

**SERVICIO NACIONAL DE APRENDIZAJE – SENA**



**ANÁLISIS Y DESARROLLO DE SOFTWARE  
FICHA 2834914**

**Evidencia de desempeño: GA7-220501096-AA3-EV01 codificación de módulos del software Stand alone, web y móvil de acuerdo al proyecto a desarrollar**

**TUTOR**

**JORGE MARTÍNEZ**

**APRENDIZ**

**JUAN CAMILO GUTIÉRREZ JIMÉNEZ**

**2025**

# INTRODUCCIÓN

Como parte del proceso de formación, se desarrolló el módulo **DorchReporting** del proyecto **Control Dorchester**, enfocado en el registro y gestión de novedades e incidentes por parte del personal de vigilancia. La codificación fue realizada utilizando el framework **Spring Boot**, facilitando una arquitectura limpia y escalable con enfoque en servicios web.

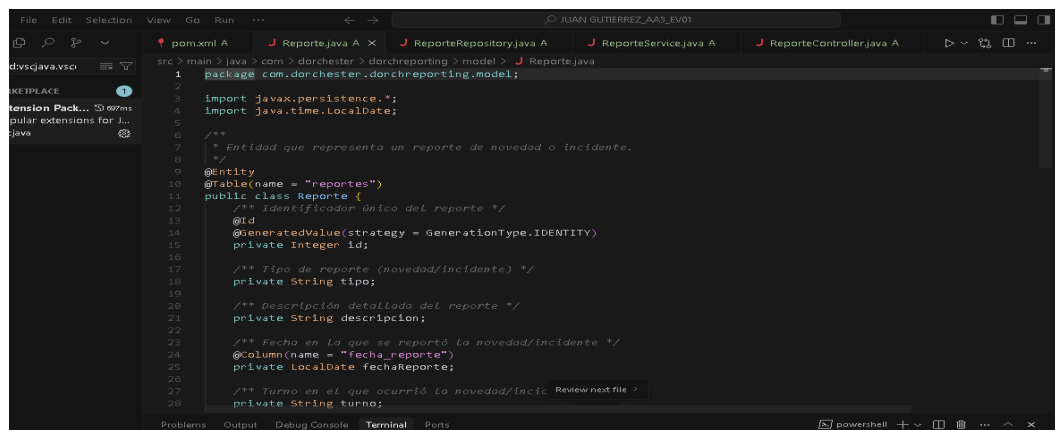
Este desarrollo se basa en los artefactos previamente elaborados en el ciclo del software: historias de usuario, casos de uso, diagrama de clases, prototipos, y plan técnico de trabajo. Además, se cumplieron los estándares de codificación, se aplicaron buenas prácticas, se incluyeron comentarios explicativos en el código y se utilizó **Git** como herramienta de versionamiento.

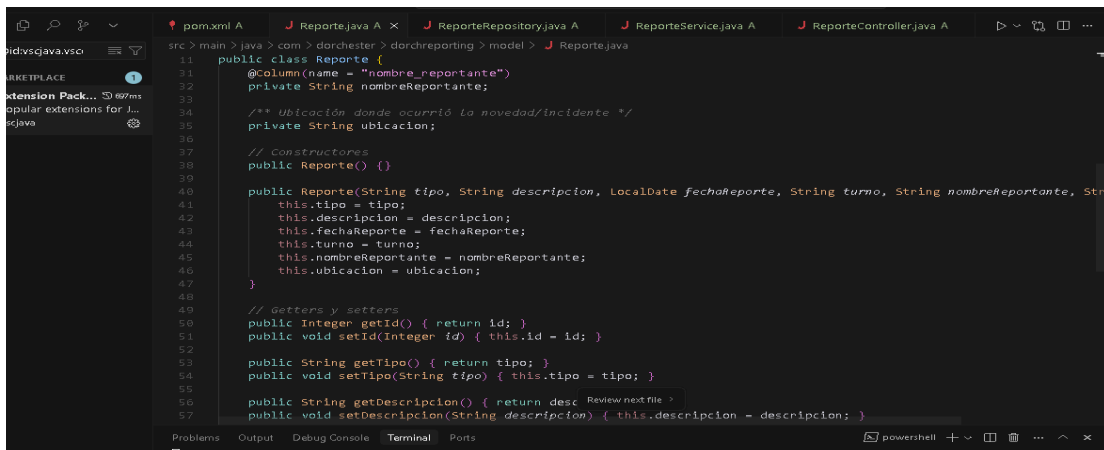
## Herramientas utilizadas

- **Lenguaje:** Java
- **Framework:** Spring Boot
- **IDE:** Visual Studio Code
- **Base de datos:** MySQL (Workbench)
- **Dependencias:** Spring Web, Spring Data JPA, MySQL Connector
- **Control de versiones:** Git y GitHub
- **Arquitectura:** MVC (Modelo - Vista - Controlador)

## Estructura del Proyecto

Organización del proyecto según buenas prácticas:

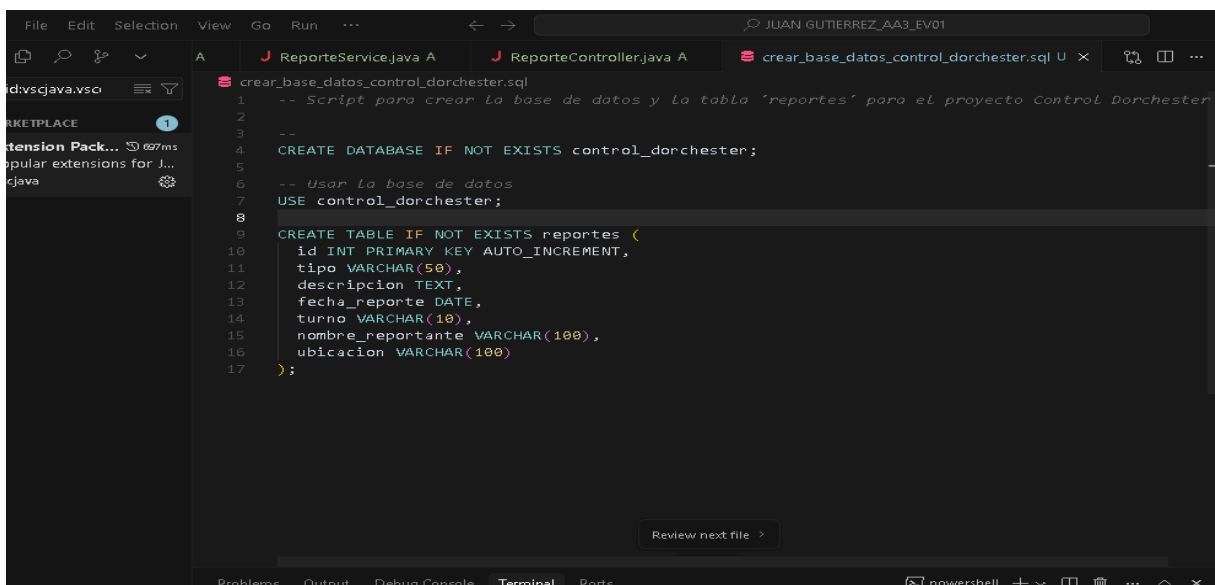




```
11 public class Reporte {
12     @Column(name = "nombre_reportante")
13     private String nombreReportante;
14
15     /** Ubicación donde ocurrió la novedad/incidente */
16     private String ubicacion;
17
18     // Constructores
19     public Reporte() {}
20
21     public Reporte(String tipo, String descripcion, LocalDate fechaReporte, String turno, String nombreReportante, String ubicacion) {
22         this.tipo = tipo;
23         this.descripcion = descripcion;
24         this.fechaReporte = fechaReporte;
25         this.turno = turno;
26         this.nombreReportante = nombreReportante;
27         this.ubicacion = ubicacion;
28     }
29
30     // getters y setters
31     public Integer getId() { return id; }
32     public void setId(Integer id) { this.id = id; }
33
34     public String getTipo() { return tipo; }
35     public void setTipo(String tipo) { this.tipo = tipo; }
36
37     public String getDescripcion() { return descripcion; }
38     public void setDescripcion(String descripcion) { this.descripcion = descripcion; }
```

## Creación de la base de datos y tabla

El script `db_reportes.sql` crea la base de datos `control_dorchester` y la tabla `reportes`, donde se almacenan los registros.



```
1 -- Script para crear la base de datos y la tabla 'reportes' para el proyecto Control Dorchester
2
3 --
4 CREATE DATABASE IF NOT EXISTS control_dorchester;
5
6 -- Usar la base de datos
7 USE control_dorchester;
8
9 CREATE TABLE IF NOT EXISTS reportes (
10     id INT PRIMARY KEY AUTO_INCREMENT,
11     tipo VARCHAR(50),
12     descripcion TEXT,
13     fecha_reporte DATE,
14     turno VARCHAR(10),
15     nombre_reportante VARCHAR(100),
16     ubicacion VARCHAR(100)
17 );
```

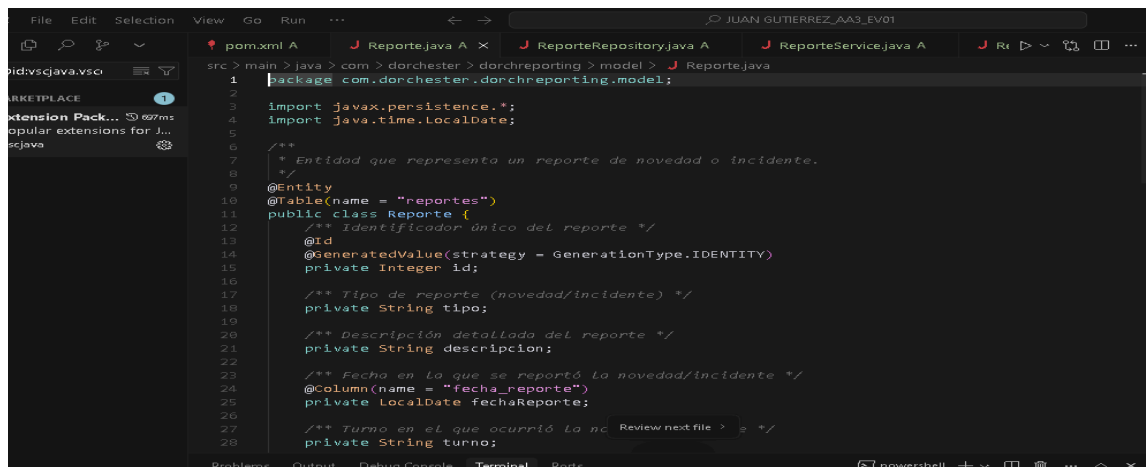
## FUNCIONALIDADES IMPLEMENTADAS

Función	Descripción
Insertar	Agrega un nuevo reporte a la base de datos.
Consultar	Muestra todos los reportes almacenados.
Actualizar	Permite modificar los datos de un reporte existente.
Eliminar	Elimina un reporte seleccionado de la base de datos.

## CÓDIGO DESTACADO DEL MÓDULO

A continuación, se presentan ejemplos del código utilizado en cada componente del módulo. Todos los archivos contienen **comentarios explicativos** y están organizados siguiendo el estándar MVC.

**Modelo:** `Reporte.java`



```
1 package com.dorchester.dorchreporting.model;
2
3 import javax.persistence.*;
4 import java.time.LocalDate;
5
6 /**
7  * Entidad que representa un reporte de novedad o incidente.
8  */
9 @Entity
10 @Table(name = "reportes")
11 public class Reporte {
12     /** Identificador unico del reporte */
13     @Id
14     @GeneratedValue(strategy = GenerationType.IDENTITY)
15     private Integer id;
16
17     /** Tipo de reporte (novedad/incidente) */
18     private String tipo;
19
20     /** Descripción detallada del reporte */
21     private String descripcion;
22
23     /** Fecha en la que se reportó la novedad/incidente */
24     @Column(name = "fecha_reporte")
25     private LocalDate fechaReporte;
26
27     /** Turno en el que ocurrió la novedad/incidente */
28     private String turno;
```

```
File Edit Selection View Go Run ... JUAN GUTIERREZ_AA3_EV01
pom.xml A Reporte.java A ReporteRepository.java A ReporteService.java A ReporteController.java A
src > main > java > com > dorchester > dorchreporting > model > Reporte.java
1 public class Reporte {
2     private Integer id;
3     private String turno;
4     /** nombre de la persona que reporta */
5     @column(name = "nombre_reportante")
6     private String nombreReportante;
7     /** ubicación donde ocurrió la suvedad/incidente */
8     private String ubicacion;
9     // Constructors
10    public Reporte() {}
11    public Reporte(Integer id, String tipo, String descripcion, LocalDate fechaReporte, String turno, String nombreReportante, String ubicacion) {
12        this.id = id;
13        this.tipo = tipo;
14        this.descripcion = descripcion;
15        this.fechaReporte = fechaReporte;
16        this.turno = turno;
17        this.nombreReportante = nombreReportante;
18        this.ubicacion = ubicacion;
19    }
20    // Getters y setters
21    public Integer getId() { return id; }
22    public void setId(Integer id) { this.id = id; }
23    public String getTipo() { return tipo; }
24    public void setTipo(String tipo) { this.tipo = tipo; }
25    public String getDescripcion() { return descripcion; }
26    public void setDescripcion(String descripcion) { this.descripcion = descripcion; }
27    public LocalDate getFechaReporte() { return fechaReporte; }
28    public void setFechaReporte(LocalDate fechaReporte) { this.fechaReporte = fechaReporte; }
29    public String getTurno() { return turno; }
30    public void setTurno(String turno) { this.turno = turno; }
31    public String getNombreReportante() { return nombreReportante; }
32    public void setNombreReportante(String nombreReportante) { this.nombreReportante = nombreReportante; }
33    public String getUbicacion() { return ubicacion; }
34    public void setUbicacion(String ubicacion) { this.ubicacion = ubicacion; }
35 }
```

```
File Edit Selection View Go Run ... JUAN GUTIERREZ_AA3_EV01
pom.xml A Reporte.java A ReporteRepository.java A ReporteService.java A ReporteController.java A
src > main > java > com > dorchester > dorchreporting > model > Reporte.java
11 public class Reporte {
12     // getters y setters
13     public Integer getId() { return id; }
14     public void setId(Integer id) { this.id = id; }
15
16     public String getTipo() { return tipo; }
17     public void setTipo(String tipo) { this.tipo = tipo; }
18
19     public String getDescripcion() { return descripcion; }
20     public void setDescripcion(String descripcion) { this.descripcion = descripcion; }
21
22     public LocalDate getFechaReporte() { return fechaReporte; }
23     public void setFechaReporte(LocalDate fechaReporte) { this.fechaReporte = fechaReporte; }
24
25     public String getTurno() { return turno; }
26     public void setTurno(String turno) { this.turno = turno; }
27
28     public String getNombreReportante() { return nombreReportante; }
29     public void setNombreReportante(String nombreReportante) { this.nombreReportante = nombreReportante; }
30
31     public String getUbicacion() { return ubicacion; }
32     public void setUbicacion(String ubicacion) { this.ubicacion = ubicacion; }
33 }
```

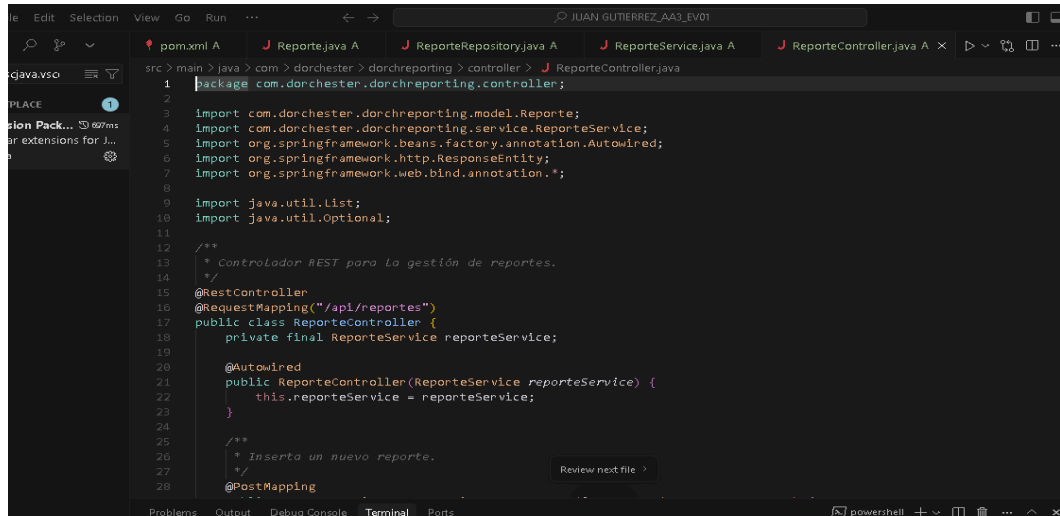
## Repositorio: ReporteRepository.java

```
File Edit Selection View Go Run ... JUAN GUTIERREZ_AA3_EV01
pom.xml A Reporte.java A ReporteRepository.java A ReporteService.java A ReporteController.java A
src > main > java > com > dorchester > dorchreporting > repository > ReporteRepository.java
1 package com.dorchester.dorchreporting.repository;
2
3 import com.dorchester.dorchreporting.model.Reporte;
4 import org.springframework.data.jpa.repository.JpaRepository;
5 import org.springframework.stereotype.Repository;
6
7 /**
8  * Repositorio para la entidad Reporte.
9  * Proporciona métodos CRUD usando Spring Data JPA.
10  */
11 @Repository
12 public interface ReporteRepository extends JpaRepository<Reporte, Integer> {
13     // Métodos CRUD ya disponibles por JpaRepository
14 }
```

## Servicio: ReporteService.java

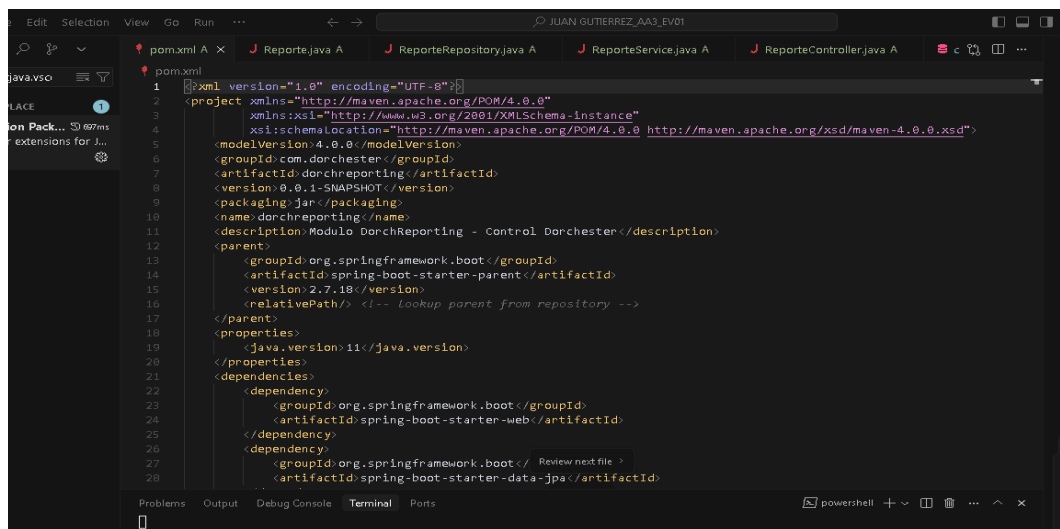
```
File Edit Selection View Go Run ... JUAN GUTIERREZ_AA3_EV01
pom.xml A Reporte.java A ReporteRepository.java A ReporteService.java A ReporteController.java A
src > main > java > com > dorchester > dorchreporting > service > ReporteService.java
1 package com.dorchester.dorchreporting.service;
2
3 import com.dorchester.dorchreporting.model.Reporte;
4 import com.dorchester.dorchreporting.repository.ReporteRepository;
5 import org.springframework.beans.factory.annotation.Autowired;
6 import org.springframework.stereotype.Service;
7
8 import java.util.List;
9 import java.util.Optional;
10
11 /**
12  * Servicio que contiene la lógica de negocio para la gestión de reportes.
13  */
14 @Service
15 public class ReporteService {
16     private final ReporteRepository reporteRepository;
17
18     @Autowired
19     public ReporteService(ReporteRepository reporteRepository) {
20         this.reporteRepository = reporteRepository;
21     }
22
23     /**
24      * Inserta un nuevo reporte en la base de datos.
25      */
26     public Reporte insertarReporte(Reporte reporte) {
27         return reporteRepository.save(reporte);
28     }
```

## Controlador: `ReporteController.java`



```
1 package com.dorchester.dorchreporting.controller;
2
3 import com.dorchester.dorchreporting.model.Reporte;
4 import com.dorchester.dorchreporting.service.ReporteService;
5 import org.springframework.beans.factory.annotation.Autowired;
6 import org.springframework.http.ResponseEntity;
7 import org.springframework.web.bind.annotation.*;
8
9 import java.util.List;
10 import java.util.Optional;
11
12 /**
13  * Controlador REST para la gestión de reportes.
14  */
15 @RestController
16 @RequestMapping("/api/reportes")
17 public class ReporteController {
18     private final ReporteService reporteService;
19
20     @Autowired
21     public ReporteController(ReporteService reporteService) {
22         this.reporteService = reporteService;
23     }
24
25     /**
26      * Inserta un nuevo reporte.
27      */
28     @PostMapping
```

## Configuración: `application.properties`



```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0"
3         xmlns:xsi="http://maven.apache.org/2001/XMLSchema-instance"
4         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
5     <modelVersion>4.0.0</modelVersion>
6     <groupId>com.dorchester</groupId>
7     <artifactId>dorchreporting</artifactId>
8     <version>0.0.1-SNAPSHOT</version>
9     <packaging>jar</packaging>
10    <name>dorchreporting</name>
11    <description>Module Dorchester - Control Dorchester</description>
12    <parent>
13        <groupId>org.springframework.boot</groupId>
14        <artifactId>spring-boot-starter-parent</artifactId>
15        <version>2.7.18</version>
16        <relativePath/> <!-- Lookup parent from repository -->
17    </parent>
18    <properties>
19        <java.version>11</java.version>
20    </properties>
21    <dependencies>
22        <dependency>
23            <groupId>org.springframework.boot</groupId>
24            <artifactId>spring-boot-starter-web</artifactId>
25        </dependency>
26        <dependency>
27            <groupId>org.springframework.boot</groupId>
28            <artifactId>spring-boot-starter-data-jpa</artifactId>
```

## ARCHIVOS ENTREGADOS

La carpeta comprimida entregada tiene por nombre:

**JUAN\_GUTIERREZ\_AA3\_EV01.zip**

Esta incluye:

- Código fuente completo (`/src`)
- Archivo `crear_base_datos_control_dorchester.sql`
- Archivo `README.md`

- Archivo `enlace_github.txt`

**Enlace al repositorio GitHub:**

 <https://github.com/juangutierrez/dorchreporting>

## CONCLUSIÓN

El desarrollo del módulo **DorchReporting** permitió aplicar un enfoque estructurado en la construcción de software, utilizando herramientas profesionales y un framework moderno como **Spring Boot**. El resultado es una aplicación funcional, documentada y preparada para su integración en el sistema completo del proyecto **Control Dorchester**.

Se evidencia el dominio de conceptos fundamentales como arquitectura MVC, control de versiones, conexión a bases de datos, diseño modular y buenas prácticas de programación, tal como lo exige la actividad GA7-220501096-AA3-EV01.