1. Que es la inyección de dependencia y representa los tipos de inyección de dependencia que existen.

R= Patron de diseño de software que nos permite integrar propiedades de objetos fuera de la clase (P)

1. Menciona al menos 5 patrones de diseño y para que los utilizas.

R=

* Patron Factory: nos permite crear objetos, pero no hay necesidad se saber que tipo de objeto es
* Patron Singleton : esto nos indica que solo existe una instancia de la clase en el programa
* Patron observer: Nos permite identificar cuando un objeto cambio de estado
* Patron Diseñador: Nos permite añadir un nuevo método sin alterar su estructura
* Patron Estructura: Este se encarga de encapsular el algoritmo

1. Representa y explica cada uno de los componentes de la arquitectura del framework Spring.

R=

* Spring web – levantas servicios web mediante servidor TOMCAT
* Spring Security – Permite utilizar un token para el envio de validaciones como OAUTH
* Spring Tools- Permite crear tu modelo de aplicación con las dependencias a utilizar
* Spring Data: Usualmente nos permite trabajar con los datos para la conexión a una DB

1. Describe para que se utilizan las siguientes anotaciones:

@Autowired: Inyecta las dependencias a la clase

@Qualifier: Va de la mano con @Autowired para ver que dependencia se debe inyectar

@PostConstruct: Indica que debemos llamar al metodo después de la inyección de dependencia

@PreDestroy: Este nos indica que debemos llamar al método antes de ser eliminado de una instancia

@Required: Indica que un campo es de uso obligatorio

@Bean: Marca un método que nos devuelve un objeto

@Configuration: Este va de la mano con @Bean para marcar una clase que tiene configuración de Spring

@Transactional:

@Secured: Indica que un campo tendrá que ser llamado por token y mediante roles(P)

@RollesAllowed: Indica que este campo será accesible mediante permisiones

@ManagedResource:

@ManageAttribute:

@ManageOperation:

@Cacheable:P

R= No recuerdo todas

1. Para que sirve un converter.

R= Usualmente se usa para convertir un tipo de dato en OTRO como DTO en Spring

1. Que es un aspecto y describe un ejemplo de cómo lo podríamos implementar

R= Un aspecto es la forma en la que vamos a trabajar cierto proyecto a nivel POO, saber en que momento llamaremos un método y que configuraciones debemos aplicar desde el archivo de configuraciones, normalmente tenemos un archivo POM para ajustar este tipo de configuraciones.

Ejemplo: Tener una clase llamada Coche

Creamos el constructor para asignar las características de los atributos del objeto Coche

Instanciamos la clase con el objeto Coche y creamos por ejemplo 3 metodos:

Coche Mazda = new Coche();

Mazda.Avanzar

Mazda.Frenar

Mazda.Girar

Estos métodos los podemos usar en cualquier momento, pero al tener atributos como:

Mazda.color;

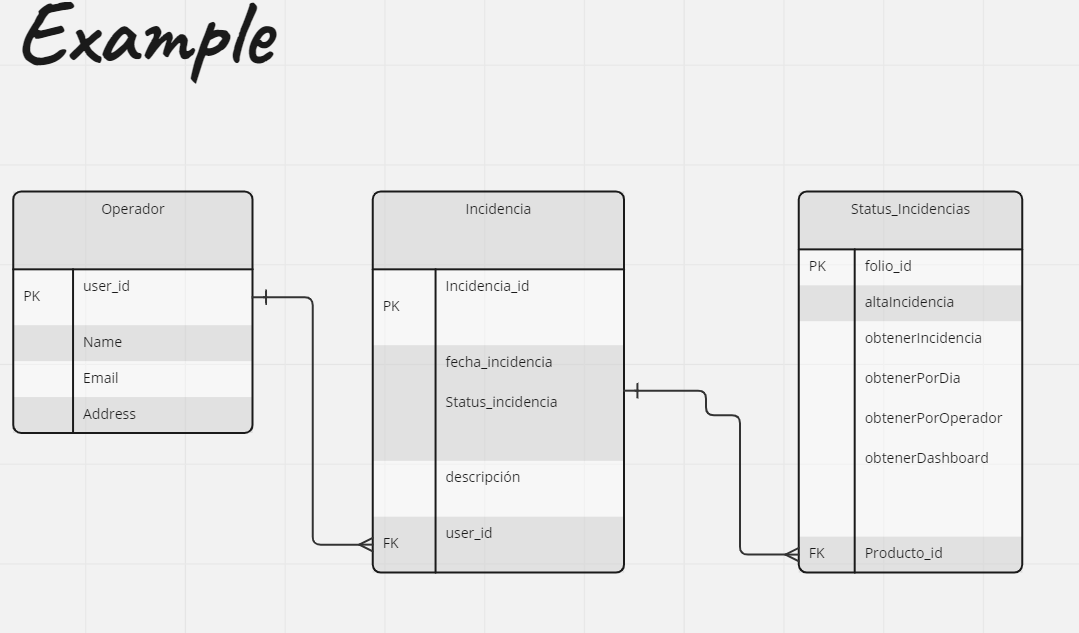
Mazda.Largo;

Mazda.Ancho;

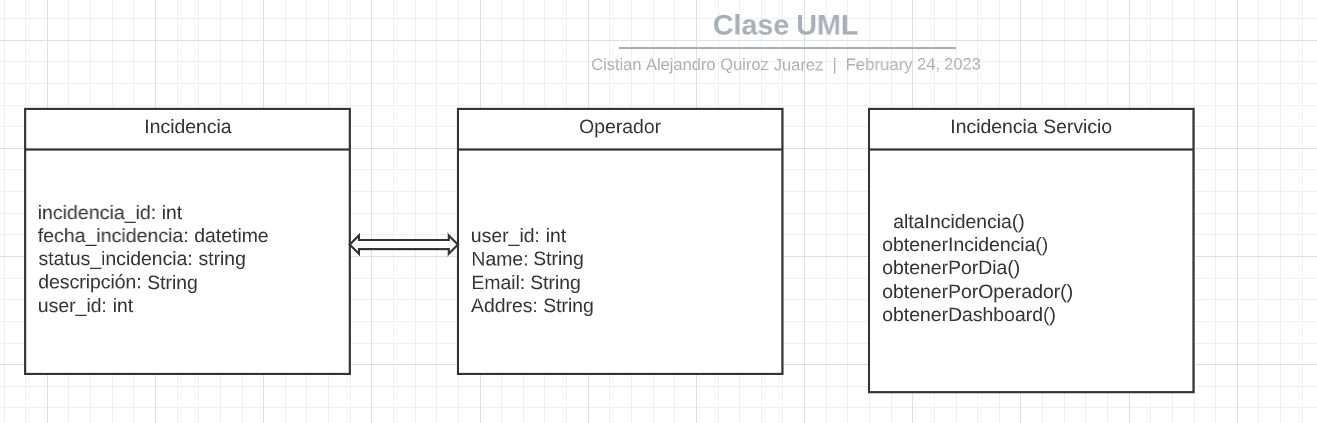
No podemos llamar a esos atributos de forma normal ya que tenemos que decidir si serán tipo Public, Private o Protected, es ahí cuando nosotros aplicamos la configuración necesaria para esta clase. Esos son los aspectos, las configuraciones que podremos aplicar.

1. La empresa XYZ es la encargada de administrar el reporte de incidencias de los equipos de cómputo, donde un operador puede dar de alta una incidencia (“definir los datos que consideres más importante ”), consultar incidencias por ID, por día , y por operador que las dio de alta , cabe mencionar que debemos tener dashboards que representen número de peticiones a los servicios consultados , insertados , cuantos se resolvieron , cuantos están abiertos.

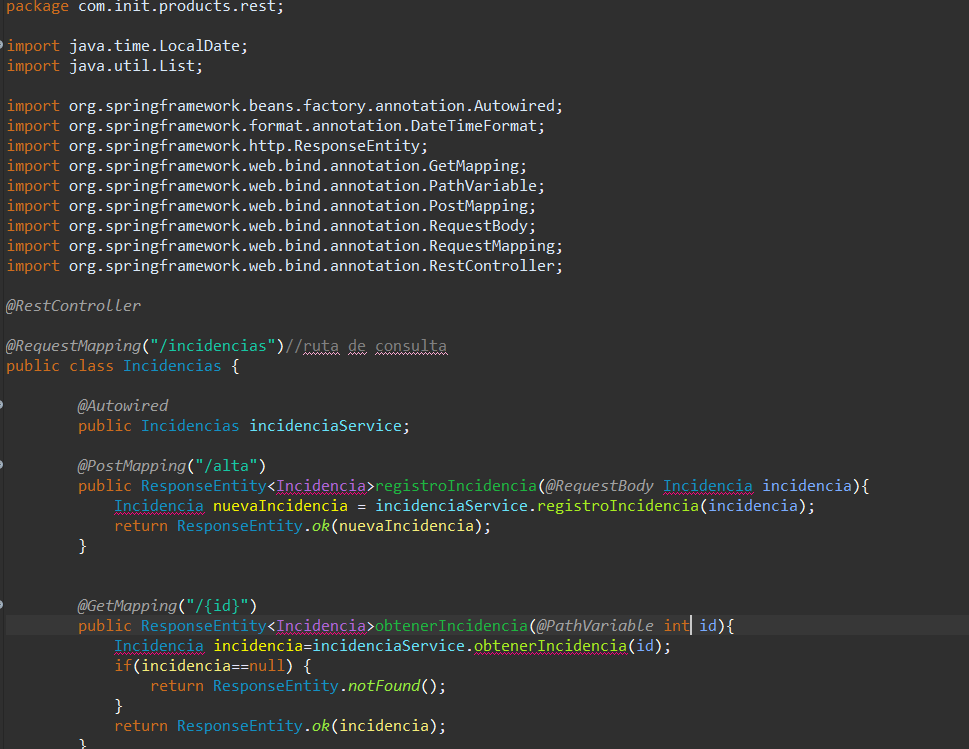
a. Realizar el diagrama E-R



b. Realizar el diagrama de clases



c. y programar un servicios rest de consulta e inserción que consideren mas importantes.





1. Que es la arquitectura de micro servicios.

R=

Te permite crear pequeños servicios como si fueran contenedores, cada se ejecuta con su propio proceso y se relacionan entre todos.

1. Menciona para que sirve cada uno de los siguientes componentes a. Spring CloudConfig b. Eurka c. Ribbon d. Hystrix e. Zuul f. OAUT g. SWAGGER h. SPLUNK

R=

Solo he usado OAUTH- Para validar claves para los usuarios al momento de realizar un login, genera un token que el server debe devolver y se debe validar.

SWAGGER- Lo puedes implementar dentro de Spring Boot para documentar tu código, toda la documentación te indica el tipo de dato, el método y la forma de consulta.