Índice general

CAPÍTULO 1

Preguntas tecnicas FullStack

TDD

Es el desarrollo en base a pruebas.

Ajax

Tecnologia para hacer peticiones de manera asincrona.

div vs span

El div es un bloque donde se puede poner otros bloques en span esta limitado a elementos en una linea

Beneficios de SOLID

Codigo mantenible,facil de escalarlo con el tiempo,desacoplado,buenas practicas.

Verbos HTTP

POST Envio de informacion PUT Actualizacion total de un recurso GET Recuperar informacion DELETE Para eliminar un recurso PACTH Actualizacion parcial de un recurso OPTION Para otro tipo de operaciones

Codigos de respuesta HTTP

1XX 2XX Operacion exitosa 3XX 4XX Error por parte del cliente 5XX Error por parte del servidor

Como despliegas una aplicacion fullstack

De forma independiente se puede desplegar el backend y frontend. El backend en un servidor tradicional Jboss,Wildfly,usar un contenedor y subirlo a un proveedor cloud. El frontend genera archivos compilados: css,js,html y estos pueden subirse a un apache,s3(ngnix). Obviamente hay un tema de arquitectura.

SQL VS NoSQL

Si el app su prioridad los datos necesitan tener estructura y respetar los principios ACID entonces BD relacional (SQL) .En cambio si se quiere escalabilidad,flexibilidad en la extructura,busqueda rapida se recomienda BD no relacionales ()NOSQL).

Diferencia entre REST Y SOAP

Rest es una arquitectura en cambio SOAP es un protocolo que necesita un contrato(WDSL).
Rest es mas sencillo de usar.

Diferencias LET,CONST,VAR

VAR :Esta obsoleto. LET :Define una variable en el alcance donde fue declarada esa variable CONST : No se puede reasignar un valor.

Funcion flecha

Una forma abreviada de declarar una funcion.Una funcion anonima.

Normalizacion de datos

Tecnica que se usa para reducir la redundancia.Mantener la integridad.

Como proteger los servicios expuestos por el backend

Protocolo u2,el header con un identificador ajeno a JWT.HTTPS.

Utilidad de JWT

Un mecanismo que tiene un formato especial,que al decodificarlo presenta una firma,payload, asignasure,header.Es un identificador.

OAUTH2

Un protocolo de autorizacion para acceder a ciertos permisos en otras aplicaciones.

Gestores o productos de seguridad

WSO2 Identity server.
Key cloack.
AWS Cognito.

Se puede modificar el payload de un JWT

Si ,pero la firma lo detectara como corrupto.

Angular VS React

Caundo hay mucha manipulacion o demanda de la pagina,react tiene mejores tiempos de respuesta. En cambio angular tiene las tecnologias listas para usar,standares y personal tecnologico mas disponible.

Como proteges los aplicativos en el frontend

Mediante accesos no autorizados.
Proteger la autorizacion.Acceder a un recurso que no me pertenece.
Validar la caducidad de los tokens.

Programacion re4activa

Procesos asincronos,mejores tiempos de respuesta,orientado a eventos.

Que es un MOCK

Objetos ya programados,con una salida esperada.Datos ficticios.Usado en testing.

Inyeccion de dependencias

Es un patron de diseno.La misma tecnologia es el encargado de proporcionar las instancias en el tiempo oportuno.Permite desacoplar el codigo.

Preprocesadores de css conoces.

less
Sas
Styles.

UI VS UIX

UI es la interfaz de usuario.Mientras que UIX se refiere a la experiencia de usuario,la usabilidad.

Que es COORS

CSFR

Ataque se envia codigo malicioso al servidor.Ataca vulneravilidades del cliente para explotar un servicio

Evitar inyeccion SQL

La misma dependencia de JPA prevee esto.Usando procedimientos almacenados

Que es una interface en Typescript

Define una estructura,como una plantilla.

Especificacion de Javascript

Ecmascript .Apartir de Ecmascript 6 aparecio la definicion de let y const. Tipo de datos de Javascript:String,booleano,objeto e indefinido.

Que hace el operador delete de JS

Elimina una propiedad de un objeto.

Storage usado en navegadores
Local storage,su alcance a pesar de cerrar session continua.
Session storage ,su alcance es mas temporal. La diferencia entre ambos es el alcance.

DOM
Es el arbol de nodos de las etiquetas HTML.

Callback
Funcion que tiene paramtro otra funcion.

Callback Hell
Se anidan varios callback.

Que es un observable
Es un flujo de datos,para que cualquier suscripcion se entere si ha existido un cambio.

Landinpage usarias jquery o angular
Landinpage si es pequeño se usa jquery, si el Landinpage va a crecer o se va a reutilizar entonces angular.

Nivel junior
1. ¿Cuál es la diferencia entre error y excepción?
2. Diferencia entre finally y finalize
3. ¿Para qué sirve la palabra reservada final?
4. ¿Cuáles son los 4 pilares de la POO?
5. ¿Cuál es la diferencia entre una interfaz y una clase abstracta?
6. ¿Qué es un Servlet?

Nivel Semi Senior
1. ¿Qué es el autoboxing?
2. Diferencia entre equals e ==
3. Define agregación y su diferencia contra composición
4. ¿Qué tipos de excepciones existen?
5. ¿Cuál es la diferencia entre overload y override?
6. ¿Qué es un deadlock?

Nivel Senior
1. ¿Para qué sirve transient?
2. Explica el ciclo de vida de un servlet
3. ¿Cuál es la diferencia entre hilo y proceso?
4. Explica el ciclo de vida de un hilo
5. ¿Cómo evitas un deadlock?
6. ¿Cómo se usa la palabra reservada synchronized?
7. ¿Cuál es la diferencia entre los métodos sleep y await en hilos?

Preguntas de Base de datos
1. ¿Para qué sirven los índices?
2. ¿Cuál es la diferencia entre where y having?
3. Diferencia entre procedimiento almacenado y función
4. Formas de unir dos tablas de una base de datos (joins)
5. ¿Qué es la normalización?

Preguntas de Middleware
1. ¿Qué métodos de HTTP existen?
2. ¿Qué es REST?
3. ¿Cuál es la diferencia entre REST y SOAP?
4. ¿Para qué sirve qualified? (realmente es pregunta sobre singleton)
5. ¿Cómo se usa la anotación value?
6. ¿Qué es la inversión de control?
7. ¿Qué es la inyección de dependencias?
8. Explica que es el contenedor IOC de spring
9. ¿Qué anotaciones se usan en spring para inyección de dependencias?

10. Diferencia entre spring y spring boot
11. Diferencia entre Repository, Service y Controller
12. Formas de inyectar dependencias en Spring
13. ¿Cuál es el ciclo de vida de spring?

CAPÍTULO 2

Preguntas tecnicas Java y Spring

Diferencias entre Spring y Spring Boot

Es un proyecto de Spring.Se añade un servidor de aplicaciones embebido,autoconfiguraciones y elimina el trabajo con XML.Es decir configuraciones en XML.

Que es un Bean

Es una pieza de software que representa un objeto dentro de spring .Son manejados por el contenedor de spring.Tiene un ciclo de vida.Son reusables.

Que es el ambito de un bean

Define el ciclo de vida y visibilidad del bean en el contexto.Se define con la anotacion @Scope.
Singleton :Por defecto y rara vez sera necesario usar otro diferente.El bean va a tener la misma instancia durante todo el aplicativo.
Prototype:Es una copia de una instancia.
Request: Crea una nueva instancia por cada peticion HTTP ,solo en una aplicacion web se puede utilizar.
Session: Crea una instancia por cada sesion HTTP.
Application
Websocket

Que son los starters y cual es su objetivo

Es una utilidad que facilita la creacion y configuracion de aplicaciones.
Agregan dependencias necesarias para un objetivo y parametros de configuracion.Ejemplo: spring-boot-starter-web spring-boot-starter-data-jpa

Cuales son las principales verbos/metodos HTTP y como se pueden implementar con Spring Boot

Se pueden implementar con Spring Boot.
GET @GetMapping : Para obtener informacion.
POST @PostMapping : Utilizado para enviar informacion.
PUT @PutMapping : Reemplaza el contenido del recurso con el nuevo.
DELETE @DeleteMapping : Borra la informacion del recurso especificado.
PATCH @PatchMapping : Aplica modificaciones parciales al recurso especificado.

Que diferencia existe entre JPA e Hibernate

Java Persistance API define una especificacion o interfaces para el manejo de persistencias.
Hibernate es una impplementacion particular de JPA.Y es usado por defecto en Spring Boot. JPA es una especificacion e hibernate es una implementacion.

Que es Spring Initializer

Es una plataforma online para generar un proyecto base de Spring Boot.Se puede usar por la web directamente o desde los principales IDES como STS o Intellij.

Que diferencias existen entre @Repository @Service @Component @Controller

No existen diferencias funcionales,solo semanticas.Son anotaciones para marcar clases en Spring y que se creen benas a partir de ellas .Ejemplo.:Las clases tipo servicio(@Component,@Service).

Explique la anotacion @RestController

Facilita la creacion de API REST usando spring MVC. Combina @Controller+@ResponseBody.Permite serializar cualquier respuesta de un controlador Spring MVC directamente a json o XML.

Que es Spring AOP

AOP significa programacion orientada a aspectos. Permite crear aspectos en nuestras aplicaciones para ser usados de forma transversal Ejemplo.: Crear una anotacion que permite auditar la realizacion de determinadas acciones.

La anotacion @Profile que funcion tiene y porque es importante

@Profile permite usar beans determinados en dependencia del perfil activo en una aplicacion Spring Boot.
Los perfiles permiten instancias de forma particular una aplicacion en dependencia de un escenario por ejemplo: dev,qa,prod.

Que hace la anotacion @SpringBootApplication internamente

Anotación principal de spring boot,ubicada por defecto en la clase principal de un proyecto de spring boot. Combina a las anotaciones de :
@Configuration
@EnableAutoConfiguration
@ComponentScan
Permite inicilizar todo lo necesario para que nuestra aplicacion SpringBoot pueda iniciar a trabajar correctamente.

Que patrones estan presentes en Spring/SpringBoot agregue un ejemplo

Creacional:Singleton.Ejemplo:Al crear un bean anotando una clase con @Service y @Scope por defecto usamos singleton.
Estructural:Proxy.Ejemplo.:Usado por @Transactional para los sistemas de persistencia.
Comportamiento:Cache Template.

Para que se usa @TransacTional

En BD una operacion es transactional si mantiene el estado en un sistema consistente.
@Transactional permite ejecutar operaciones en spring en forma de todo o nada.Si algo sale mal durante la ejecucion de una operacion @Transactional el sistema se deja en su estado inicial.

Que es la inyeccion de dependencias

DI es un patron para separar responsabilidades evitando que una clase sea responsable de instanciar objetos de otra.Promueve el bajo acoplamiento.
Al arrancar Spring se crean los objetos (beans) y luego se inyectan donde son necesarios usando @Autowired.

Cuales son las formas de inyecccion de dependencias que existen en Spring y cual es el mas adecuado

Inyeccion por atributo.
Inyeccion por setter.
Inyeccion por constructor;Facilita el testeo de la aplicacion de una manera mas sencilla.

Para que se usa @Value y que lenguaje usa la anotacion para su funcion

@Value permite leer configuraciones y colocarlas en la propiedad anotada.Usa lenguaje SpEL.Ejemplo.:

```
@Value("${value.from.file}")
```

Cual es el puerto por defecto de una aplicacion Spring Boot y como podemos cambiarlo

Por defecto es el puerto 8080.Se cambia en la propiedad server.port del fichero de configuracion.

Que es Spring Actuator

Libreria que proporciona herramientas de monitoreo y administracion para un API REST.
Habilita varios endpoints para ser usados .Ejemplo:/beans
/health
/shutdown.
Permite ser extendido

Que es Spring Cloud

Spring Cloud es un proyecto basado en Spring Boot .
Implementa soluciones production-ready para el desarrollo de microservicios.
Algunos componentes son: Spring Cloud Config.
Spring Cloud Gateway.
Esta influenciado por NetflixOSS.

Spring IoC Conteiner

Es el contenedor priincipal de Spring.Utiliza las inyecciones de independencias para administrar el ciclo de vida de todos los objetos que Spring va a gestionar.

Que es un Autowired

Es la tecnica o forma de como hacer las inyecciones automaticamente de las instancias de los beans.
Con la anotacion @Autowired.

Como indicas que bean utilizar

Mediante la anotacion @Qualifine me permite indicar el nombre exacto del bean que necesito.Tambien con @Primary.

Como se gestionan las transacciones en Spring

Se trabaja mediante el uso de @Transactional.

Se puede trabajar Spring solo con java

También se puede trabajar con Kotlin y Groovy.

Cuales son los archivos de configuración de Spring Boot

application.property (yml/properties).
Bootstrap.yml.

Cuales son los estereotipos de Spring y que funciones cumplen

@Repository : Contextualizar la clase para acceso a B.D..
@Service : Contextualizar la clase para lógica de negocios.
@Controller : Contextualizar la clase para que sea un controlador.
@Component : Contextualizar la clase para que sea un utilitario.
Cuando se estereotipa una clase se puede hacer una inyeccion de dependencias de una instancia de esa clase.

Las aplicaciones que SpringApplication resume

@SpringBootConfiguration.
@EnableAutoConfiguration.
@ComponentScan.

Que contenedores embebeidos puede tener Spring Boot

Tomcat y Jetty.

Como ejecutas en consola el codigo de Spring

Se debe implementar la interface comand Nai Runner.

Como lees el archivo properties

Mediante @Value para extraer la información mediante una llave

Establecer entornos de QA,DEV,PROD

Se usa properties para cada entorno y se reconoce mediante sufijos .
Application-dev.properties
Application-test.properties
Application-prod.properties

Spring Boot es compatible con proyectos antiguos de Spring

Si es compatible usando la anotacion @importVisors

Que anotacion usas para obtener el json y enviarlo en objeto en java

@RequestBody.

Como se gestiona el tema de la internalización o 18n

El local resolver maneja concepto de bundle.Se puede cambiar el tema del idioma a las respuestas que Spring ofrece.

Spring Security forma parte de Spring Framework

No forma parte , es un proyecto independiente como cloud,data,batch.

Defina la arquitectura de microservicios

Es un enfoque para desarrollar una aplicación como un conjunto de pequeños servicios.
Cada uno de los cuales se ejecuta en su propio proceso.
Se comunican con mecanismos ligeros.
Estos servicios se basan en el negocio y se pueden implementar de forma independiente.
Existe un minimo de gestión de centralizada de estos servicios.
Pueden escribirse en diferentes lenguajes de programación y utilizar diferentes tecnologias de almacenamiento de datos.

Ventajas de arquitectura de microservicios

Perrmite asignar equipos especificos pequeños y manejables.
Permite implementar pruebas de forma mas atomicas.
Permite implementar pruebas de integración sectorrizadas.
Es posible usar las tecnologias mas adecuadas en cada microservicio.Por ejemplo : Para busqueda de clientes y productos,podemos incorporar un Elastic Search para busqueda de texto completo.
Facilita desagregar el despliegue de ciertos microserrvicios en diversos nodos y con multiples instancias a demanda (Escalamiento horizntal).

Implementar los retos para la arquitectura de microservicios

Lograr un diseño adecuado.
Gestión de la infraestructurra.
Correcta gestión de la transaccionalidad y la consistencia de los datos.
Gestion del entorno distrribuido y asumir las 8 falacias de la computación distribuida son un elemento a tene en cuenta en la implementación.
Necesidad de la implementacińn de la cultura DevOps.

En la arquitectura de microservicios como se puede resolver el problema de los parametros de configuración de los microservicios y sus instancias

Patrón de configuración centralizada.Spring Cloud Config.

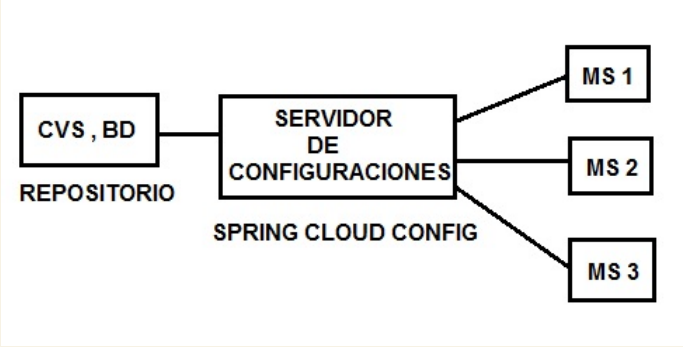

Figura 2.1: Spring Cloud Config

En una arquitectura de microservicios son comunes las llamadas servicio-servicio que mecanismos puede implementar para evitar posibles fallos

Debemos conocer las falacias de la computacińn distribuida.
Dar una solución con reintentos.
Dar una solución con Circuit Breaker.

Porque Java Developer

Es un lenguaje altamente demandado por las industrias y diversos sectores.Java esta consolidado y abierto a un abanico de posibilidades.

Tenemos una semana de atraso, como afrontarias esta situación de estres

Tratar de evaluar la situacion actual, dar solucion a lo prioritario y despues resolver los otros items de acuerdo a los tiempos y apoyo que brinden el equipo en conjunto.

Porque java es independiente

Es gracias a la maquina virrtual que java es multiplataforma y se puede usar en cualquier sistema operativo.

En que situaciones utilizarias los tipos de dato wrappe

Los wrapper pueden tener valores nulos.

Los conceptos claves de POO

Encapsulamiento : Protege el codigo de las demas variables privadas.Getter Setter.
Herencia : Una clase puede extenderrse a otra clase
Polimorfismo : Tiene muchas formas.Un objeto puede referirse a la subclase o superclase segun el tipo de referencia.
Abstraccion: Se selecciona los datos esenciales de un objeto para ser mostrados y oculta otros elementos menos relevantes.

Una interface puede heredar o implementar una interface

Una interface solo puede heredar otra interface.Las clases son las que implementan las interfaces.

Cuando usarias una clase abstracta de una interface

Una clase puede implementar n interfaces (Herencia multiple) .Una herencia tradicional se puede usar una clase abstracta.

Cuando se usaria una list o un set

Una lista permite valores repetidos . Un set (Hash set) no pemite valores duplicados solo uno nomas se tomara en cuenta.

Cuantos metodos abstractos o no implementados puede tener una interface funcional

Una interfaz funcional solo puede tener un metodo abstracto.Para que mas adelante a partir de esa interface se pueda generar una expresion lamda.

Cual es la diferencia entre List y ArayList

List es la interface y ArrayList es la implementacion de esa interface.

Interfaces funcionales que ofrece java

Las mas comunes son el predicate (Test) , consumer (Void) , suplayer (Retorna un objeto).

Cuando usarias Runnnable y thread en una clase

Runnable es una interface y thread es una clase.
Cuando la clase tenga extend se usaria la interface Runnable.
Si el proceso es asincrono y trabaja de la forma tradicional se puede heredar a thread.
Si se quisiera implementar varias interfaces y aparte heredar a otras clases seria mejor usar runnable. Asi pudiendo sobreescribir sus metodos.

Que es la sincronizacion en java

Se debe esperar para que un proceso termine.Para poder pasar al siguiente.Protegiendo la zona critica (asincrono).
Asincrono : Thread,runnable,webflux.

Comprueba usted su codigo

Aplico TDD y trabajo con librerias Junit,Mockito.Tecnicas de definicion de escenarios.

Como ejecutas javascript desde java

Con Nashoin y jjs.

De un conjunto de elementos,como eliminar los elementos duplicados

Con el API stream con filter y disting.
Transformamos la lista a Set (Uso la clase colector).

Comenta sobre el esquema de licencia Java

Java 17 : Es gratis LTS.Hasta un año despues de la proxima libreria.
Java 11 : En produccion es de pago
Java 8 : Empieza ser de pago.
Nota : Amazon Coretto es gratis.

Cambios de java 8 a 11/17

Eliminacion de modulos Java EE.
Sintaxis de variables locales para parametros CORBA en lambdas (@NotNull).
Un cliente HTTP (java.net.http).
Ejecucion desde archivo de codigo fuente unico.
Soporte unicode IO.
Nuevos metodos para nuestros Strings : isBlank(),lines(),strrip(),repeat(int).
Lectura y escritura de string en archivos(readString(path),writeString().
Acceso a clases interrnas mediante metodos que usan reflexion.
Eliminacion de codigo no deseable,deprecables u obsoletos.

Programacion funcional

Los lenguajes de programacion funcional son aquellos lenguajes donde las variables no tienen estado.

Streams

Los streams simplemente ejecutan el flujo de trabajo para el primer elemento o el elemento que cumpla la condicion dada.El flujo termina.Esto permite mejorar el rendimiento ya que no es necesario recorrer la lista completa.
Son envoltorios de condiciones de datos que nos permite operar con estas colecciones y hacer que el procesamiento masivo de datos sea rapido y facil de leer.

Lambda

Son expresiones anonimas , es decir funciones que no necesitan una clase. (parametros) -¿ cuerpo-lambda .
Tipos de lambda : Consumidores,proveedores,funciones(op. unitarios,binarios).Predicados.
Consumidores : Acepta un solo valor y no devuelve valor alguno.
Proveedores : Expresiones que no tienen parametros,sin embargo devuelven un resultado.
Funciones : Aceptan un argumento y devuelven un valor como resultado(Cuyos tipos no tienen que ser iguales).
Predicados : Se trata de expresiones que aceptan un parametro y devuelven un valor logico.

Java optional

Hace referencia a una variable que puede tener un valor asignado o que puede contener un valor null.Metodos isPresent,get().

Patrones de Arquitectura

Patron de capas :
Patron cliente-servidor :
Patron maestro esclavo :
Patron bus :
Patron MVC :
Patron pizarra :

Como activa un perfil

En application.properrties se agrega : spring.prrofiles.active=dev.
La siguiente popiedad en la VM : DSping.profiles.active=dev.

SPEL

Es un lenguaje de expresiones que permite realizar operaciones sobre la informacion en tiempo de ejecucion.

AOP

Agrega comportamioento a codigo existente sin necesidad de modificarlo.
Creacion de aspecto,se define un bean de spring y se anota con @Aspect.Se define el metodo que se ejecutara con el advice.@Before,@AfterReturning,@After.
Nota : Si se tiene multiples advices es posible definir el orden con @Order.

Peticion HTTP

Metodo HTTP,URI,version de protocolo utilizado,heades,body.

Respuesta HTTP

Version del protocolo,status code,headers,body.Status Http:
1xx : Informacional.
2xx : Peticion exitosa.
3xx : Redirecciones.
4xx : Error del lado del cliente.
5xx : Error del lado del servidor.

Soporte para REST

Se debe incluir el modulo Spring MVC.Mediante la dependencia :
spring-boot-stater-web

Configuracion de metricas

Es posible crear metricas propias para analizar el funcionamiento de mi app.Dependencia :
micrometer-registry-prometheus
Consulta de metricas : /actuator/prometheus

Dependencias para usar Swagger

springfox-swagger2
spring-swagger-ui
Para configurar swagger se debe configurar un docket.
Para acceder a la documentacion por defecto:
http://localhost:8080/swagger-ui.html

Configuracion de Cache

Se agrega la dependencia spring-boot-starter-cache.
La anotacion @EnableCaching.
Una vez configurado se debe agregar los objetos al cache ,para esto @Cacheable(Çodes").
Es posible revocar valores de un cache a traves de @CacheEvict.

Concurrent MapCacheManager

Concurrent MapCacheManage el cache se realiza en la maquina que esta ejecutando la app.

Integracion con REDIS

Para hacer la integracion con redis,se agrega la dependencia redisson.
Agregamos el bean de redison al contexto.
Modificamos el CacheManager,para indicar a spring que use redis.
Nota : Los objetos que se escriban en el cache deben implementar la interfaz serializable.

Cuales son los chicos malos.seguridad

El sitio web debe protegerse debido a imitadores,mejoradores y espias.
Imitadores:Doble identificacion MFA de este modo se combina el saber algo(Password) con el tener algo(certificado).
Mejoradores: Permisos suficientes para acceder a un recurso.Esto se consigue mediante la configuracion de ROLES.
ROL es un grupo de usuarios que tiene acceso a un conjunto de recursos especificos.Un usuario puede tener multiples roles.
Espias: Confidencialidad.El cifrado de datos es un metodo comun para asegurar la confidencialidad.

Cambiar password en spring segurity

spring-boot-stater-securrity.Para cambiar el password por defecto de application.properties:
spring.security.user.name
spring.security.user.password

Spring Segurity autentificacion contra una B.D.

Se debe implementar la interfaz UserDetails.No se valida el password , solo se obtiene de la base de datos.

Formas de proteger apliaciones Spring Boot

Usar HTTPS en produccion.
Usar snyk para verificar tus dependencias.
Actualizar la ultima version.
Habilitar proteccion CSRF.
Use estrategias de seguridad de contenido para evitar ataques XSS.

Security . Holder

Es una clase que permite acceder a la informacion tanto del usuario como del rol.
Nota: Los roles seran modificados por spring agregando el prefijo ROLE.Ejemplo : ROLE_ADMIN

Global Security Config

Clase que permite asegurar a nivel de metodo en esta clase se agrega la notacion @EnableBlobalMethodSecurrity.

@Secured

En spring nos permite definir la lista de roles permitidos al ejecutar un metodo.

@RolesAllowed

Es equivalente a @Secured,la diferencia es que la primera no es propia de spring.

@PreAuthorize @PostAuthorize

@PreAuthoize: Hace una evaluacion antes de ejecutar el metodo, dependiendo del resultado de la evaluacion decide si ejecutalo o no.
@PostAuthorize: Permite realizar una evaluacion despues de ejecutar el metodo,dependiendo del resultado de la evaluacion decide devolver o no el valor.Meta anotaciones es posible crear metaanotaciones.
Nota: Para combinar el uso de las anotaciones de seguridad.
Ejemplo: @IsUserOrAdmin tendra el mismo efecto que usar @PreAuthorize y @PostAuthorize.

Status HTTP 403

El error indica que el usuario tiene una autentificacion exitosa, pero fallo al realizar la autorizacion.

S.O.L.I.D.

S Principio de responsabilidad unica : Un obejeto deberia tener una unica razon para cambiar.
O Principio de abierto o cerrado : Las entidades de software deben estar abiertas para su extension, pero cerrados para su modificacion.
L Principio de sustitucion de Liskov : Los objetos de un programa deberia ser reemplazables por instancias de sus subtipos sin alterar el correcto funcionamiento del programa.
I Principio de segregacion de interfaz : Muchas interfaces cliente especificas son mejores que una interfaz de proposito general.
D Principio de inversion de dependencias : Se debe depender de abstracciones,no depender de implementaciones.

Artifactory

Artifactory es un Repositorio de Maven. En el se pueden desplegar las dependencias
que necesiten los proyectos en maven para su ciclo de vida. Otra de las principales
funciones de artifactory es hacer de proxy cache de un repositorio propio con otros existentes.
Facilidad de instalación.
Facilidad de configuración.
Soporte para múltiples repositorios locales.

Sonar Cube

Se puede ver la calidad del codigo.Analiza,hace test,auditoria.No se recomienda usar la BD embebida.

Jenkins4

Jenkins ayuda en la automatización de parte del proceso de desarrollo de software mediante integración continua y facilita ciertos aspectos de la entrega continua. Admite herramientas de control de versiones como CVS, Subversion, Git, Mercurial y puede ejecutar proyectos basados en Apache Ant y Apache Maven.

Streams que mas has usado.Map

Agrupacion de elementos sobrelos que podemos especificar operaciones.
Map es para convertir .
filter es para filtrar.
sorted es para ordenar.
reduce opera sobre los elementos de un stream.

Diferencias entre Rest y SOAP

Rest es una arquitectura en cambio SOAP es un protocolo que necesita un contrato(WDSL).
Rest es mas sencillo de usar.
El codigo es declarativo.
Mejor gestion de la complejidad.
Integracion de la API funcional.
Uso potencial del paralelismo.

Que es una interfaz funcional

Interface con solo un metodo abstract,puede tener metodos default.
Con @FunctionalInterface, el compilador valida que es una funcional interface.

SCRUM

Los roles en SCRUM son :
Product Owner:Es la representacion del cliente dentro del equipo de trabajo.responsable de expresar claramente la necesida del cliente dentro del Product Backlog.Define el Product Backlog
Scrum Master :Responsable de aseguarar que el SCRUM es entendido y realizado por el equipo. El equipo debe trabajar ajustandose a la teoria,practica y reglas de SCRUM
El Team Developer :Responsables de dar cumplimiento a los SPRINT.
Reunion Sprint Planning Meeting. Resultado de esta reunion se tiene el Sprint Backlog,son conjuntos de requisitos que se deben cumplir en una o 4 semanas.(Este tiempo es el SPRINT)
En el SPRINT intervienen el scrum master y team developer .
Los Daily son los seguimientos diarios no deben durar mas de 15 min.
Al final de un SPRINT se hace una reunion llamada sprint review(ScrumMaster,Team developer y product owner),se revisa la entrega o el producto incremental.
Despues de entregar el producto se hace una reunion Sprint Retrospective.Mejoras o problemas

Patrones de microservicios

Patron Saga: Es una secuencia de transacciones locales donde cada transacción actualiza información dentro de un servicio.

Patrones de diseno software

builder : un patrón de construcción para construir objetos. A veces, los objetos que creamos pueden ser complejos, estar formados por varios subobjetos.
Observer: Este patrón es una dependencia de uno a muchos entre objetos, de modo que cuando un objeto cambia de estado, se notifica a todos sus dependientes.
Singleto : se utiliza para limitar la creación de una clase a un solo objeto. Esto es beneficioso cuando se necesita un objeto (y solo uno) para coordinar acciones en todo el sistema.
Adapter :Esto permite que las clases incompatibles trabajen juntas al convertir la interfaz de una clase en otra. Piense en ello como una especie de traductor: cuando dos jefes de estado que no hablan un idioma común se encuentran, generalmente un intérprete se sienta entre los dos y traduce la conversación, lo que permite la comunicación.
Proxy : un proxy es un contenedor o un objeto de agente que el cliente está llamando para acceder al objeto de servicio.

OWASP

OWASP Security Framework, que es una aplicación web que explica los principios de codificación segura en múltiples lenguajes de programación y tiene como objetivo ayudar a los desarrolladores a crear aplicaciones seguras por diseño.

Tipos de arquitectura de Software

Arquitectura MVC, la programación por capas, peer-to-peer, orientada a servicios (SOA), cliente-servidor.

Arquitectura Limpia

Independientes del framework.
Testables.
Independientes de la UI
Independientes de la base de datos.
Independiente de agentes externos
Más tolerantes al cambio
. Reutilizables.
Mantenibles.

Arquitectura Hexagonal

La intención de la Arquitectura Hexagonal no es otra que permitir que una aplicación sea usada de la misma forma por usuarios, programas, pruebas automatizadas o scripts, y sea desarrollada y probada de forma aislada de sus eventuales dispositivos y bases de datos en tiempo de ejecución.
También conocida como arquitectura de puertos y adaptadores, tiene como principal motivación separar nuestra aplicación en distintas capas o regiones con su propia responsabilidad. De esta manera consigue desacoplar capas de nuestra aplicación permitiendo que evolucionen de manera aislada. Además, tener el sistema separado por responsabilidades nos facilitará la reutilización.

Arquitectura en Microservicios y EventSource

El patrón Event Sourcing es una técnica de diseño de software que consiste en almacenar un registro secuencial de todos los cambios que ocurren en el estado de una aplicación como una secuencia de eventos inmutables. Cada evento representa un cambio único y atómico en el estado de la aplicación y se almacena de manera secuencial en un registro de eventos, generalmente llamado "log de eventos.º "journal". Este enfoque permite reconstruir el estado actual de la aplicación en cualquier punto en el tiempo mediante la reproducción de todos los eventos desde el inicio hasta el momento deseado.

Ventajas de Event Sourcing Auditoría y seguimiento: proporciona un registro completo de todas las acciones y cambios realizados en la aplicación, lo que facilita la auditoría y el seguimiento. Reproducibilidad: permite reproducir el estado de la aplicación en cualquier momento y facilita la resolución de problemas y la depuración. Flexibilidad y evolución: permite agregar nuevas funcionalidades retrospectivamente y adaptarse a cambios en los requisitos del negocio mediante la introducción de nuevos tipos de eventos. Consistencia y atomicidad: los eventos son atómicos y se aplican de manera consistente, lo que garantiza la integridad del estado de la aplicación.

Desafíos del patrón Event Sourcing Complejidad de implementación: la implementación de Event Sourcing puede ser más compleja que otros enfoques tradicionales debido a la necesidad de gestionar el registro de eventos y la reconstrucción del estado de la aplicación. Escalabilidad del registro de eventos: el tamaño del registro de eventos puede crecer considerablemente con el tiempo, lo que puede plantear desafíos en términos de almacenamiento y rendimiento. Consistencia eventual: la reconstrucción del estado de la aplicación a partir de eventos puede llevar tiempo, lo que puede resultar en una consistencia eventual en lugar de una consistencia inmediata. Modelado de eventos: requiere un diseño cuidadoso del modelo de eventos para representar de manera precisa todos los cambios relevantes en el estado de la aplicación.

En resumen, el patrón Event Sourcing es una técnica poderosa para gestionar el estado de una aplicación de manera eficiente y escalable. Si se implementa correctamente, puede proporcionar una mayor transparencia, flexibilidad y capacidad de evolución en comparación con otros enfoques más tradicionales. Sin embargo, también presenta desafíos en términos de complejidad de implementación y escalabilidad del registro de eventos.

Azure Storage

Azure Storage es una solución de almacenamiento en la nube, proporcionada por Microsoft, que le permite almacenar y acceder a los datos en la nube. Ofrece un almacenamiento de alta disponibilidad y masivamente escalable, en forma de cinco servicios de almacenamiento diferentes: archivos, blobs, colas, tablas y discos.

Azure Key Vault

Es un servicio en la nube que proporciona un almacenamiento seguro de secretos. Puede almacenar de forma segura claves, contraseñas, certificados y otros secretos. Los almacenes de claves de Azure se pueden crear y administrar a través de Azure Portal.

Azure SQL Database

Está disponible como una base de datos única con su propio conjunto de recursos administrados a través de un servidor lógico y como una base de datos agrupada en un grupo elástico de , con un conjunto compartido de recursos administrados a través de un servidor lógico. Basado en el conocido motor SQL Server , Azure SQL le permite utilizar las herramientas, los lenguajes y los recursos que conoce. Azure SQL es un servicio de base de datos relacional totalmente administrado creado para la nube de Azure . Cree su próxima aplicación con la ayuda de una base de datos SQL completamente administrada con capacidades de inteligencia artificial integradas, escalado automático y copias de seguridad.

Azure Kubernetes

Azure Kubernetes Service (AKS) es un servicio de Kubernetes administrado de Microsoft Azure que tiene como objetivo simplificar la implementación y la administración de clústeres de Kubernetes . Kubernetes es responsable de trasladar y entregar las cajas de forma segura a las ubicaciones en las que puedan usarse. Kubernetes es un software de orquestación de código abierto que proporciona una API para controlar cómo y dónde se ejecutarán esos contenedores . Le permite ejecutar sus contenedores y cargas de trabajo de Docker y lo ayuda a abordar algunas de las complejidades operativas al pasar a escalar múltiples contenedores, implementados en varios servidores.

Azure APIM

Azure API Management es una plataforma totalmente integrada que permite a las empresas crear, publicar, proteger y analizar las API. Al actuar como un portal, permite a las empresas controlar el acceso, supervisar el uso y proteger tus API de posibles amenazas. Azure API Management es una plataforma de gestión híbrida y multicloud para API en todos los entornos . Como plataforma como servicio, API Management admite el ciclo de vida completo de las API.

Azure Functions

Azure Functions, en su modo básico, es un servicio Serverless ofrecido por Microsoft con las siguientes características: Permiten ejecutar cargas de trabajo on-demand y en respuesta a eventos o event-driven: peticiones HTTP externas, eventos recibidos de otros servicios de Azure, alertas de monitorización, eventos programados con tareas cron, etc. Proporcionan escalado y desescalado de recursos elástico de forma automática, rápida y transparente al cliente consumidor. Permiten de dar soporte a picos de demanda, volviendo a estados anteriores cuando las peticiones decrecen. Típicamente ejecutadas en un escenario stateless o sin estado, es decir, entre diferentes ejecuciones las funciones no almacenan ni utilizan información de estado. Los entornos de ejecución se crean, se ponen a disposición del código del cliente y se destruyen. No hay posibilidad de almacenar y recuperar información entre peticiones. Cuanto más rápido termine la función, menor es el coste. Por tanto, encajan muy bien en entornos en los que el código de las funciones es pequeño, específico (single-purpose) y de corta duración. El proveedor Cloud necesita un tiempo para provisionar los recursos antes de que la función se ejecute. Se denomina cold-start. Este tipo de servicio se conoce también como FaaS o Function as a Service.

Azure REDIS Cache

Azure Cache for Redis, un agente de mensajería y memoria caché de datos segura que proporciona a las aplicaciones un acceso de alto rendimiento y baja latencia a los datos.

Patrones de diseno Factory

Patron creacional factory permite desacoplar la lógica de creación de forma centralizada. Responsable de crear objetos evitando exponer la lógica de instanciación al cliente. proporciona una interfaz para crear objetos en una superclase, mientras permite a las subclases alterar el tipo de objetos que se crearán.

Patrones microservicios saga

Secuencia de transacciones que actualiza cada servicio y publica un mensaje o evento para desencadenar el siguiente paso de la transacción. Si se produce un error en un paso, la saga ejecuta transacciones de compensación que contrarrestan las transacciones anteriores.

Microservicios Patron de Orquestacion y coreografia

El patrón de coreografía en la arquitectura de microservicios es un enfoque descentralizado para coordinar la interacción entre servicios.En lugar de depender de un orquestador central que controla el flujo de trabajo, cada microservicio colabora con otros emitiendo y respondiendo a eventos. Con la orquestación, la lógica de control reside en un controlador central, mientras que con la coreografía, las interacciones están descentralizadas y cada componente es responsable de gestionar su parte del flujo de trabajo.

Microservicios Patrones

Database per Microservice
Event Sourcing
CQRS
BFF
API Gateway
Strangler.
Circuit Breaker
Externalized Configuration
Consumer-Driven Contract Tracing

DDD

El Domain-Driven Design (DDD) no es una tecnología ni una metodología, sino una práctica de desarrollo de software con necesidades complejas, que sitúa el Dominio del Negocio como faro del proyecto y en su Modelo, como herramienta de comunicación entre negocio y tecnología.

Comandos basicos Kubernetes

La siguiente lista es un recopilatorio de comandos basicos para poder obtener la información que necesitamos acerca de lo que ocurre en nuestro cluster de kubernetes:
información del cluter kubectl cluster-info

lista de los nodos del cluster
kubectl get nodes

lista de los servicios
kubectl get service

lista de los pods
kubectl get pods

lista de deployments
kubectl get deployments

lista de namespaces
kubectl get namespaces

lista de los pods del namespace prueba
kubectl get pods -n prueba

exponer un deployment
kubectl expose deployment first-deployment –port=80 –type=NodePort

información detallada del pod apache1
kubectl describe pod apache1

eliminar servicio
kubectl delete service hello-world

eliminar deployment
kubectl delete deployment hello-world

escalar a 3 replicas un deployment
kubectl scale –replicas=3 deployment prestashop -n prestashop

acceder al pod ubuntu-test
kubectl –namespace=enmilocalfunciona exec -it ubuntu-test bash

crear un secret
kubectl create secret generic mysql-pass –from-literal=password=mypassword

aplicar el contenido del fichero deployment.yaml
kubectl apply -f deployment.yaml

listar los tokens
kubeadm token list

agregar nodo al cluster
kubeadm join –discovery-token-unsafe-skip-ca-verification –token=102952.1a7dd4cc8d1f4cc5 172.17.0.52:6443

cosmoDB
Azure VM
ciclo de vida de pod services, comandos en pods, usando la CLI de Azure, troubleshooting
API Gateway
IntegraciOn y despliegue continuo con herramientas: (Jenkins y Kubernetes)
Junit 5 y Mockito : Uso de Asserts, anotaciones Mock, Test, InjectMock, webClientTest
Clean Code y Principios Solid
Key cloak
Kubernetes docker
TDD Karate
RxJava
xsd y definicion wsdl, xquery
Eureka api getway Zeull
Circuit Breaker
Programacion reactiva como lo has usado,dependencias
Webflux
MongoDB
Monitoreo Kibana,prometheus,grafena,dinatrace,new Relic
git privacy
Jenkins, Logstash, Kibana, Fluentd
ORM