVariable Long id) {
26     Proyecto proyecto = null;
27     String response = "";
28     try {
29     proyecto = proyectoService.findById(id);
30     } catch (DataAccessException e) {
31     response = "Error al realizar la consulta: " + e.getMessage();
32     return new ResponseEntity<>(response, HttpStatus.INTERNAL_SERVER_ERROR);
33     }
34
35     if (proyecto == null) {
36     response = "El proyecto con el id ".concat(id.toString()).concat(" no existe en la base de datos");
37     return new ResponseEntity<>(response, HttpStatus.NOT_FOUND);
38     }
39
40     return new ResponseEntity<>(proyecto, HttpStatus.OK);
41     }
42
43 }

```

Imagen de elaboración propia.

Departamento.controller

```

TareaDto.java | Tarea.java | TareaServic... | Responsable... | Departamento... | TareaControl... | ResponsableC...
1 package com.crud_tareas.controller;
2
3 import java.util.HashMap;
4
5
6 @CrossOrigin(origins= {"http://localhost:4200"})
7 @RestController
8 @RequestMapping("/api/departamento")
9 public class DepartamentoController {
10
11     @Autowired
12     private DepartamentoService departamentoService;
13
14
15     //Obtengan a todas las departamentos
16     @GetMapping("")
17     @ResponseStatus(HttpStatus.OK)
18
19     public List<Departamento> consulta(){
20     return departamentoService.findAll();
21     }
22
23     //Obtener departamento por id
24     @GetMapping("/{id}")
25     public ResponseEntity<?> consultaPorId(@PathVariable Long id) {
26     Departamento departamento = null;
27     String response = "";
28
29     try {
30     departamento = departamentoService.findById(id);
31     } catch (DataAccessException e) {
32     response = "Error al realizar la consulta: " + e.getMessage();
33     return new ResponseEntity<>(response, HttpStatus.INTERNAL_SERVER_ERROR);
34     }
35
36     if (departamento == null) {
37     response = "El departamento con el id ".concat(id.toString()).concat(" no existe en la base de datos");
38     return new ResponseEntity<String>(response, HttpStatus.NOT_FOUND);
39     }
40
41     return new ResponseEntity<Departamento>(departamento, HttpStatus.OK);
42     }
43
44 }

```

Imagen de elaboración propia.

Puesto.controller

```

TareaDto.java Tarea.java TareaServic... Responsable... Departament... PuestoContr... TareaCon
1 package com.crud_tareas.controller;
2
3 import java.util.HashMap;
4
25
26 @CrossOrigin(origins= {"http://localhost:4200"})
27 @RestController
28 @RequestMapping("/api/puesto")
29 public class PuestoController {
30
31     @Autowired
32     private PuestoService puestoService;
33
34
35     //Busqueda de todos los puestos
36     @GetMapping("")
37     @ResponseStatus(HttpStatus.OK)
38
39     public List<Puesto> consulta(){
40     return puestoService.findAll();
41     }
42
43
44     //Busqueda de los puestos por ID
45     @GetMapping("/{id}")
46     public ResponseEntity<?> consultaPorId(@PathVariable Long id) {
47         Puesto puesto = null;
48         String response = "";
49         try {
50             puesto = puestoService.findById(id);
51         } catch (DataAccessException e) {
52             response = "Error al realizar la consulta: " + e.getMessage();
53             return new ResponseEntity<String>(response, HttpStatus.INTERNAL_SERVER_ERROR);
54         }
55
56         if (puesto == null) {
57             response = "El puesto con el id ".concat(id.toString()).concat(" no existe en la base de datos");
58             return new ResponseEntity<>(response, HttpStatus.NOT_FOUND);
59         }
60
61         return new ResponseEntity<Puesto>(puesto, HttpStatus.OK);
62     }
63

```

Imagen de elaboración propia.

Responsable.dto

```

TareaDto.java Tarea.java TareaServic... Departament...
1 package com.crud_tareas.dto;
2
3 import java.io.Serializable;
4
7
8
9 public class ResponsableDto implements Serializable {
10
11     private String nombre;
12     private String apellido;
13     private String correo;
14     private String celular;
15     private DepartamentoDto departamento;
16     private PuestoDto puesto;
17     public String getNombre() {
18         return nombre;
19     }
20     public void setNombre(String nombre) {
21         this.nombre = nombre;
22     }
23     public String getApellido() {
24         return apellido;
25     }
26     public void setApellido(String apellido) {
27         this.apellido = apellido;
28     }
29     public String getCorreo() {
30         return correo;
31     }
32     public void setCorreo(String correo) {
33         this.correo = correo;
34     }
35     public String getCelular() {
36         return celular;
37     }
38     public void setCelular(String celular) {
39         this.celular = celular;
40     }
41     public DepartamentoDto getDepartamento() {
42         return departamento;
43     }
44     public void setDepartamento(DepartamentoDto departamento) {
45         this.departamento = departamento;
46     }

```

Imagen de elaboración propia.

Proyecto.dto

```
TareaDto.java | Tarea.java | TareaServic... | Departar
1 package com.crud_tareas.dto;
2
3 import java.io.Serializable;
4
5 public class ProyectoDto implements Serializable {
6
7     private String nombre;
8
9     private String descripcion;
10
11     public String getNombre() {
12         return nombre;
13     }
14
15     public void setNombre(String nombre) {
16         this.nombre = nombre;
17     }
18
19     public String getDescripcion() {
20         return descripcion;
21     }
22
23     public void setDescripcion(String descripcion) {
24         this.descripcion = descripcion;
25     }
26
27 }
28
```

Imagen de elaboración propia.

Departamento.dto

```
TareaDto.java | Tarea.java | TareaServic... | Departan
1 package com.crud_tareas.dto;
2
3 import java.io.Serializable;
4
5 public class DepartamentoDto implements Serializable {
6
7     private Long id;
8
9     private String nombre;
10
11     public String getNombre() {
12         return nombre;
13     }
14
15     public void setNombre(String nombre) {
16         this.nombre = nombre;
17     }
18
19     public Long getId() {
20         return id;
21     }
22
23     public void setId(Long id) {
24         this.id = id;
25     }
26
27 }
```

Imagen de elaboración propia.

Puesto.dto

```

1 package com.crud_tareas.dto;
2
3 import java.io.Serializable;
4
5 public class PuestoDto implements Serializable {
6
7     private Long id;
8
9     private String nombre;
10
11     public String getNombre() {
12         return nombre;
13     }
14
15     public void setNombre(String nombre) {
16         this.nombre = nombre;
17     }
18
19     public Long getId() {
20         return id;
21     }
22
23     public void setId(Long id) {
24         this.id = id;
25     }
26
27 }

```

Imagen de elaboración propia.

Responsable.entity

```

1 package com.crud_tareas.entity;
2
3 import java.util.List;
4
5 @Entity
6 @Table(name = "responsable")
7 public class Responsable {
8
9     @Id
10    @GeneratedValue(strategy = GenerationType.IDENTITY)
11    private Long id;
12    @Column(name="nombreResponsable")
13    private String nombre;
14    @Column(name="apellidoResponsable")
15    private String apellido;
16    private String correo;
17    @Column(name = "celular")
18    private String celular;
19
20    @ManyToOne(fetch = FetchType.LAZY) //Relación de un responsable a un departamento
21    @JoinColumn(name = "departamento_id")
22    @JsonIgnoreProperties({"hibernateLazyInitializer", "handler"})
23    private Departamento departamento;
24
25    @ManyToOne(fetch = FetchType.EAGER) //Relación de un responsable a un puesto
26    @JoinColumn(name = "puesto_id")
27    @JsonIgnoreProperties({"hibernateLazyInitializer", "handler"})
28    private Puesto puesto;
29
30    @OneToMany(mappedBy = "responsable", fetch = FetchType.LAZY) // Relación de un responsable a muchas tareas
31    @JsonBackReference // Se mantiene @JsonBackReference aquí para evitar ciclos de referencia
32    private List<Tarea> tareas; // Relación con tareas
33
34    public Long getId() {
35        return id;
36    }
37
38    public void setId(Long id) {
39        this.id = id;
40    }
41
42    public String getNombre() {
43        return nombre;
44    }
45
46 }

```

Imagen de elaboración propia.

Proyecto.entity

```

1 package com.crud_tareas.entity;
2
3 import java.util.List;
4
13
14 @Entity
15 @Table (name = "proyecto")
16 public class Proyecto {
17
18     @Id
19     @GeneratedValue(strategy = GenerationType.IDENTITY)
20     private Long id;
21     @Column (name="nombreProyecto")
22     private String nombre;
23     private String descripcion;
24     public Long getId() {
25         return id;
26     }
27
28     @OneToMany(mappedBy = "proyecto", fetch = FetchType.EAGER) // Relación de un proyecto a muchas tareas
29     private List<Tarea> tareas;
30
31     public void setId(Long id) {
32         this.id = id;
33     }
34     public String getNombre() {
35         return nombre;
36     }
37     public void setNombre(String nombre) {
38         this.nombre = nombre;
39     }
40     public String getDescripcion() {
41         return descripcion;
42     }
43     public void setDescripcion(String descripcion) {
44         this.descripcion = descripcion;
45     }
46 }
47 }
48

```

Imagen de elaboración propia.

Departamento.entity

```

1 package com.crud_tareas.entity;
2
3 import java.util.List;
4
13
14 @Entity
15 @Table (name = "departamento")
16 public class Departamento {
17
18     @Id
19     @GeneratedValue(strategy = GenerationType.IDENTITY)
20
21     private Long id;
22
23     @Column (name="nombreDepartamento")
24     private String nombre;
25
26
27     @OneToOne(mappedBy = "departamento", fetch = FetchType.EAGER) // Relación de un responsable a un departamento
28     private Responsable responsable;
29
30
31     public Long getId() {
32         return id;
33     }
34     public void setId(Long id) {
35         this.id = id;
36     }
37     public String getNombre() {
38         return nombre;
39     }
40     public void setNombre(String nombre) {
41         this.nombre = nombre;
42     }
43 }
44 }
45

```

Imagen de elaboración propia.

Puesto.entity

```

1 package com.crud_tareas.entity;
2
3 import java.util.List;
4
5
6
7
8
9
10
11
12
13
14
15
16
17 @Entity
18 @Table (name = "puesto")
19 public class Puesto {
20
21
22
23     @Id
24     @GeneratedValue(strategy = GenerationType.IDENTITY)
25     private Long id;
26
27
28     @Column (name="nombrePuesto")
29     private String nombre;
30
31     @ManyToOne(mappedBy = "puesto", fetch = FetchType.EAGER) // Relación de un responsable a un puesto
32     @JsonIgnoreProperties({"puesto"}) // Ignora la propiedad "puesto" de Responsable para evitar el bucle
33     @JsonIgnore
34     private Responsable responsable;
35
36
37     public Long getId() {
38         return id;
39     }
40
41     public void setId(Long id) {
42         this.id = id;
43     }
44
45     public String getNombre() {
46         return nombre;
47     }
48
49     public void setNombre(String nombre) {
50         this.nombre = nombre;
51     }
52
53     public Responsable getResponsable() {
54         return responsable;
55     }
56
57     public void setResponsable(Responsable responsable) {
58         this.responsable = responsable;
59     }
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

```

Imagen de elaboración propia.

Responsable.repository

```

1 package com.crud_tareas.repository;
2
3 import org.springframework.data.jpa.repository.JpaRepository;
4
5
6
7
8
9 @Repository
10 public interface ResponsableRepository extends JpaRepository<Responsable, Long> {
11
12     Responsable save(ResponsableDto responsable);
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

```

Imagen de elaboración propia.

Proyecto.repository

```

1 package com.crud_tareas.repository;
2
3 import org.springframework.data.jpa.repository.JpaRepository;
4
5
6
7
8
9 @Repository
10 public interface ProyectoRepository extends JpaRepository<Proyecto, Long> {
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

```

Imagen de elaboración propia.

Departamento.repository

```

1 package com.crud_tareas.repository;
2
3 import org.springframework.data.jpa.repository.JpaRepository;
4
5
6
7
8 @Repository
9 public interface DepartamentoRepository extends JpaRepository<Departamento, Long> {}
10 }
    
```

Imagen de elaboración propia.

Puesto.repository

```

1 package com.crud_tareas.repository;
2
3 import org.springframework.data.jpa.repository.JpaRepository;
4
5
6
7
8 @Repository
9 public interface PuestoRepository extends JpaRepository<Puesto, Long> { }
10 }
    
```

Imagen de elaboración propia.

Responsable.service

```

1 package com.crud_tareas.service;
2
3 import java.util.List;
4
5 @Service
6 public class ResponsableService {
7
8     @Autowired
9     private ResponsableRepository responsableRepository;
10
11     @Autowired
12     private PuestoService puestoService;
13
14     @Autowired
15     private DepartamentoService departamentoService;
16
17     //Consulta de todos los responsables
18     @Transactional(readOnly = true)
19     public List<Responsable> findAll() {
20         return (List<Responsable>)responsableRepository.findAll();
21     }
22
23     //Consulta por ID
24     @Transactional(readOnly = true)
25     public Responsable findById(Long id) {
26         return (Responsable) responsableRepository.findById(id).orElse(null);
27     }
28
29
30     //Crear nueva responsable
31     @Transactional
32     public Responsable createResponsable (ResponsableDto responsable) {
33         Responsable responsableEntity= new Responsable();
34         responsableEntity.setNombre(responsable.getNombre());
35         responsableEntity.setApellido(responsable.getApellido());
36         responsableEntity.setCorreo(responsable.getCorreo());
37         responsableEntity.setCelular(responsable.getCelular());
38
39         // Obtener el departamento y el puesto a partir de sus IDs
40         if (responsable.getDepartamento() != null) {
41             Departamento departamento = departamentoService.findById(responsable.getDepartamento().getId());
42             responsableEntity.setDepartamento(departamento);
43         }
44     }
45 }
    
```

Imagen de elaboración propia.

Proyecto.service

```

1 package com.crud_tareas.service;
2
3 import java.util.List;
4
15 @Service
16 public class ProyectoService {
17
18     @Autowired
19     private ProyectoRepository proyectoRepository;
20
21     //Consulta de todos los proyectos
22     @Transactional(readOnly = true)
23     public List<Proyecto> findAll() {
24         return (List<Proyecto>)proyectoRepository.findAll();
25     }
26
27     //Consulta mediante ID
28     @Transactional(readOnly = true)
29     public Proyecto findById(Long id) {
30         return (Proyecto) proyectoRepository.findById(id).orElse(null);
31     }
32
33     //Crear nuevo proyecto
34     @Transactional
35     public Proyecto createProyecto (ProyectoDto proyecto) {
36         Proyecto proyectoEntity = new Proyecto();
37         proyectoEntity.setNombre(proyecto.getNombre());
38         proyectoEntity.setDescripcion(proyecto.getDescripcion());
39
40         return proyectoRepository.save(proyectoEntity);
41     }
42
43     //Eliminar proyecto
44     @Transactional
45     public void delete(Long id) {
46         proyectoRepository.deleteByid(id);
47     }
48
49     // Actualizar proyecto existente
50     @Transactional
51     public Proyecto updateProyecto(Long id, ProyectoDto proyectoDto) {
52         Proyecto proyectoExistente = proyectoRepository.findById(id).orElse(null);
53
54         // Verificar proyecto existente
55         if (proyectoExistente != null) {

```

Imagen de elaboración propia.

Departamento.service

```

1 package com.crud_tareas.service;
2
3 import java.util.List;
4
13 @Service
14 public class DepartamentoService {
15
16     @Autowired
17     private DepartamentoRepository departamentoRepository;
18
19     //Consulta de todos los departamentos
20     @Transactional(readOnly = true)
21     public List<Departamento> findAll() {
22         return (List<Departamento>)departamentoRepository.findAll();
23     }
24
25     //Consulta por ID
26     @Transactional(readOnly = true)
27     public Departamento findById(Long id) {
28         return (Departamento) departamentoRepository.findById(id).orElse(null);
29     }
30
31     //Crear nuevo departamento
32     @Transactional
33     public Departamento createDepartamento (DepartamentoDto departamento) {
34         Departamento departamentoEntity = new Departamento();
35         departamentoEntity.setNombre(departamento.getNombre());
36
37         return departamentoRepository.save(departamentoEntity);
38     }
39
40     //Eliminar departamento
41     @Transactional
42     public void delete(Long id) {
43         departamentoRepository.deleteByid(id);
44     }
45

```

Imagen de elaboración propia.

Puesto.service

```
17 private PuestoRepository puestoRepository;
18
19 //Consulta de todos los puestos
20 @Transactional(readOnly = true)
21 public List<Puesto> findAll() {
22     return (List<Puesto>)puestoRepository.findAll();
23 }
24
25 //Consulta por ID
26 @Transactional(readOnly = true)
27 public Puesto findById(Long id) {
28     return (Puesto) puestoRepository.findById(id).orElse(null);
29 }
30
31 //Crear nuevo puesto
32 @Transactional
33 public Puesto createPuesto (PuestoDto puesto) {
34     Puesto puestoEntity = new Puesto();
35     puestoEntity.setNombre(puesto.getNombre());
36
37     return puestoRepository.save(puestoEntity);
38 }
39
40 //Eliminar puesto
41 @Transactional
42 public void delete (Long id) {
43     puestoRepository.deleteById(id);
44 }
45
46 // Actualizar puesto existente
47 @Transactional
48 public Puesto updatePuesto(Long id, PuestoDto puesto) {
49     Puesto puestoExistente = puestoRepository.findById(id).orElse(null);
50
51     // Verificar puesto existente
52     if (puestoExistente != null) {
```

Imagen de elaboración propia.

Ahora correremos nuestro proyecto dando clic derecho en nuestro proyecto, clic en "run as" y seleccionamos "Spring Boot App".

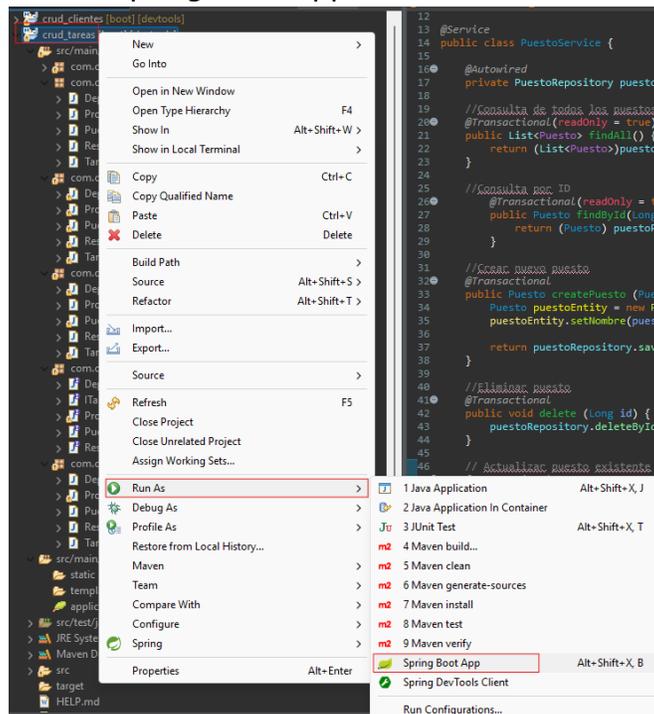


Imagen de elaboración propia.

Nos mostrará en consola como se va generando el proyecto y en caso de existir errores los mostrará como "error", de lo contrario se mostrará todo lo que se generó.

```
hibernate: select r1_0.id,p1_0.apellido_responsable,r1_0.celular,r1_0.correo,r1_0.departamento_id,r1_0.nombre_responsable,r1_0.puesto_id from responsable r1_0
hibernate: select p1_0.id,p1_0.descripcion,p1_0.nombre_proyecto from proyecto p1_0
2024-11-28T17:10:03.030-06:00 [nio-8080-exec-1] org.hibernate.SQL
: select p1_0.id,p1_0.nombre_puesto,r1_0.id,r1_0.apellido_responsable,r1_0.celular,r1_0.correo,r1_0.departamento_id,r1_0.nombre_responsable from puesto p1_0 left
: select p1_0.prioridad,t1_0.fecha_registro,t1_0.fecha_cierre,t1_0.estado,t1_0.nombre_tarea,t1_0.prioridad,t1_0.responsable_id from tarea t1_0 where t1_0
hibernate: select t1_0.proyecto_id,t1_0.id,t1_0.fecha_registro,t1_0.fecha_cierre,t1_0.estado,t1_0.nombre_tarea,t1_0.prioridad,t1_0.responsable_id from tarea t1_0 where t1_0.proyecto_id=?
2024-11-28T17:10:03.031-06:00 [nio-8080-exec-1] org.hibernate.SQL
: select p1_0.id,p1_0.nombre_puesto,r1_0.id,r1_0.apellido_responsable,r1_0.celular,r1_0.correo,r1_0.departamento_id,r1_0.nombre_responsable from puesto p1_0 left
hibernate: select p1_0.id,p1_0.nombre_puesto,r1_0.id,r1_0.apellido_responsable,r1_0.celular,r1_0.correo,r1_0.departamento_id,r1_0.nombre_responsable from puesto p1_0 left join responsable r1_0 on p1_0.id=r1_0.puesto_id where p1_0.id=?
2024-11-28T17:10:03.033-06:00 [nio-8080-exec-1] org.hibernate.SQL
: select p1_0.id,p1_0.nombre_puesto,r1_0.id,r1_0.apellido_responsable,r1_0.celular,r1_0.correo,r1_0.departamento_id,r1_0.nombre_responsable from puesto p1_0 left
hibernate: select p1_0.id,p1_0.nombre_puesto,r1_0.id,r1_0.apellido_responsable,r1_0.celular,r1_0.correo,r1_0.departamento_id,r1_0.nombre_responsable from puesto p1_0 left
: select p1_0.id,p1_0.nombre_puesto,r1_0.id,r1_0.apellido_responsable,r1_0.celular,r1_0.correo,r1_0.departamento_id,r1_0.nombre_responsable from puesto p1_0 left
2024-11-28T17:10:03.034-06:00 [nio-8080-exec-8] org.hibernate.SQL
: select p1_0.id,p1_0.nombre_puesto,r1_0.id,r1_0.apellido_responsable,r1_0.celular,r1_0.correo,r1_0.departamento_id,r1_0.nombre_responsable from puesto p1_0 left
hibernate: select p1_0.id,p1_0.nombre_puesto,r1_0.id,r1_0.apellido_responsable,r1_0.celular,r1_0.correo,r1_0.departamento_id,r1_0.nombre_responsable from puesto p1_0 left join responsable r1_0 on p1_0.id=r1_0.puesto_id where p1_0.id=?
2024-11-28T17:10:03.035-06:00 [nio-8080-exec-1] org.hibernate.SQL
: select p1_0.id,p1_0.nombre_puesto,r1_0.id,r1_0.apellido_responsable,r1_0.celular,r1_0.correo,r1_0.departamento_id,r1_0.nombre_responsable from puesto p1_0 left
hibernate: select p1_0.id,p1_0.nombre_puesto,r1_0.id,r1_0.apellido_responsable,r1_0.celular,r1_0.correo,r1_0.departamento_id,r1_0.nombre_responsable from puesto p1_0 left
: select p1_0.id,p1_0.nombre_puesto,r1_0.id,r1_0.apellido_responsable,r1_0.celular,r1_0.correo,r1_0.departamento_id,r1_0.nombre_responsable from puesto p1_0 left
2024-11-28T17:10:03.038-06:00 [nio-8080-exec-1] o.s.orm.jpa.JpaTransactionManager
: Initiating transaction commit
2024-11-28T17:10:03.038-06:00 [nio-8080-exec-1] o.s.orm.jpa.JpaTransactionManager
: Committing JPA transaction on EntityManager [SessionImpl(1831445999open)]
2024-11-28T17:10:03.041-06:00 [nio-8080-exec-8] org.hibernate.SQL
: select t1_0.proyecto_id,t1_0.id,t1_0.fecha_registro,t1_0.fecha_cierre,t1_0.estado,t1_0.nombre_tarea,t1_0.prioridad,t1_0.responsable_id from tarea t1_0 where t1_0
hibernate: select t1_0.proyecto_id,t1_0.id,t1_0.fecha_registro,t1_0.fecha_cierre,t1_0.estado,t1_0.nombre_tarea,t1_0.prioridad,t1_0.responsable_id from tarea t1_0 where t1_0.proyecto_id=?
2024-11-28T17:10:03.042-06:00 [nio-8080-exec-1] o.s.jdbc.datasource.DataSourceUtils
: Resetting read-only flag of JDBC Connection [HikariProxyConnection@172743929 wrapping org.postgresql.jdbc.PgConnection@3eb053da]
2024-11-28T17:10:03.042-06:00 [nio-8080-exec-1] o.s.orm.jpa.JpaTransactionManager
: Not closing pre-bound JPA EntityManager after transaction
2024-11-28T17:10:03.042-06:00 [nio-8080-exec-8] org.hibernate.SQL
: select t1_0.proyecto_id,t1_0.id,t1_0.fecha_registro,t1_0.fecha_cierre,t1_0.estado,t1_0.nombre_tarea,t1_0.prioridad,t1_0.responsable_id from tarea t1_0 where t1_0
hibernate: select t1_0.proyecto_id,t1_0.id,t1_0.fecha_registro,t1_0.fecha_cierre,t1_0.estado,t1_0.nombre_tarea,t1_0.prioridad,t1_0.responsable_id from tarea t1_0 where t1_0.proyecto_id=?
2024-11-28T17:10:03.043-06:00 [nio-8080-exec-8] org.hibernate.SQL
: select t1_0.proyecto_id,t1_0.id,t1_0.fecha_registro,t1_0.fecha_cierre,t1_0.estado,t1_0.nombre_tarea,t1_0.prioridad,t1_0.responsable_id from tarea t1_0 where t1_0
hibernate: select t1_0.proyecto_id,t1_0.id,t1_0.fecha_registro,t1_0.fecha_cierre,t1_0.estado,t1_0.nombre_tarea,t1_0.prioridad,t1_0.responsable_id from tarea t1_0 where t1_0.proyecto_id=?
2024-11-28T17:10:03.044-06:00 [nio-8080-exec-8] org.hibernate.SQL
: select t1_0.proyecto_id,t1_0.id,t1_0.fecha_registro,t1_0.fecha_cierre,t1_0.estado,t1_0.nombre_tarea,t1_0.prioridad,t1_0.responsable_id from tarea t1_0 where t1_0
hibernate: select t1_0.proyecto_id,t1_0.id,t1_0.fecha_registro,t1_0.fecha_cierre,t1_0.estado,t1_0.nombre_tarea,t1_0.prioridad,t1_0.responsable_id from tarea t1_0 where t1_0.proyecto_id=?
2024-11-28T17:10:03.046-06:00 [nio-8080-exec-8] o.s.orm.jpa.JpaTransactionManager
: Initiating transaction commit
2024-11-28T17:10:03.046-06:00 [nio-8080-exec-8] o.s.orm.jpa.JpaTransactionManager
: Committing JPA transaction on EntityManager [SessionImpl(1484858681open)]
2024-11-28T17:10:03.046-06:00 [nio-8080-exec-8] o.s.jdbc.datasource.DataSourceUtils
: Resetting read-only flag of JDBC Connection [HikariProxyConnection@7618660 wrapping org.postgresql.jdbc.PgConnection@41f6eb01]
2024-11-28T17:10:03.046-06:00 [nio-8080-exec-8] o.s.orm.jpa.JpaTransactionManager
: Not closing pre-bound JPA EntityManager after transaction
2024-11-28T17:10:03.052-06:00 [nio-8080-exec-8] m.a.a.RequestResponseBodyMethodProcessor
: Using 'application/json', given [application/json, text/plain, */*] and supported [application/json, application/*+json]
2024-11-28T17:10:03.052-06:00 [nio-8080-exec-1] m.a.a.RequestResponseBodyMethodProcessor
: Using 'application/json', given [application/json, text/plain, */*] and supported [application/json, application/*+json]
2024-11-28T17:10:03.056-06:00 [nio-8080-exec-8] m.a.a.RequestResponseBodyMethodProcessor
: Writing [com.crud_tareas.entity.Responsable@66235cc7, com.crud_tareas.entity.Responsable@7de85d42, com.crud_tareas.entity.Proyecto@3dfbecf51, com.crud_tareas.entity.Proyecto@7b24ec6d, com.crud_tareas (truncated)...]
2024-11-28T17:10:03.056-06:00 [nio-8080-exec-8] m.a.a.RequestResponseBodyMethodProcessor
: Writing [com.crud_tareas.entity.Proyecto@3dfbecf51, com.crud_tareas.entity.Proyecto@7b24ec6d, com.crud_tareas (truncated)...]
2024-11-28T17:10:03.058-06:00 [nio-8080-exec-0] o.j.i.OpenEntityManagerInViewInterceptor
: Closing JPA EntityManager: in OpenEntityManagerInViewInterceptor
```

Imagen de elaboración propia.

Una vez que nos corre en consola sin errores, tenemos que validar que en nuestra base de datos de PostgreSQL se hayan creado las tablas.

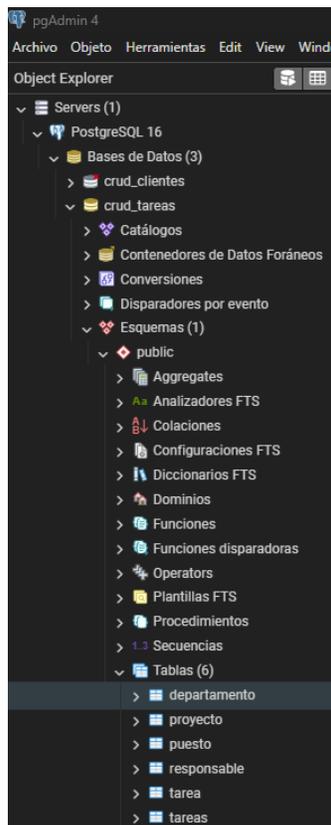


Imagen de elaboración propia.

Una vez que las tablas nos aparecen en nuestra base de datos, podremos insertar datos en cada una de las tablas, a continuación, se muestra un ejemplo del INSERT INTO en una de las tablas de la base de datos.

```
SQL x crud_tareas/postgres@PostgreSQL 16 x
crud_tareas/postgres@PostgreSQL 16
Sin límite
Consulta Historial de Consultas
1 INSERT INTO tarea (nombre_tarea, prioridad, responsable_id, estado, fecha_registro, fecha_cierre, proyecto_id)
2 VALUES
3 ('Diseño de interfaz', 'ALTA', 9, 'NUEVO', '2024-11-25', NULL, 2),
4 ('Revisión de requerimientos', 'BAJA', 8, 'EN_PROGRESO', '2024-11-20', '2024-11-22', 1),
5 ('Pruebas de funcionalidad', 'ALTA', 11, 'EN_PROGRESO', '2024-11-15', NULL, 3),
6 ('Implementación de API', 'ALTA', 12, 'CERRADO', '2024-10-01', '2024-11-10', 4),
7 ('Documentación técnica', 'BAJA', 10, 'RECHAZADO', '2024-11-18', NULL, 5);
```

Imagen de elaboración propia.

Una vez que corremos nuestro query para el INSERT de la tabla tarea realizaremos nuestra consulta de los datos guardados con SELECT * FROM tarea.

```
SQL x crud_tareas/postgres@PostgreSQL 16 x
crud_tareas/postgres@PostgreSQL 16
Sin límite
Consulta Historial de Consultas
1 SELECT * FROM tarea;
2
```

Imagen de elaboración propia.

Nos debería mostrar el siguiente resultado.

id	fecha_registro	fecha_cierre	estado	nombre_tarea	prioridad	proyecto_id	responsable_id
1	2024-11-25 00:00:00	[null]	NUEVO	Diseño de interfaz	ALTA		9
2	2024-11-20 00:00:00	2024-11-22 00:00:00	EN_PROGRESO	Revisión de requerimientos	BAJA	1	8
3	2024-11-15 00:00:00	[null]	EN_PROGRESO	Pruebas de funcionalidad	ALTA	3	11
4	2024-10-01 00:00:00	2024-11-10 00:00:00	CERRADO	Implementación de API	ALTA	4	12
5	2024-11-18 00:00:00	[null]	RECHAZADO	Documentación técnica	BAJA	5	10

Imagen de elaboración propia.

Así mismo se fueron realizando los registros de las demás tablas (responsable, proyecto, departamento y puesto).

Responsable

id	apellido	celular	como	nombre	departamento_id	puesto_id	departamento	puesto	apellido_responsable	nombre_responsable	departamento_id
1	López	5552345678	maria.lopez@empresa.com	María	8	3	[null]	[null]	López	María	[null]
2	Torres	5554567890	ana.torres@empresa.com	Ana	10	4	[null]	[null]	Torres	Ana	[null]
3	Ramírez	5555678901	lucia.ramirez@empresa.com	Lucía	11	5	[null]	[null]	Ramírez	Lucía	[null]
4	Gómez	5551234567	carlos.gomez@empresa.com	Carlos	7	1	[null]	[null]	Gómez	Carlos	[null]
5	Martínez	5553456789	pedro.martinez@empresa.com	Pedro	9	2	[null]	[null]	Martínez	Pedro	[null]

Imagen de elaboración propia.

Proyecto

	id [PK] bigint	descripcion character varying (255)	nombre character varying (255)	nombre_proyecto character varying (255)
1	1	Desarrollo de un sistema integral para gestión empresari...	Sistema de Gestión	Sistema de Gestión
2	2	Creación de una app móvil para clientes.	Aplicación Móvil	Aplicación Móvil
3	3	Planificación y ejecución de una campaña publicitaria.	Campaña Publicitaria	Campaña Publicitaria
4	4	Implementación de procesos automatizados en finanzas.	Automatización Financiera	Automatización Financiera
5	5	Mejora de procesos logísticos y de producción.	Optimización de Operaciones	Optimización de Operaciones

Imagen de elaboración propia.

Departamento

	id [PK] bigint	nombre character varying (255)	nombre_departamento character varying (255)
1	7	Recursos Humanos	RH
2	8	Tecnología de la Información	TI
3	9	Marketing	MKT
4	10	Finanzas	FNZ
5	11	Operaciones	OPS

Imagen de elaboración propia.

Puesto

	id [PK] bigint	nombre character varying (255)	nombre_puesto character varying (255)
1	1	Gerencia	Gerente
2	2	Análisis	Analista
3	3	Desarrollo	Desarrollador
4	4	Diseño	Diseñador
5	5	Soporte	Soporte Técnico

Imagen de elaboración propia.

Como ya funciona nuestra base de datos, resta realizar nuestras pruebas en **postman** para validar que está tomando la base de datos en las rutas designadas en Spring, y así mismo en el puerto que indicamos. Realizaremos nuestra búsqueda con nuestro método GET y colocamos la ruta localhost en el puerto 8080/tareas.

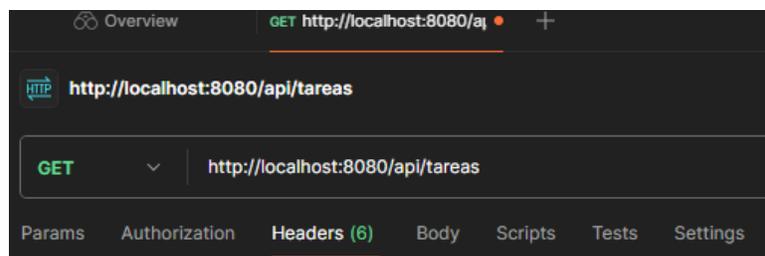


Imagen de elaboración propia.

Una vez que ingresamos la ruta de tareas, damos clic en SEND para visualizar la consulta.

```

1  [
2    {
3      "id": 1,
4      "nombre": "Diseño de interfaz",
5      "prioridad": "ALTA",
6      "estado": "NUEVO",
7      "createAt": "2024-11-25T00:00:00",
8      "createAtc": null,
9      "responsable": {
10       "id": 9,
11       "nombre": "María",
12       "apellido": "López",
13       "correo": "maria.lopez@empresa.com",
14       "celular": "5552345678",
15       "departamento": {
16         "id": 8,
17         "nombre": "TI"
18       },
19       "puesto": {
20         "id": 3,
21         "nombre": "Desarrollador"
22       }
23     },
24     "proyecto": {
25       "id": 2,
26       "nombre": "Aplicación Móvil",
27       "descripcion": "Creación de una app móvil para clientes."
28     }
29   },
30   {
31     "id": 2,
32     "nombre": "Revisión de requerimientos".

```

Imagen de elaboración propia.

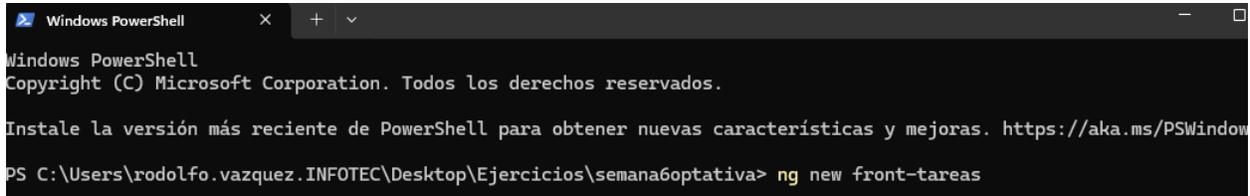
De esta manera realizaremos las pruebas para las otras entidades (responsable, proyecto, departamento y puesto). Así mismo haciendo uso de los métodos Post, Put y Delete.

Para continuar, ahora se mostrará lo realizado para la parte del frontend con el IDE Visual Studio Code y los frameworks Angular, Bootstrap, así como el uso de HTML.

Abriremos la carpeta donde guardaremos nuestro proyecto y abriremos la consola ya sea de powershell, o la misma que nos proporciona el IDE VSC, y crearemos nuestro proyecto en Angular.

NOTA: Anteriormente ya debimos haber hecho la instalación de la versión 16 de Angular y la correspondiente soportada para Bootsatrap.

Una vez que nos abrió la terminal en la ruta de la carpeta donde guardaremos el proyecto, vamos a crear nuestro proyecto con el comando “**ng new front-tareas**”, es decir, el comando **ng new** de angular y en seguida el nombre que le vamos a poner al proyecto.



```
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Instale la versión más reciente de PowerShell para obtener nuevas características y mejoras. https://aka.ms/PSWindows

PS C:\Users\rodolfo.vazquez.INFOTEC\Desktop\Ejercicios\semana6optativa> ng new front-tareas
```

Imagen de elaboración propia.

Una vez que creamos el proyecto, de una vez crearemos también nuestro primer componente **tarea** con el comando “ng generate component components/tarea”.



```
PS C:\Users\rodolfo.vazquez.INFOTEC\Desktop\Ejercicios\semana6optativa\front-tareas> ng generate component components/tarea
```

Imagen de elaboración propia.

Una vez que se crea el proyecto Angular, podemos correr el servidor con el comando “**ng serve -o**” para asegurarnos de que el proyecto fue creado correctamente.



```
PS C:\Users\rodolfo.vazquez.INFOTEC\Desktop\Ejercicios\semana6optativa\front-tareas> ng serve -o
```

Imagen de elaboración propia.

También podemos crear nuestras carpetas para los componentes y nuestros archivos desde el IDE, dando clic derecho sobre nuestro proyecto.

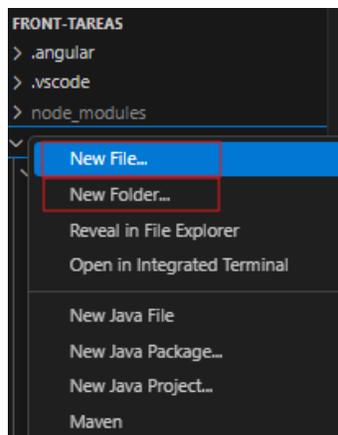


Imagen de elaboración propia.

Solo tenemos que asegurarnos que clase de archivo vamos a crear, para archivos HTML, con la extensión “html”, para archivos de estilos con la extensión “css”, y para los componentes y servicios con la extensión de TypeScript “ts”.

Como vamos a trabajar con otras 4 entidades aparte de la entidad "tarea", crearemos los componentes y servicios de cada una del resto de las entidades (responsable, proyecto, departamento y puesto). De tal manera que nuestro explorador de archivos de VSC se deberá ver de la siguiente manera:

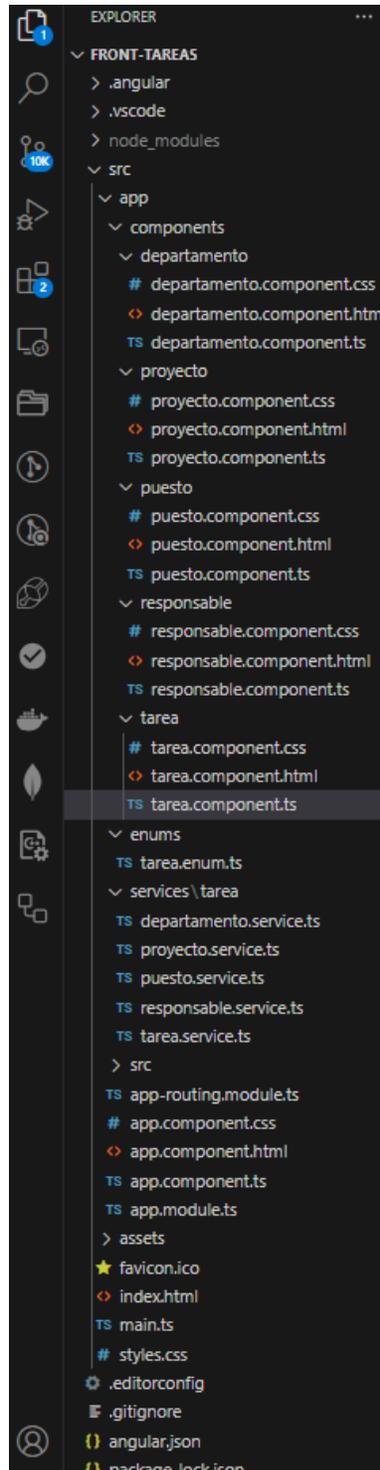


Imagen de elaboración propia.

De esta manera es como empezaremos a trabajar nuestro código dentro de nuestros archivos.

Con los archivos que inicialmente vamos a trabajar es con los "app.component", los cuales se generan automáticamente cuando creamos el proyecto desde la consola, pero antes de ello vamos a insertar nuestro enlace para poder hacer uso de **Bootstrap** mediante una etiqueta "link".

```
src > index.html > html > body > app-root
1 <!doctype html>
2 <html lang="es">
3 <head>
4   <meta charset="utf-8">
5   <title>Front-Tareas</title>
6   <base href="/">
7   <meta name="viewport" content="width=device-width, initial-scale=1">
8   <link rel="icon" type="image/x-icon" href="favicon.ico">
9   <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">
10 </head>
11 <body>
12   <app-root>
13     <app-responsible></app-responsible>
14   </app-root>
15 </body>
16 </html>
```

Imagen de elaboración propia.

En nuestro archivo app.component iniciamos dándole un poco de forma a lo que va a ser nuestra página web, integrando una barra de navegación nuestro contenedor que vamos a usar como el body, y nuestro pie de página.

```
src > app > app.component.html > footer.bg-dark.text-center.text-white.py-3
Go to component
1 <!-- Navbar -->
2 <nav class="navbar navbar-expand-lg navbar-success bg-success text-white">
3   <div class="container">
4     <div class="navbar-nav mx-auto">
5       <a class="nav-item nav-link text-white" routerLink="/tarea">Tareas</a>
6       <a class="nav-item nav-link text-white" routerLink="/responsable">Responsables</a>
7       <a class="nav-item nav-link text-white" routerLink="/proyecto">Proyectos</a>
8       <a class="nav-item nav-link text-white" routerLink="/departamento">Departamentos</a>
9       <a class="nav-item nav-link text-white" routerLink="/puesto">Puestos</a>
10    </div>
11  </div>
12 </nav>
13
14 <div class="container my-4">
15   <!-- Contenido Dinámico de las Entidades -->
16   <router-outlet></router-outlet>
17 </div>
18
19 <!-- Footer -->
20 <footer class="bg-dark text-center text-white py-3">
21   <p>CRUD de Entidades © 2024</p>
22 </footer>
```

Imagen de elaboración propia.

Seguimos con un poco de estilos generales de nuestro proyecto agregando un fondo de gradiente.

```
src > # styles.css > body
1 body{
2   background: #2980B9; /* fallback for old browsers */
3   background: -webkit-linear-gradient(to right, #FFFFFF, #60D5FA, #2980B9); /* Chrome 10-25, Safari 5.1-6 */
4   background: linear-gradient(to right, #FFFFFF, #60D5FA, #2980B9); /* W3C, IE 10+/ Edge, Firefox 16+, Chrome 26+, Opera 12+, Safari 7+ */
5 }
6 }
```

Imagen de elaboración propia.

Continuamos generando nuestro constructor para el formulario y tarea service, así como nuestras variables y nuestros métodos que vamos a utilizar para que consuma los datos para crear las consultas y las demás acciones del CRUD.

```
src > app > TS app.component.ts > ...
1 import { Component, OnInit } from '@angular/core';
2 import { FormGroup, FormBuilder, Validators } from '@angular/forms';
3 import { Prioridad, Estado } from './enums/tarea.enum'; // Importar los ENUMs
4 import { TareaService } from './services/tarea/tarea.service';
5
6 @Component({
7   selector: 'app-root',
8   templateUrl: './app.component.html',
9   styleUrls: ['./app.component.css']
10 })
11 export class AppComponent implements OnInit{
12
13   tareaForm!: FormGroup;
14   prioridades = Object.values(Prioridad); // Cargar las opciones del ENUM
15   estados = Object.values(Estado); // Cargar las opciones del ENUM
16   tarea: any[] = [];
17   filteredTarea: any[] = [];
18   tareaSeleccionada: any = null;
19
20
21   constructor(
22     public fb: FormBuilder,
23     public tareaService: TareaService){ }
24
25   ngOnInit(): void {
26
27     this.tareaForm = this.fb.group({
28
29       nombre: ['', Validators.required],
30       prioridad: ['', Validators.required],
31       responsableId: ['', Validators.required],
32       estado: ['', Validators.required],
33       fechaRegistro: ['', Validators.required],
34       fechaCierre: ['', Validators.required],
35       proyectoId: ['', Validators.required]
36     });
37
38     this.cargarTareas();
39   }
40
41   cargarTareas(): void {
42     this.tareaService.getAllTarea().subscribe(resp => {
43       this.tarea = resp;
44       this.filteredTarea = this.tarea;
45     });
46 }
```

Imagen de elaboración propia.

Nos aseguramos que nuestros archivos estén en ruta adecuadamente en nuestro archivo app-routing.module.ts

```
src > app > TS app-routing.module.ts > ...
1  import { NgModule } from '@angular/core';
2  import { RouterModule, Routes } from '@angular/router';
3  import { TareaComponent } from './components/tarea/tarea.component';
4  import { ResponsableComponent } from './components/responsable/responsable.component';
5  import { ProyectoComponent } from './components/proyecto/proyecto.component';
6  import { DepartamentoComponent } from './components/departamento/departamento.component';
7  import { PuestoComponent } from './components/puesto/puesto.component';
8
9  const routes: Routes = [
10   { path: 'tarea', component: TareaComponent },
11   { path: 'responsable', component: ResponsableComponent },
12   { path: 'proyecto', component: ProyectoComponent },
13   { path: 'departamento', component: DepartamentoComponent },
14   { path: 'puesto', component: PuestoComponent },
15   { path: '**', redirectTo: 'tarea' }
16 ];
17
18
19
20 @NgModule({
21   imports: [RouterModule.forRoot(routes)],
22   exports: [RouterModule]
23 })
24 export class AppRoutingModule { }
```

Imagen de elaboración propia.

De la misma manera nos aseguramos de que estemos importando nuestros archivos correctamente en nuestro archivo app.module.ts y así mismo se lo estemos exportando a nuestro AppModule.

```
src > app > TS app.module.ts > AppModule
4  import { AppRoutingModule } from './app-routing.module';
5  import { AppComponent } from './app.component';
6  import { ResponsableComponent } from './components/responsable/responsable.component';
7  import { ProyectoComponent } from './components/proyecto/proyecto.component';
8  import { DepartamentoComponent } from './components/departamento/departamento.component';
9  import { PuestoComponent } from './components/puesto/puesto.component';
10 import { TareaComponent } from './components/tarea/tarea.component';
11
12 import { FormsModule } from '@angular/forms';
13 import { ReactiveFormsModule } from '@angular/forms';
14 import { HttpClientModule } from '@angular/common/http';
15 import { TareaService } from './services/tarea/tarea.service';
16 import { ResponsableService } from './services/tarea/responsable.service';
17 import { ProyectoService } from './services/tarea/proyecto.service';
18 import { DepartamentoService } from './services/tarea/departamento.service';
19 import { PuestoService } from './services/tarea/puesto.service';
20
21
22 @NgModule({
23   declarations: [
24     AppComponent,
25     TareaComponent,
26     ResponsableComponent,
27     ProyectoComponent,
28     DepartamentoComponent,
29     PuestoComponent,
30   ],
31 },
32
33 imports: [
34   BrowserModule,
35   AppRoutingModule,
36   FormsModule,
37   ReactiveFormsModule,
38   HttpClientModule,
39 ],
40
41 providers: [TareaService,
42             ResponsableService,
43             ProyectoService,
44             DepartamentoService,
45             PuestoService
46 ],
47 bootstrap: [AppComponent, ResponsableComponent]
48 })
49 export class AppModule { }
```

Imagen de elaboración propia.

Continuamos creando la parte visual con html y bootstrap, como nuestros formularios, campos de texto, labels, botones de las entidades que estaremos usando.

```
src > app > components > tarea > tarea.component.html > div.container > form.mb-4 > div.row > div.col-md-6.mb-3
Go to component
1 <div class="container">
2 <h2 class="text-center my-3">Gestión de Tareas</h2>
3
4 <!-- Formulario -->
5 <form [formGroup]="tareaForm" (ngSubmit)="guardar()" class="mb-4">
6 <div class="row">
7
8 <!-- Nombre -->
9 <div class="col-md-6 mb-3">
10 <label for="nombre" class="form-label">Nombre</label>
11 <input id="nombre" type="text" class="form-control" formControlName="nombre">
12 <div *ngIf="tareaForm.get('nombre')?.invalid && tareaForm.get('nombre')?.touched" class="text-danger">
13 El nombre es obligatorio.
14 </div>
15 </div>
16
17 <!-- Prioridad -->
18 <div class="col-md-6 mb-3">
19 <label for="prioridad" class="form-label">Prioridad</label>
20 <select id="prioridad" class="form-select" formControlName="prioridad">
21 <option value="">Seleccione</option>
22 <option *ngFor="let prioridad of prioridades" [value]="prioridad">
23 {{ prioridad }}
24 </option>
25 </select>
26 <div *ngIf="tareaForm.get('prioridad')?.invalid && tareaForm.get('prioridad')?.touched" class="text-danger">
27 La prioridad es obligatoria.
28 </div>
29 </div>
30
31 <!-- Responsable -->
32 <div class="col-md-6 mb-3">
33 <label for="responsableId" class="form-label">Responsable</label>
34 <select id="responsableId" class="form-select" formControlName="responsableId">
35 <option value="">Seleccione</option>
36 <option *ngFor="let responsable of responsables" [value]="responsable.id">
37 {{ responsable.nombre }} {{ responsable.apellido }}
38 </option>
39 </select>
40 <div *ngIf="tareaForm.get('responsableId')?.invalid && tareaForm.get('responsableId')?.touched" class="text-danger">
41 El responsable es obligatorio.
42 </div>
43 </div>
44
45 <!-- Estado -->
46 <div class="col-md-6 mb-3">
```

Imagen de elaboración propia.

De esta misma manera vamos a estar generando nuestro código de las entidades de los archivos responsable.component.html, proyecto.component.html, departamento.component.html y puesto.component.html

Ahora vamos a generar nuestra parte lógica y funcional en nuestro archivo `tarea.component.ts` generando nuestros métodos para cargar tareas, responsables, proyectos, guardar y eliminar tareas.

```
src > app > components > tarea > TS tarea.component.ts > TareaComponent > ngOnInit
/
8  @Component({
9  selector: 'app-tarea',
10  templateUrl: './tarea.component.html',
11  styleUrls: ['./tarea.component.css']
12  })
13  export class TareaComponent implements OnInit {
14
15  tareaForm!: FormGroup;
16  tareas: any[] = [];
17  responsables: any[] = []; // Definir responsables
18  proyectos: any[] = []; // Definir proyectos
19  tareaSeleccionada: any = null;
20
21  prioridades = Object.values(Prioridad); // Array dinámico de prioridades
22  estados = Object.values(Estado); // Array dinámico de estados
23
24  constructor(
25    private fb: FormBuilder,
26    private tareaService: TareaService,
27    private responsableService: ResponsableService,
28    private proyectoService: ProyectoService
29  ) { }
30
31  ngOnInit(): void {
32    this.tareaForm = this.fb.group({
33      nombre: ['', Validators.required],
34      prioridad: ['', Validators.required],
35      responsableId: ['', Validators.required],
36      estado: ['', Validators.required],
37      fechaRegistro: ['', Validators.required],
38      fechaCierre: ['', Validators.required],
39      proyectoId: ['', Validators.required]
40    });
41
42    this.cargarTareas();
43    this.cargarResponsables();
44    this.cargarProyectos();
45  }
46
47  cargarTareas(): void {
48    this.tareaService.getAllTarea().subscribe(
49      resp => this.tareas = resp,
50      error => console.error(error)
51    );
52  }
53 }
```

Imagen de elaboración propia.

De esta misma manera vamos a estar generando nuestro código de las entidades de los archivos `responsable.component.ts`, `proyecto.component.ts`, `departamento.component.ts` y `puesto.component.ts`

Seguido de nuestro deberemos tener las siguientes vistas en el navegador.

Tarea



Gestión de Tareas

Nombre:

Prioridad:

Responsable:

Estado:

Fecha de Registro:

Fecha de Cierre:

Proyecto:

Listado de Tareas

ID	Nombre	Prioridad	Responsable	Estado	Fecha Registro	Fecha Cierre	Proyecto	Acciones
----	--------	-----------	-------------	--------	----------------	--------------	----------	----------

CRUD de Entidades © 2024

Imagen de elaboración propia.

Responsable



Gestión de Responsables

Nombre:

Apellido:

Correo:

Celular:

Departamento:

Puesto:

Listado de Responsables

ID	Nombre	Apellido	Correo	Celular	Departamento	Puesto	Acciones
9	María	López	maria.lopez@empresa.com	5552345678			<input type="button" value="Editar"/> <input type="button" value="Eliminar"/>
11	Ana	Torres	ana.torres@empresa.com	5554567890			<input type="button" value="Editar"/> <input type="button" value="Eliminar"/>
12	Lucía	Ramírez	lucia.ramirez@empresa.com	5555678901			<input type="button" value="Editar"/> <input type="button" value="Eliminar"/>
8	Carlos	Gómez	carlos.gomez@empresa.com	5551234567			<input type="button" value="Editar"/> <input type="button" value="Eliminar"/>

CRUD de Entidades © 2024

Imagen de elaboración propia.

Proyecto

Gestión de Proyectos

Nombre:

Descripción:

[Guardar](#) [Limpiar](#)

Listado de Proyectos

ID	Nombre	Descripción	Acciones
1	Sistema de Gestión	Desarrollo de un sistema integral para gestión empresarial.	Editar Eliminar
2	Aplicación Móvil	Creación de una app móvil para clientes.	Editar Eliminar
3	Campaña Publicitaria	Planificación y ejecución de una campaña publicitaria.	Editar Eliminar
4	Automatización Financiera	Implementación de procesos automatizados en finanzas.	Editar Eliminar
5	Optimización de Operaciones	Mejora de procesos logísticos y de producción.	Editar Eliminar

Imagen de elaboración propia.

Departamento

Gestión de Departamentos

Nombre del Departamento:

[Guardar](#) [Limpiar](#)

Listado de Departamentos

ID	Nombre	Acciones
7	RH	Editar Eliminar
8	TI	Editar Eliminar
9	MKT	Editar Eliminar
10	FNZ	Editar Eliminar
11	OPS	Editar Eliminar

Imagen de elaboración propia.

Puesto



Imagen de elaboración propia.

A continuación, se muestran nuestras relaciones que utilizamos para el gestor de tareas:

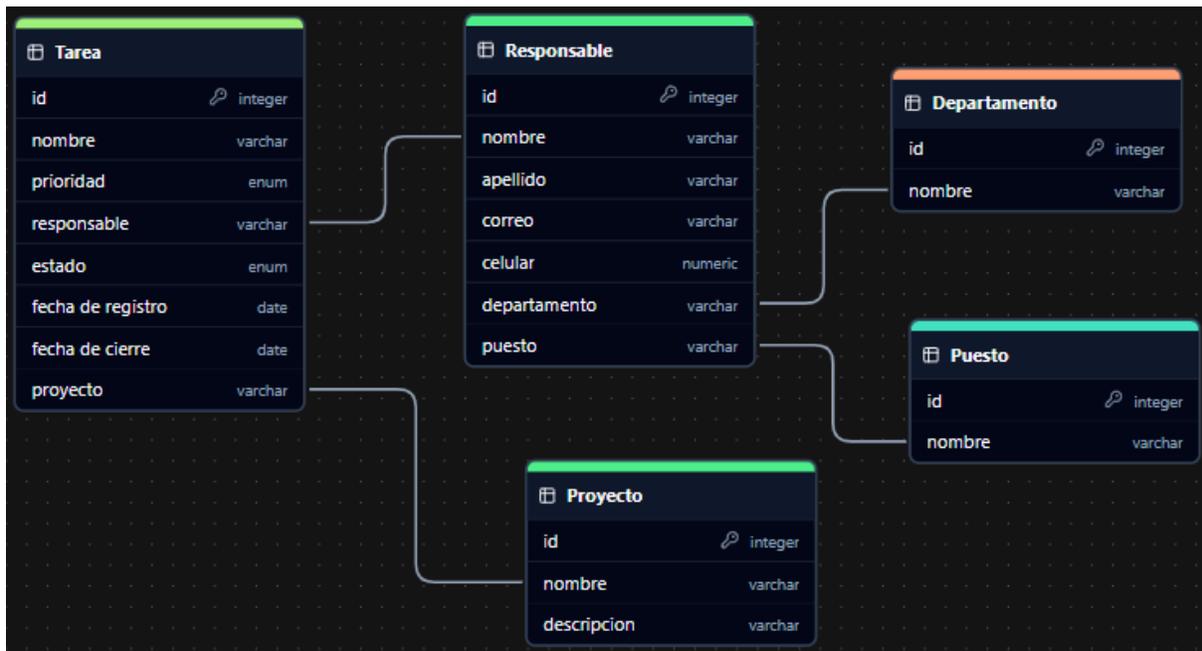


Imagen de elaboración propia.

Práctica profesional

Datos personales del estudiante			
Nombre completo:	Rodolfo Gael Vazquez Ubaldo		
Matrícula:	Eliminado	Teléfono:	Eliminado
Correo:	Eliminado		
Especialidad de TSU:	Desarrollo de Software de Código Abierto		
Año:	2024	Celular:	Eliminado
Número de horas de práctica profesional:	280 horas		

Datos de la institución	
Nombre de la institución:	Centro de Investigación e Innovación en Tecnologías de la Información y Comunicación (INFOTEC)
Departamento en el que realizarás tus prácticas:	Dirección Adjunta de Administración de Proyectos - Desarrollo de Software
Nombre de la persona responsable del departamento:	Claudio Moran Ponce

Estructura organizacional

- **El sector de actividad**

El INFOTEC, tiene contribución en la Transformación Digital para todo México, por medio de investigación, innovación, educación o formación académica y ofrece productos como desarrollo y servicios de tecnologías de la información. Sus alcances abarcan al sector público y privado, habilitando caminos que conduzcan hacia un México moderno y de inclusión digital.

• **Presentación de la empresa e historia empresarial**

INFOTEC es una institución con más de 40 años de vida que, desde sus orígenes en 1974, se ha dedicado a la distribución del conocimiento; primero, en materia de innovación en la industria, proyectos específicos sobre las TIC's, con la intención de incrementar los avances en México.

En la actualidad, se cuenta con un par de sedes, una en la CDMX y, a partir del 2013 la segunda en Aguascalientes. Con respecto al portafolio de soluciones, las principales líneas de negocio incluyen desarrollo de aplicaciones, IoT (internet de las cosas), tecnología en infraestructura, así como docencia y temas de investigación.

Nuestros servicios de desarrollo están acreditados con la certificación CMMI nivel 5 y contamos con un Centro de Datos en la ciudad de Aguascalientes, reconocido como el primer data center del sector público mexicano certificado internacionalmente como Tier III por el Uptime Institute.

Nuestra oferta académica incluye programas especializados de maestría y doctorado en dirección estratégica, gestión de innovación, y derecho de las TIC, así como en ciencia de datos y sistemas embebidos, entre otras.

La investigación que se desarrolla en INFOTEC tiene por finalidad la generación de conocimiento y su aplicación en el campo de las TIC para el desarrollo de productos y servicios tecnológicos, así como de modelos y metodologías destinados al crecimiento nacional.

Por esta razón, contamos con diversos ámbitos de conocimiento, los cuales representan tendencias en TIC a nivel global y derivan en proyectos de investigación que inciden de manera directa en el desarrollo de sectores clave del país.

La organización de la empresa (incluir organigrama)

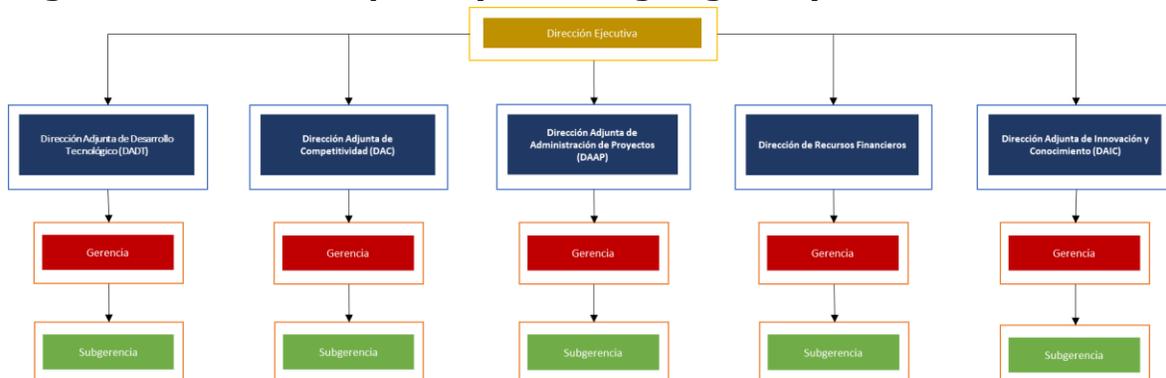


Imagen de elaboración propia.

Informe de la práctica profesional

- **Descripción de las funciones**

- Dailys semanales con Scrum Master y Líder Técnico, revisión de material de apoyo semanal y optativo, realización de actividades por semana.
- Revisión semanal de recursos de apoyo y aplicación de lo revisado en visual Studio Code (directivas, formularios basados en plantillas, formularios reactivos, pipes, tipado estático, clases, funciones, métodos, interfaces)
- Revisión semanal de recursos de apoyo y aplicación de lo revisado dentro del proyecto (generación de bases de datos con Spring, CRUD, JPA).
- Inserciones de datos a las tablas generadas dentro de la base de datos (INSERT INTO) y consultas de los datos de las entidades (SELECT *FROM "entidad").
- Validaciones de los métodos de consultas, creación, actualización y eliminación de los datos.
- Implementación de relación entre entidades en Spring (FetchType LAZY y EAGER).
- Generación de formularios con CRUD únicamente con Angular y arrays.
- Avances de semanales de la conclusión del proyecto "Gestor de tareas".

- **Desglose de actividades**

- Creación de página web con HTML, Bootstrap + CSS + JS
- Instalación de ambiente de trabajo para Angular
- Instalación de Spring Tool Suite
- Aplicación "presupuesto" – Uso de observables

- Creación de página web con HTML, Bootstrap + CSS + JS
- Instalación de ambiente de trabajo para Angular
- Instalación de Spring Tool Suite
- Aplicación "presupuesto" – Uso de observables

- Se realizó ejercicio práctico acerca de formularios reactivos en donde conocimos una mejor funcionalidad de estos formularios y una manera distinta de aplicar un required de los campos de un formulario – 30 de septiembre al 03 de oct.

- Se realizó una API REST con conexión a Postgresql/MySQL (Personalizado), es decir, retomamos la primera conexión de la Base de Datos que habíamos realizado anteriormente y se realizaron implementaciones adicionales y/o diferentes a las que se hicieron inicialmente- 07 al 13 de oct.
- Realizamos una Aplicación CRUD en Angular en donde al guardamos la información capturada mediante arreglos para poder consultarla, editarla y eliminarla- 07 al 20 de oct.
- Se revisó el material de "Relación entre entidades" con el objetivo de realizar implementaciones de relación a nuestro CRUD y dar avances al proyecto final – 21 al 27 de oct.
- Se va a llevar a cabo durante esta semana el "Cuestionario teórico" con conceptos y temas que se han visto y aprendido durante la estancia, y ver el "Material complementario "curso de Angular" – 21 de oct al 03 de nov.

Aplicación Web Spring Boot y Angular – 28 de oct al 03 de nov.

- **Bitácora de prácticas**

Semanas	Actividades realizadas	Observaciones
1 a 2	<ul style="list-style-type: none"> • En el transcurso de la semana se han revisado los materiales de apoyo y se realizan ejercicios prácticos con base al material revisado, así mismo, con lo visto se llevan a cabo las actividades que se entregan al final de la semana, en este caso lo que se ha visto en los materiales de apoyo es HTML, Bootstrap, la instalación de los ambientes de Angular y Spring, y TypeScript. 	
3 a 4	<ul style="list-style-type: none"> • Introducción a Angular – 16 al 22 de septiembre ("¿Qué es Angular-Herramientas Necesarias?", "Creación de un proyecto, archivos y carpetas básicas", "Directivas y capturas de eventos", "Componentes", "Módulos: Creación y consumo", "Petición de un archivo JSON a un servidor", "Instalación de ambiente de trabajo para Angular"). • Ejercicio: Instalación de Spring Tool 	

	<p>Suite – 16 al 22 de sep.</p> <ul style="list-style-type: none"> • TypeScript – 23 al 29 de septiembre (“Router: definición de rutas”, “Servicios: concepto y pasos para crearlos”, “Servicios: recuperación de datos de un servidor web”, “Pipes: definición”, “Pipes: creación de pipes personalizadas”, “TypeScript”, “TypeScript: tipado estático”, “TypeScript: clases”, “TypeScript: funciones y métodos”, “TypeScript: herencia”, “TypeScript: interfaces”, “TypeScript: clases genéricas”). 	
5 a 6	<ul style="list-style-type: none"> • Ejercicio: Aplicación presupuesto_ uso de observables– 23 al 29 de sep. • Spring Boot – 23 de septiembre al 03 de oct (“Curso Spring Boot CRUD + JPA + Postgresql”, “Clase Spring - BDD”, “Ejercicio: API REST con conexión a Postgresql/MySQL”). • Ejercicio: Formularios reactivos – 30 de septiembre al 03 de oct. • Ejercicio: API REST con conexión a Postgresql/MySQL (Personalizado) - 07 al 13 de oct. • Aplicación CRUD en Angular - 07 al 20 de oct. 	
7 a 8	<ul style="list-style-type: none"> • Relación entre entidades – 21 al 27 de oct. • Cuestionario teórico, Material complementario “curso de Angular” – 21 de oct al 03 de nov. • Aplicación Web Spring Boot y Angular – 28 de oct al 03 de nov. 	
9 a 10	<ul style="list-style-type: none"> • Semana 9 – Avance Proyecto “Gestor de tareas” – 28 de oct al 28 de nov. • Semana 10 – Avance Proyecto “Gestor de tareas” – 28 de oct al 28 de nov. 	

11 a 12	<ul style="list-style-type: none"> • Semana 11 – Avance Proyecto “Gestor de tareas” – 28 de oct al 28 de nov. • Ejercicio: Integración Spring Boot - Angular – 28 de oct al 28 de nov. 	
---------	--	--

Conclusiones

De manera general puedo decir que en lo personal fue una gran experiencia el haber tomado las prácticas de este TSU en desarrollo de software de código abierto, pues to que lo realizado fue algo que supero mis expectativas vs lo que se vio en el transcurso de los cuatrimestres a lo largo del curso, ya que nos enfocamos a algo en específico, haciendo uso de herramientas que no tuvimos oportunidad de conocer en las materias vistas y lo que considero mejor, hicimos uso de lenguajes específicos y frameworks durante todas las prácticas, lo cual me permitió enfocar la atención a un solo objetivo desarrollando conocimientos con herramientas y lenguajes populares pero robustos dentro de la industria del desarrollo de software.

Claro, no es de más mencionar que todo esto conlleva complicaciones en el transcurso del aprendizaje y sobretodo en la parte de la implementación, definitivamente no es lo mismo la teoría que la parte práctica, y esto me lleva a concluir que ¿el desarrollo de software es algo complejo de aprender y llevar a cabo?, puedo decir que si, ¿es algo imposible?, para nada imposible, pero como todo en lo absoluto, mientras más se practica se vuelve más fácil de entender y así mismo de llevar a cabo, me parece que la parte más satisfactoria es ver lo que lograste al final visualmente y funcionalmente hablando, después de arduas jornadas de código en donde seguramente vi muchos errores y la mayoría costaron trabajo resolverlos, pero llegar a ese resultado final, es lo que emociona y crea las ganas de continuar en este ámbito.

Cuadro CQA de mi estancia en la organización

C ¿Qué C onozco?	Q ¿Qué Q u aporte?	A ¿Qué A prendí?
Lo que conozco al inicio de estas prácticas era un poco sobre las bases de datos relacionales, el lenguaje html y un poco sobre java script, realmente un	Realmente no puedo decir que aporte conocimientos, ya que estos fueron los que realmente quien nos instruía en las prácticas era quien nos despejaba	Aprendí muchísimo en estas prácticas, como a usar TypeScript que está basado en JavaScript, Bootstrap, Angular y la generación de proyectos,

<p>poquito más de lo básico, pero sabía que eso no sería suficiente para lo que veía venir en el proyecto.</p>	<p>de las dudas y nos apoyaba cuando teníamos complicaciones de cualquier tipo, pero lo que puedo decir, es que aporte mis ganas de cumplir este objetivo final y tomar lo mejor de las prácticas para llevarlo a un nivel más profesional, y que a pesar de las complicaciones personales que pude llegar a tener, no me quitaron el compromiso de intentar terminal de la mejor manera toda esta parte práctica del TSU y de la estancia.</p>	<p>componentes y servicios, tenía nociones de SQL pero aprendía a usar al menos de manera básica PostgreSQL, Spring haciendo uso de Java y realizando la conexión con la base de datos, a realizar la relación entre entidades, a generar las pruebas de lo anteriormente mencionado con Postman, en donde no iba a funcionar el GET, POST, PUT o DELETE si no había funcionado algo de lo anteriormente mencionado, y aprendí muchas más funcionalidades de html conto con el framework Bootstrap.</p>
--	---	---

Evaluación de desempeño

Considero que mi evaluación de desempeño en la estancia dentro de la organización ha sido bastante buena, si bien no era de los mejores de la clase, me parece que siempre fui constante en todo lo que se pedía, como anteriormente comentaba, tuve situaciones personales que no me permitieron avanzar al ritmo que me hubiera gustado, siempre he hecho el esfuerzo por realizar todas las entregas en tiempo y forma y con la mejor calidad posible, y yo creo que en eso me doy más valor, porque siempre he sido una persona perseverante y que siempre y bajo la situación que sea intenta lograr sus objetivos, he tenido algunas lecciones de vida que en sus momento me complicaron el avance en mi vida profesional, y parte de lo que he aprendido es que nada es gratis y todo logro conlleva un esfuerzo a veces de más del 100%, pero que nada es imposible cuando quieres crecer y ser una persona más preparada para el día a día.

Referencias

- INFOTEC (s/f). "Trayectoria con más de cuatro décadas de experiencia". <https://www.infotec.mx/Infotec>

- Referencia desde el punto de vista y conocimiento personal de Rodolfo Gael Vazquez Ubaldo.
- TSU de "INFOTEC" (2023 - 2024). "Desarrollo de Software de Código Abierto".
- TSU de "INFOTEC" (2023 - 2024). "Estancias Profesionales - Desarrollo de Software de Código Abierto".
- INFOTEC (2024). "¿Qué es INFOTEC?". <https://www.infotec.mx/Infotec>
- Bootcamp INFOTEC (2024). "¿Qué es HTML y para qué sirve?". <https://www.youtube.com/watch?v=eN39GOWCWzQ>
- Bootcamp INFOTEC (2024). "¿Qué es una etiqueta en HTML5 y cómo funciona?". <https://youtu.be/q8ibtKwbt6c>
- Victor robles web (2020). "Aprende HTML5 en 15 minutos?". <https://www.youtube.com/watch?v=mNbnV3aN3KA>
- Bootcamp INFOTEC (2024). "¿Qué es CSS?". <https://youtu.be/pSM43Sj5vlq>
- Bootcamp INFOTEC (2024). "Aplicar estilos usando propiedades de CSS". <https://youtu.be/idn1ybWFaEM>
- Bootcamp INFOTEC (2024). "¿Qué es un framework?". <https://youtu.be/J1hmAIrEya4>
- Bootcamp INFOTEC (2024). "Funcionalidades de Bootstrap". https://youtu.be/iyU_g6DL5dU
- Dostin Hurtado (2023). Creación de página web con HTML, Bootstrap + CSS + JS". <https://youtu.be/-ZInT4p9hH8>
- Bootcamp INFOTEC (2024). Creación de un proyecto, archivos y carpetas básicas en Angular". <https://youtu.be/yMqCrv-631w>
- Bootcamp INFOTEC (2024). "Directivas y capturas de eventos". <https://youtu.be/Z5UJn95z8p4>
- Bootcamp INFOTEC (2024). "Componentes". <https://youtu.be/QZDwe7jAOU>
- Bootcamp INFOTEC (2024). "Módulos, creación y consumo". <https://youtu.be/ySXPF78hJQ>

- Electronikoin Team (2023). "Instalación de ambiente de trabajo para Angular". <https://youtu.be/Z6kiJkqiCFI>
- Cristian Ballesteros (2023). "Instalación de Spring Tool Suite".
- Bootcamp INFOTEC (2024). "Servicios, conceptos y pasos para crearlos". https://youtu.be/bf2_sIDsl8
- Bootcamp INFOTEC (2024). "Pipes: definición". <https://youtu.be/9BStgKzeKY0>
- Bootcamp INFOTEC (2024). "Pipes: creación de pipes personalizadas". <https://youtu.be/AM063v6DTk8>
- Bootcamp INFOTEC (2024). "TypeScript". <https://youtu.be/NQwedBch0YM>
- Bootcamp INFOTEC (2024). "TypeScript: tipado estático". <https://youtu.be/mXCnVbITRFI>
- Bootcamp INFOTEC (2024). "TypeScript: clases". https://youtu.be/44uSc-S_7xQ
- Bootcamp INFOTEC (2024). "TypeScript: Funciones y métodos". <https://youtu.be/FdhgBhEJBYS>
- Bootcamp INFOTEC (2024). "TypeScript: herencia". <https://youtu.be/Kb-1bIG124I>
- Tomas Ruíz Díaz (2021). "Ejercicio: Aplicación presupuesto - uso de observables". <https://youtu.be/JsNMnXD-NRM>
- Yo Androide (2020). "Curso Spring Boot Crud + jpa + Postgresql". <https://youtu.be/liGDWn55fxw>
- Coders Free (2023). "Formularios reactivos". <https://www.youtube.com/playlist?list=PLZ2ovOqdl-kWDh3jDh-GvgToRIVfwlUFW>
- Fazt Code (2024). "Ejercicio: Aplicación CRUD en Angular". <https://youtu.be/arGRTVdG--c>
- JAAX (2023). "Cascading, FetchType". <https://youtu.be/AnBXZK-WA1U>