Arquitectura AWS 2

Clean Arquitecture 10

Conocimientos DDD 15

Programación reactiva 18

Principios SOLID 24

DevOps CI 28

DevOps CD 30

DevOps CM. 32

DevOps CT 34

Docker y contenedores 38

Seguridad en el desarrollo de SW 42

Comunicación 45

Arquitectura AWS 20 preguntas

1. Una empresa necesita implementar instancias EC2 para manejar el procesamiento por lotes durante la noche. Esto incluye la transcodificación de medios y algunas transcripciones de voz a texto.
Este no es un trabajo de alta prioridad y está bien si estas ejecuciones por lotes se interrumpen.
¿Cuál es la mejor opción de compra de instancias EC2 para este trabajo?

- Reservado
- Spot
- Hosts dedicados
- Bajo demanda

2. Su organización utiliza AWS CodeDeploy para implementaciones. Ahora está iniciando un proyecto en la plataforma AWS Lambda. Para sus implementaciones, se le ha dado el requisito de realizar implementaciones blue-green.
Cuando realiza implementaciones, desea dividir el tráfico, enviando un pequeño porcentaje del tráfico a la nueva versión de su aplicación.
¿Qué configuración de implementación permitirá esta división del tráfico?

- All at Once
- Lineal (Linear)
- Canary
- Enrutamiento ponderado (Weighted routing)

3. Está administrando el almacenamiento de datos para su empresa y hay muchos volúmenes de EBS. Sus líderes le han dado algunos requisitos nuevos.
Es necesario monitorear ciertas métricas en los volúmenes de EBS, y el equipo de bases de datos debe recibir una notificación por correo electrónico cuando se superen ciertos umbrales.
¿Qué servicios de AWS se pueden configurar para cumplir con estos requisitos?

SNS	SES	SWF	CloudWatch	SQS

4. Su empresa ha pasado por una auditoría enfocada en el almacenamiento de datos. Actualmente está almacenando datos históricos en Amazon Glacier.
Uno de los resultados de la auditoría es que se debe poder acceder de forma inmediata y por solicitud a una parte de esos datos históricos a los que se accede con poca frecuencia.
¿Dónde puede almacenar estos datos para cumplir con este requisito?

- Deje los datos a los que se accede con poca frecuencia en Glacier.
- Almacenar los datos en EBS
- S3 estándar-IA
- Estándar S3

5. Usted trabaja como arquitecto líder de AWS para un gran proveedor de atención médica. Hay una necesidad de recopilar datos en tiempo real de los dispositivos de toda la organización.
Los datos incluirán registros de logs y eventos de fuentes como servidores, computadoras de escritorio y dispositivos móviles. Los datos capturados inicialmente serán datos técnicos, pero el objetivo es expandir el esfuerzo para recopilar datos clínicos en tiempo real de dispositivos portátiles utilizados por enfermeras y médicos.
¿Qué servicio de AWS cumple mejor con este requisito?

- Kinesis Video Streams
- Kinesis Data Streams
- AWS Lambda
- AWS Redshift

6.

Una nueva empresa emergente decide utilizar AWS para alojar su aplicación web. Configuran una VPC y dos subredes dentro de la VPC. También conectan un internet gateway a la VPC.
En la primera subred, crean una instancia EC2 para alojar una aplicación web. Hay una ACL de red y un grupo de seguridad, que tienen la entrada y salida adecuadas hacia y desde Internet. Hay una ruta en la tabla de enrutamiento hacia el internet gateway.
Las instancias EC2 agregadas a la subred deben tener una dirección IP única a nivel mundial para garantizar el acceso a Internet.
¿Cuál no es una dirección IP única a nivel mundial?

- Dirección IP privada
- Dirección IPv6
- Dirección IP elástica
- Dirección IP pública

7.

Una empresa utilizará varias instancias EC2 para alojar varias aplicaciones de referencia. Se espera que las aplicaciones reciban un tráfico constante y relativamente bajo.
Se espera que estas aplicaciones funcionen durante 3 años, momento en el que se evaluarán las aplicaciones para su actualización.
¿Qué tipo de EC2 cumplirá con este requisito considerando el costo como un factor adicional?

- On-Demand
- Reserved
- Spot
- Dedicated Hosts

8.

Su empresa ha realizado un simulacro de recuperación ante desastres que no cumplió con el objetivo de tiempo de recuperación (RTO) deseado por la dirección ejecutiva.
La falla se debió en gran parte a la cantidad de tiempo necesario para restaurar el funcionamiento adecuado de la base de datos. Usted le ha dado a la gerencia una recomendación para implementar la replicación de datos síncrona para la base de datos RDS para ayudar a cumplir con el RTO.
¿Cuál de estas opciones puede realizar la replicación de datos sincrónica en RDS?

- AWS Database Migration
- DAX
- RDS Multi-AZ
- Leer réplicas

9. Una empresa internacional tiene muchos clientes en todo el mundo. Estos clientes necesitan transferir gigabytes a terabytes de datos de forma rápida y periódica a un bucket de S3.
¿Qué función de S3 permitirá estas transferencias de datos a larga distancia de una manera segura y rápida?

- Transfer Acceleration
- Multipart upload
- AWS SnowMobile
- Cross-account replication

10. Después de una reunión del Comité Directivo de TI, se le asignó la responsabilidad de configurar un entorno híbrido para los recursos informáticos de la empresa. Usted analiza los pros y los contras de varias tecnologías en función de los requisitos que se le dan. Los principales requisitos para impulsar esta selección son consideraciones de costos generales y la capacidad de reutilizar las conexiones de Internet existentes.
¿Qué tecnología cumple mejor con estos requisitos?

- Emparejamiento de VPC (VPC Peering)
- AWS VPN
- AWS Direct Connect
- AWS Direct Gateway

11. Una pequeña startup está comenzando a configurar IAM para su organización. Se han creado los inicios de sesión de los usuarios y ahora el enfoque serán los permisos para para esos usuarios.
Un administrador comienza a crear políticas basadas en identidad.
¿A qué elemento no se puede adjuntar una política basada en identidad?

- roles
- usuarios
- Recursos
- grupos

12.

Usted trabaja para una gran institución financiera y se prepara para la recuperación ante desastres y los próximos simulacros de recuperación ante desastres.
Un componente clave del plan de recuperación ante desastres serán las instancias de la base de datos y sus datos. Un objetivo de tiempo de recuperación (RTO) agresivo dicta que la base de datos debe replicarse sincrónicamente.
¿Qué configuración puede cumplir con este requisito?

- AWS Lambda para desencadenar una plantilla de CloudFormation en otra región.
- RDS Multi-AZ
- RDS read replicas
- RDS Multi-Region

13.

Ha estado evaluando las NACL en su empresa. La mayoría de las NACL se configuran de la misma manera:
100 All Traffic Allow
200 All Traffic Deny
* All Traffic Deny
Si entra una solicitud, ¿cómo se evaluará?

- Se permitirá la solicitud.
- Se utilizará la regla con el número más alto, una negación.
- Se evaluarán todas las reglas y el resultado final será denegar.
- El valor predeterminado denegará el tráfico.

14.

Una liga de béisbol profesional ha optado por utilizar una base de datos de documentos y valores clave para el almacenamiento, el procesamiento y la entrega de datos.
Muchos de los requisitos de datos implican un procesamiento de datos de alta velocidad, como un sistema de radar Doppler que muestrea la posición de la pelota de béisbol 2000 veces por segundo.
¿Qué almacenamiento de datos de AWS puede cumplir con estos requisitos?

- S3
- Redshift
- RDS
- DynamoDB

15. Usted ha asumido la gestión de varias instancias en la empresa AWS. Usted desea revisar rápidamente los scripts utilizados para iniciar las instancias en tiempo de ejecución.
Se puede utilizar un comando de URL para hacer esto.
¿Qué puede agregar a la URL http://169.254.169.254/latest/ para traer estos datos?

- meta-data/
- instance-demographic-data/
- instance-data/
- user-data/

16. Una nueva pequeña empresa ha comenzado a utilizar AWS para toda su infraestructura de TI. La empresa tiene un arquitecto de soluciones de AWS y las demandas de su tiempo son abrumadoras.
El equipo de software ha recibido permiso para implementar sus aplicaciones Python y PHP por su cuenta. A ellos les gustaría implementar estas aplicaciones sin tener que preocuparse por la infraestructura subyacente.
¿Qué servicio de AWS usarían para las implementaciones?

- CloudFormation
- Elastic Beanstalk
- CloudFront
- CodeDeploy

17. Una compañía de seguros está creando una aplicación que realizará análisis casi en tiempo real en enormes conjuntos de datos en el rango de terabytes y potencialmente incluso petabytes.
La empresa está evaluando una opción de almacenamiento de datos de AWS.
¿Qué servicio de AWS permitirá el almacenamiento de datos a escala de petabytes y también permitirá la consulta rápida de estos datos?

- Redshift
- RDS
- DynamoDb
- ElastiCache

18.

Se ha unido a una empresa de software recién formada como arquitecto de soluciones. Es una empresa pequeña y usted es el único empleado con experiencia en AWS.
El propietario le ha solicitado sus recomendaciones para asegurarse de que los recursos de AWS se implementen para mantenerse de manera proactiva dentro del presupuesto.
¿Qué servicio de AWS puede utilizar para asegurarse de no tener sobrecostos para sus recursos de AWS?

- Inspector
- Billing and Cost Management
- Cost Explorer
- AWS Budgets

19.

Una empresa de juegos está diseñando varios juegos nuevos que se centran en gran medida en la interacción jugador-juego. El jugador realiza un movimiento determinado y el juego tiene que reaccionar muy rápidamente para cambiar el entorno en función de ese movimiento y presentar la próxima decisión para el jugador en tiempo real.
Se necesita una herramienta para recopilar continuamente datos sobre las interacciones entre el jugador y el juego y alimentar los datos a la plataforma de juego en tiempo real.
¿Qué servicio de AWS puede satisfacer mejor esta necesidad?

- AWS IoT
- AWS Lambda
- Kinesis Data Analytics
- Kinesis Data Streams

20.

Una empresa de software ha producido varias aplicaciones sin estado. El equipo de software ahora necesita servidores que actúen como servidores de prueba para estas aplicaciones.
Estos servidores no son de misión crítica, por lo que el tiempo de inactividad ocasional es aceptable.
¿Qué tipo de servidores EC2 se pueden utilizar para cumplir con estos requisitos al menor costo?

- Hosts dedicados
- Reservado
- Spot
- Bajo demanda

Clean Arquitecture 10 preguntas

21. **En el contexto de Arquitectura de Software, ¿Qué es la complejidad esencial?**

- Es la cualidad de lo que está compuesto de diversos elementos interrelacionados. La complejidad tiende a ser utilizada para caracterizar un conjunto intrincado y difícil de comprender.
- Es aquella que es inherente al problema que queremos solucionar, refleja la intención y el propósito por el cual fue creado el sistema y está expresada en los términos del dominio del problema.
- Es aquella que existe y es inyectada por la solución misma, es decir por los detalles y los medios por los cuales se materializa la solución, es altamente variable y depende en gran medida de detalles y decisiones tomadas al momento de crear la solución.
- Ninguna de las anteriores.

(en medium)

22. **¿Cuál de las siguientes frases es cierta con respecto al concepto de Clean Architecture de Robert C. Martin?**

- Es una evolución del concepto de "Space based architectures" de Neil Ford el cual se base en la aplicación del principio de sustitución de Liskov.
- Hace parte de lo que son las arquitecturas centradas en el dominio como lo son la "arquitectura hexagonal" y "Ports and Adapters".
- Es la base conceptual del modelo de actores de Carl Hewitt y por lo tanto la base de los sistemas Reactivos modernos.
- Dió origen al concepto de Reactive Streams y Programación Reactiva al implementar las bases del principio de inyección de dependencias.

23. **¿Cuál de las siguientes se podrían considerar un beneficio de una arquitectura centrada en el dominio, cómo la arquitectura limpia o hexagonal?**

- Proteger y aislar las reglas del negocio de un entorno cambiante como lo es la tecnología que la rodea.
- Generación automática del código basado en los diagramas o modelos de dominio.
- Uso de herramientas de modelado de dominio para realizar ingeniería inversa de sistemas complejos.
- Minimizar el uso de recursos informáticos para hacer un sistema resiliente.
- 2 y 3 son correctas.

24. **¿Cuál es el objetivo de una arquitectura centrada en el dominio?**

- Automatizar la generación de código a partir de modelos.
- Construir sistemas autónomos capaces de tolerar fallos.
- Proteger y preservar en el largo plazo las reglas y procesos definidos por el negocio de los cambios en los detalles tecnológicos.
- Todos los anteriores.

25. **Cuando se emplea inversión de dependencias, si se desea cambiar la implementación concreta de un componente:**

- Siempre se deben cambiar las interfaces para representar el cambio de la implementación y conservar la coherencia.
- Ninguno de los componentes que usan (indirectamente) la dependencia se ven afectados ya que dependen de la abstracción.
- Se deben cambiar siempre los componentes que usan la dependencia sin importar como la usen.
- La 1 y la 3 son correctas.

26. **¿Cuál de los siguientes describe mejor el patrón de diseño "The HumbleObject pattern"?**

- Este patrón le permite a los "unit tests" a separar las funcionalidades que son difíciles de probar de las que son fáciles de probar.
- Este patrón permite establecer las prioridades de pruebas de acuerdo a los casos de uso que contenga el sistema.
- Este patrón permite unir las casos de prueba en una sola carga de trabajo, sin importar la dificultad de la prueba.
- Todos los anteriores.

27.

En el contexto de Arquitectura de Software, ¿Qué es la complejidad accidental?

- Es la cualidad de lo que está compuesto de diversos elementos interrelacionados. La complejidad tiende a ser utilizada para caracterizar un conjunto intrincado y difícil de comprender.
- Es aquella que es inherente al problema que queremos solucionar, refleja la intención y el propósito por el cual fue creado el sistema y está expresada en los términos del dominio del problema.
- Es aquella que existe y es inyectada por la solución misma, es decir por los detalles y los medios por los cuales se materializa la solución, es altamente variable y depende en gran medida de detalles y decisiones tomadas al momento de crear la solución.
- Ninguna de las anteriores.

(en medium)

28.

(Los/la/el) _____ nos permite(n) contener el impacto del cambio al no depender de los detalles, invirtiendo el flujo de control y la dirección de dependencia.

- Inyección de dependencias
- Inversión de dependencias
- Controladores e inversores de flujo
- Control de impactos basado en polimorfismo

29.

En un sistema construido empleando arquitectura limpia, cuando se desea cambiar la tecnología de acceso a datos:

- Es necesario cambiar las entidades de dominio ya que en éstas se encuentras las anotaciones de mapeo de datos del framework usado
- No es necesario cambiar las entidades de dominio si el cambio es netamente tecnológico ya que no existe dependencia directa del dominio con la tecnología de acceso a datos.
- El cambio debe empezar desde los Entry-Points definidos y posteriormente modificando las interfaces segregadas del dominio
- El cambio de la base de datos representa un costo muy alto ya que la arquitectura limpia se genera a partir de los modelos entidad relación de la base de datos, por ende requiere una regeneración del proyecto.

30.

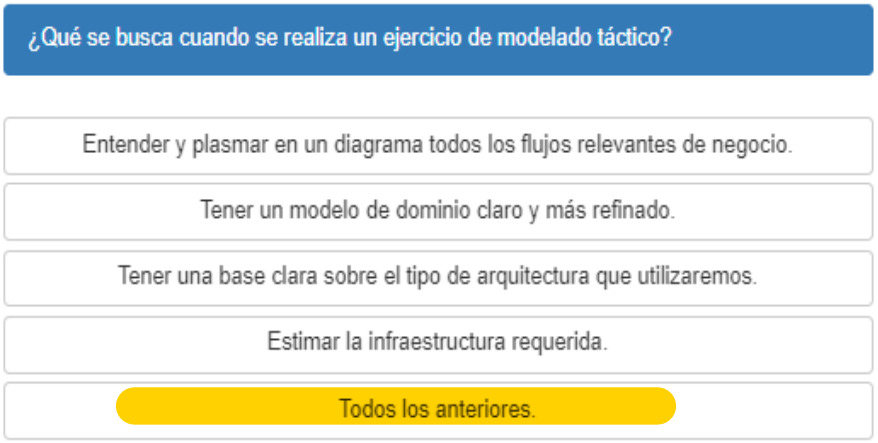
"Representan los datos y objetos de valor del negocio sin llegar a ser un módulo compuesto sólo de estructuras de datos u objetos anémicos (sin comportamiento); sino que por el contrario es donde se recomienda expresar la lógica de negocio crítica a través de funciones puras (sin efectos secundarios)."
¿Cuál de los siguientes se relaciona con la definición anterior?

- Casos de uso
- Controladores
- Dispositivos externos
- Entidades
- Ninguno de los anteriores.

(en medium)

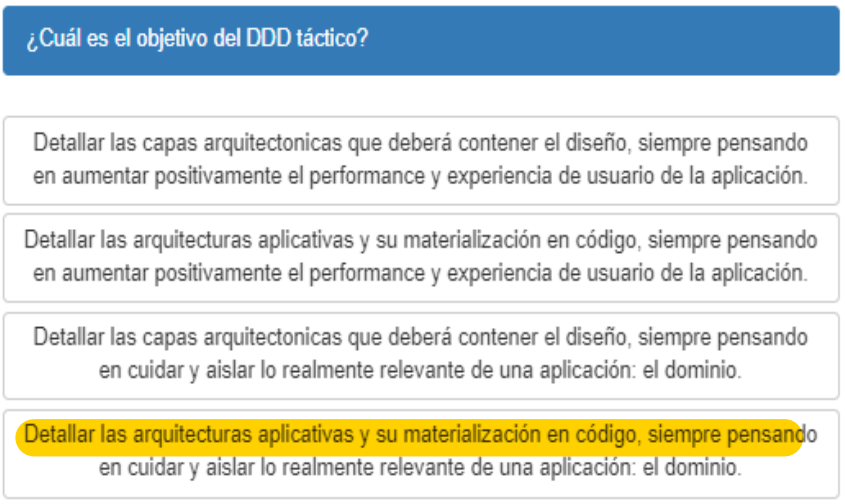
Conocimientos DDD 9 preguntas

31.



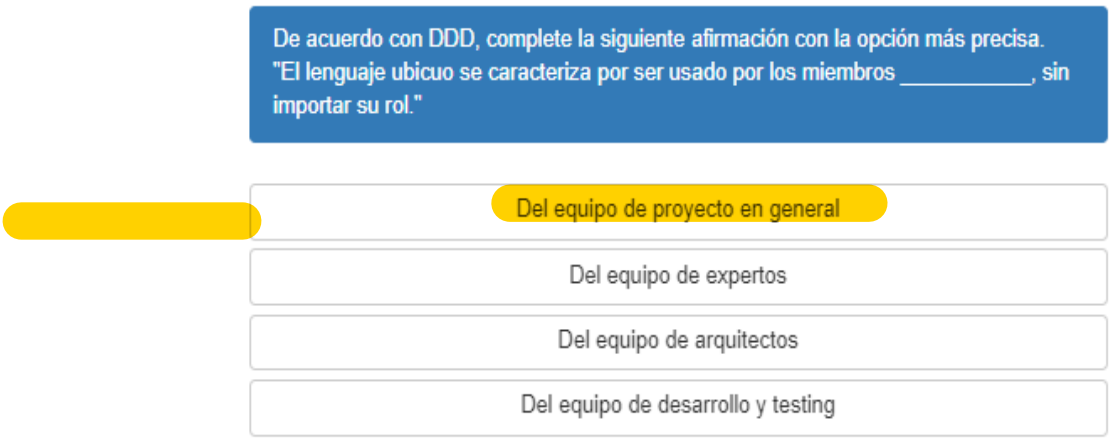
(en medium)

32.



(Modelado en medium)

33.



(ubicuo medium)

34.

Cuando hablamos de microservicios como estilo arquitectónico, podemos decir:

- Los microservicios separan las responsabilidades de las capas técnicas de la aplicación, es decir, debemos tener microservicios de orquestación, persistencia y logging, cada uno siendo una unidad desplegable independiente.
- Los microservicios representan unidades funcionales con la mayor autonomía posible, representando idealmente contextos delimitados identificados mediante Domain Driven Design.
- Los microservicios deben contener una sola funcionalidad expuesta, y deben contener cuanto mucho una sola entidad de negocio.
- La 1 y 3 son verdaderas.

(contexto en medium)

35.

¿Quién debería participar de las sesiones de modelado estratégico?

- Arquitecto Empresarial
- Expertos de dominio (negocio)
- Líderes de tecnología involucrados
- Todos los anteriores

(contexto en medium)

36.

En DDD, es un límite específico dentro del cual un modelo de dominio existe. Dentro de este límite todos los términos y frases del lenguaje ubicuo tienen un significado específico.

- Bounded Map
- Bounded Context
- Subdomain Context
- Context Map

37.

¿Quiénes deberían necesariamente participar de las sesiones de modelado táctico?

- Expertos de dominio (negocio) y Arquitecto de la solución
- Equipo implementador de la solución y Expertos de dominio (negocio)
- Equipo implementador de la solución y Arquitecto de la solución
- Usuarios del negocio y equipo de pruebas especializadas

(en medium)

38.

EventStorming es un taller en el contexto de DDD en el cual:

- Se realiza una identificación de los objetos y las clases del sistema partiendo de sus métodos y valores de retorno.
- Se identifican de forma colaborativa los eventos desde el punto de vista de las lineas de tiempo del negocio y posteriormente se agrupan y modelan tanto comandos como contextos delimitados.
- Se realiza la estructuración del proyecto por parte del equipo de arquitectura de forma exclusiva.
- Se elige el stack tecnológico de la solución con base en los lineamientos de ingeniería.

(Event Storming medium)

39.

En DDD, se debe capturar el terreno existente. Aquí se debe describir el presente y no el futuro imaginado.

- Bounded Map
- Bounded Context
- Subdomain Context
- Context Map

Programación reactiva 14 preguntas

40. **¿Qué es la escalabilidad en un sistema reactivo?**

- Es la capacidad de un sistema para hacer uso de más recursos informáticos con el fin de aumentar su desempeño
- Es la capacidad de un sistema de ampliar los recursos de maquina con el fin de aumento del desempeño
- Es la capacidad de un sistema de no ejecutar cambios para mantener la respuesta optima
- Es la caracteristica que adquiere un sitema cuando no tiene la flexibilidad necesaria

41. **En relación al Modelo de Actores se puede decir que:**

- Un actor es la unidad básica de computo para representar/analizar sistemas sistemas concurrentes y/o orientados a mensajes.
- Un actor es una entidad externa al sistema concurrente que normalmente representa un usuario o sistema del que depende.
- El conjunto de actores de un sistema representa los distintos roles de los usuarios externos del sistema o sistemas de los cuales depende.
- La 2 y 3 son correctas.

42. **Según las definiciones del manifiesto de los sistemas reactivos, el medio que habilita los valores de dichos sistemas es:**

- La programación funcional y el uso de reactive streams en todos los componentes.
- La orientación a mensajes (asincronismo)
- El uso de contenedores y orquestadores como kubernetes en entornos cloud native.
- El concepto de microservicios con DDD junto con infraestructura serverless.

43.

Con respecto a los términos Programación reactiva y Sistemas reactivos podemos decir:

- Son equivalentes y hacen referencia al mismo concepto.
- La programación reactiva hace referencia a un estilo de programación principalmente basado en la manipulación de flujos de datos asíncronos mediante patrones que vienen de la programación funcional.
- Cuando aplicamos programación reactiva el sistema resultante se comporta de forma proactiva ante los eventos ocurridos y no puede considerarse reactivo.
- Un sistema reactivo debe estar construido netamente con programación funcional pura.

44.

En lo referente a la programación funcional, la inmutabilidad:

- No debe usarse al interior de las funciones puras ya que éstas modifican siempre sus parámetros
- Es una propiedad esencial y deseable de las estructuras de datos en lenguajes funcionales
- Solo debe aplicarse a datos estáticos como propiedades de configuración y variables de entorno.
- La 1 y 3 son verdaderas.

45. ¿Cuál de estas se considera una caracteristica de un sistema reactivo?

1. Debe poder hacer uso de patrones de comunicación asíncrona y orientada a mensajes de forma optima y eficiente

2. No debe adaptarse de forma elástica ante el aumento del volumen concurrente.

3. El Sistema debe ser capaz de manejar un alto volumen de usuarios y tareas concurrentes de forma óptima y eficiente.

4. 1 y 3 son correctas

46. ¿Cuáles son las condiciones puras que deben cumplir las funciones puras?

Dados los argumentos siempre retorna datos diferentes y no tiene efectos secundarios

Dados los mismos argumentos siempre retorna lo mismo y tiene efectos secundarios

Dados los mismos argumentos siempre retorna lo mismo y no tiene efectos secundarios

Ninguno de los anteriores.

47. **Cuando hablamos de sistemas reactivos estamos hablando de:**

- Sistemas distribuidos con énfasis en alta tolerancia a fallos, elasticidad, alta eficiencia y responsividad, es decir sistemas distribuidos basados en principios solidos de ciencias de la computación.
- Sistemas que se consideran lo contrario a sistemas proactivos, es decir no consumen recursos hasta la ocurrencia de algún evento disparador, presentando un modelo de costos basado en el uso.
- Cualquier construido con programación funcional que haga uso de Reactor como librería de control de flujos.
- Sistemas construidos usando como base la arquitectura LMAX Disruptor son los únicos que pueden ser considerados reactivos.

48. **En referencia a la programación reactiva en Java se puede decir que:**

- Hace un uso más eficiente de los recursos al usar IO no bloqueante para permitir que cada Thread atienda más de una petición de forma concurrente.
- Requiere que se haga uso de Spring Native y de esta forma acelerar el tiempo de inicio de las aplicaciones.
- Requiere que se use el modelo de Threads and Locks para poder asignar un Thread del SO a cada petición que llegue y así tener alta escalabilidad.
- La 1 y 2 son verdaderas.

49. **¿Cuáles son algunos de los beneficios de los sistemas reactivos?**

- Alta responsividad, alta escalabilidad/elasticidad y tolerancia a fallos (resiliencia).
- Simplicidad en el desarrollo, velocidad en desarrollo e interoperabilidad con mensajería empresarial.
- Low code, mapeo automático de entidades de dominio a contenedores funcionales a través del tooling del ecosistema.
- Ninguna de las anteriores.

50. ¿Cuándo se debe evaluar como primera opción el uso de programación reactiva?

- Hay un alto énfasis en orquestación y coordinación de flujos y acciones que involucran I/O o llamados remotos a otros componentes o sistemas
- Se cuentan con los drivers o adaptadores reactivos necesarios para tener un flujo reactivo end2end.
- Es un componente que manejará un alto nivel de concurrencia.
- Todas las anteriores

51. La tolerancia a fallos en sistemas inspirados en Erlang y/o en el modelo de actores (Erlang/Elixir, Akka entre otros) se logra principalmente por:

- Las funciones puras y el manejo de excepciones sincronizadas en los bloques try… catch.
- El aislamiento fuerte de fallas entre procesos/actores y las capacidades de supervisión y reinicio de procesos/actores.
- El agendamiento de nuevos contenedores a travez de los eventos enviados a Kubernetes desde los supervisores de cada framework.
- La garantía de entrega de los mensajes enviados al mailbox de cada proceso/actor, los cuales son persistentes en DB por defecto.

52.

Cuando un sistema posee el modelo one-thread-per-request, esto implica:

- Que el nivel de escalabilidad del sistema es óptimo ya que al existir un Thread del SO por cada petición se garantiza independencia y máxima eficiencia según la Ley de Amdahl.
- Que no es un modelo óptimo en escenarios de alta concurrencia ya que en altos volúmenes tiene un efecto negativo dado el alto costo de mantener un Thread del SO por cada petición.
- Que es un prerrequisito esencial para lograr la escalabilidad lineal según la ley de escalabilidad universal de Gunther (USL)
- La 1 y 3 son verdaderas.

53.

Con respecto a el modelo de concurrencia tradicional Threads, Locks y memoria compartidas se puede decir:

- Es más fácil de programar escenarios concurrentes complejos y garantiza que no se presentarán race-conditions, pero presenta un performance inferior.
- Es un modelo de concurrencia de bajo nivel donde se puede tener un control muy granular de la interacción del procesador y la memoria, pero no es el nivel de abstracción adecuado para la mayoría de las aplicaciones.
- Es difícil programar escenarios concurrentes complejos y se pueden cometer fácilmente errores que lleven a race-conditions, deadlocks, starvation y en general problemas difíciles de depurar.
- La 2 y 3 son verdaderas.

Principios SOLID 8 preguntas

54. **¿Cuál definición corresponde al Principio de abierto/cerrado (Open/closed principle)?**

- Noción de que un sistema debe tener claramente definida un API expuesta (Abierta) y una API privada (Cerrada).
- Noción de que las unidades de software deben estar abiertas para su extensión, pero cerradas para su modificación.
- Noción de que las unidades de software deben implementar interfaces abiertas o públicas a través de implementaciones principalmente cerradas o privadas.
- Noción de que muchas interfaces cliente específicas son mejores que una interfaz de propósito general.
- Noción de que se debe "depender de abstracciones, no depender de implementaciones"

55. **¿Cuál definición corresponde al Principio de responsabilidad única (Single responsibility principle)?**

- Noción de que debe usarse un punto de entrada único y gobernado para acceder a las APIs del sistema.
- Noción de que las variables en un programa deben ser usadas para un sólo fin, y con esto aumentar la mantenibilidad
- Noción de que los "objetos de un programa deberían ser reemplazables por instancias de sus subtipos sin alterar el correcto funcionamiento del programa".
- Noción de que una unidad de software solo debería tener una sola razón para cambiar.
- Noción de que una función debe tener un único punto de retorno, evitando poner dicha responsabilidad en diferentes partes del programa.

56. **¿Cuál no es un principio SOLID?**

- Single responsibility
- Open Close
- Liskov substitution
- Interface Segregation
- Dependency Inversion
- Inversion of control

57. Según este principio "Las interfaces deben ser específicas para un tipo de cliente".

- Corresponde al principio DIP de SOLID.
- Corresponde al principio LSP de SOLID.
- Corresponde al principio OCP de SOLID.
- Corresponde al principio de segregación funcional generalizada de SOLID.
- Ninguna de las anteriores.

(referencia)

58. ¿Cuál definición corresponde al Principio de segregación de la interfaz (Interface segregation principle))?

- Noción de que se deben sustituir los métodos públicos por definiciones de interfaces y cambiar la dependencia directa.
- Noción de que la capa de APIs del sistema debe ser implementada de forma aislada en un componente no dependiente.
- Noción de que los objetos de un programa deberían ser reemplazables por instancias de sus subtipos sin alterar el correcto funcionamiento del programa.
- Noción de que muchas interfaces cliente específicas son mejores que una interfaz de propósito general.
- Noción de que se debe depender de abstracciones, no depender de implementaciones.

(referencia)

59. **Según este principio "las clases derivadas deben poder sustituirse por sus clases base".**

- Es considerada una buena practica de ingeniería de software pero no hace parte de los principios SOLID.
- Corresponde al principio LSP de SOLID
- Es una practica de extensibilidad enmarcada en los principios GRASP.
- Corresponde al principio de sustitución funcional generalizada de GRASP.
- Ninguna de las anteriores.

60. **Según este principio "una clase debería tener una, y solo una, razón para cambiar". Hace referencia a:**

- Corresponde al principio SRP de SOLID.
- Es considerada una buena practica de ingeniería de software pero no hace parte de los principios SOLID.
- Corresponde al principio OCP de SOLID.
- Es una practica de extensibilidad enmarcada en los principios GRASP pero no en SOLID.
- Ninguna de las anteriores.

61.

¿Cuál definición corresponde alPrincipio de inversión de la dependencia (Dependency inversion principle)?

- Noción de que un objeto solo debería tener una única responsabilidad
- Noción de que las “entidades de software deben estar abiertas para su extensión, pero cerradas para su modificación”.
- Noción de que los “objetos de un programa deberían ser reemplazables por instancias de sus subtipos sin alterar el correcto funcionamiento del programa”.
- Noción de que “muchas interfaces cliente específicas son mejores que una interfaz de propósito genera.
- Noción de que se debe “depender de abstracciones, no depender de implementaciones”

DevOps CI 4 preguntas

62. Un equipo tiene la responsabilidad de desarrollar una nueva aplicación para pagos virtuales. Te han consultado cuántos pipelines se deben crear si han validado que una estrategia de mono-repositorio es aplicable al proyecto y tendrán 8 microservicios inicialmente. ¿Cuál sería tu respuesta?

- Se debe construir un mono-pipeline para la integración de todos los componentes del repositorio.
- Se deben crear 8 definiciones de pipeline que correspondan a cada componente de la aplicación de forma independiente.
- Se deben crear 4 definiciones de pipeline para todos los componentes de la aplicación. Uno para la fase de build, el segundo para la fase de pruebas , el tercero para la fase de análisis y un cuarto para la fase de publicación.
- Se deben crear 2 definiciones de pipeline, las cuales serán nombradas CI Y CD.

63. ¿Cuál de las siguientes NO es una práctica de CI?

- Ejecutar compilación rápida.
- Desplegar a producción.
- Construcción de escenarios.
- Commits frecuentes.

64. ¿Cuál es el primer análisis de código que debe ser ejecutado?

- Style check.
- Cobertura de código.
- Pruebas unitarias.
- Ninguno de los anteriores.

65. ¿Cuál es el objetivo clave de la integración continua ?

- Encontrar y arreglar errores con mayor rapidez, mejorar la calidad del software y reducir el tiempo que se tarda en validar y publicar nuevas actualizaciones de software.
- Compilar el código fuente lo más rápido posible.
- Automatizar la forma de ejecutar las pruebas unitarias.
- Gestionar las versiones del código fuente a través de procesos automáticos que conectan herramientas de despliegue y ejecución de pruebas.

DevOps CD 4 preguntas

66. **¿Cuál de las siguientes afirmación es la acertada con respecto al continuos Deployment y las buenas practicas?**

- Los despliegues a ambientes productivos deben realizarse solo en sistemas operativos Linux.
- Los pipeline deben tener un promedio de ejecución de 5 a 10 minutos.
- Se deben realizar análisis de seguridad en todos los stage del pipeline de release para evitar la fuga de información.
- Tanto las recetas de despliegue y los ambientes deben estar homologados para contar con despliegues predecibles, mantenibles y repetibles.

67. **Seleccione las afirmaciones correctas entre Entrega Continua(Continuous Delivery) e Implementación Continua(Continuous Deployment). Seleccione 3 opciones.**

- En la Entrega Continua (Continuous Delivery), todos los cambios en el código solo se implementan en un entorno de prueba. En la Implementación Continua las revisiones se implementan en un entorno de producción.
- La Entrega Continua (Continuous Delivery), es una práctica de desarrollo de software donde los cambios de código son construidos, probados y preparados automáticamente para su salida a producción y esta debe estar aprobada por el Product Owner.
- Para que exista Implementación Continua(Continuous Deployment), antes debe haber Entrega Continua(Continuous Delivery), y en cualquiera de los casos debe haber una buena Integración Continua(Continuous Integration).
- Entrega Continua (Continuous Delivery), es una práctica fundamental recomendada de DevOps en la que los desarrolladores fusionan con frecuencia los cambios de código en un repositorio central donde se ejecutan compilaciones y pruebas automatizadas.
- En la Implementación Continua(Continuous Deployment), la premisa es ir liberando gradualmente, y si algún usuario tiene algún problema, entonces se hace rollback y el feedback llega al equipo desarrollador para que corrija el problema.

68. **Un equipo es nuevo trabajando con un repositorio para componentes de SQL. ¿Cuál afirmación es correcta con respecto al uso del repositorio?**

- No es necesario hacer un code review en el pull request porque la herramienta realiza las validaciones del lenguaje.
- Se debe crear una rama principal con el nombre de sql-branch para indicar dónde se encuentran los fuentes.
- No se debe modificar el directorio "sql_code/archive". Realizar cambios sobre el directorio ocasiona errores en los depliegues.
- Los repositorios que versionan lenguajes como SQL, solo deben ser gestionados por los administradores de base de datos.

69. **Como desarrollador de un equipo usted fue asignado para construir un pipeline de release. Al momento de realizar su primer despliegue, el release es abandonado. ¿Cuál podría ser la causa para que el release sea abandonado ?**

- El release definition tiene tareas deshabilitadas y la política indica que no pueden haber tareas deshabilitadas en el pipeline.
- El release fue creado en un día con restricciones de cambios a producción.
- El release debe ser lanzado por el líder del equipo.
- El release id debe ser registrado en el work item de tipo DOD.

DevOps CM 5 preguntas

70.

¿Qué es un sistema de control de versiones?

- Son herramientas de software que ayudan a los equipos a gestionar los cambios en el código fuente a lo largo del tiempo.
- Son herramientas de software que ayudan a los equipos a encontrar las vulnerabilidades de seguridad e información sensible.
- Son herramientas que permiten almacenar binarios y dependencias en cualquier lenguaje de programación.
- Es una herramienta que permite generar versión en el pipeline de CI.

.

71.

¿En cuál proceso se involucra un administrador de dependencias en el ciclo de desarrollo y entrega de software?

- Durante el proceso de compilación del software.
- En el proceso de integración de cambios entre ramas.
- En la ejecución de las pruebas unitarias de los proyectos.
- Durante el proceso de certificación de los artefactos.

72.

¿Cuál de las siguientes afirmaciones corresponde a la definición de un repositorio virtual?

- Permiten agrupar un conjunto de repositorios remotos y locales. El objetivo es permitirle al equipo tener un único punto de descarga y publicación de dependencias.
- Permiten almacenar las dependencias de cada proyecto o producto. En estos repositorios también se definen los repositorios build (repositorios donde se almacenan los componentes o artefactos compilados, estos se publican desde el pipeline de CD para ser desplegados a través del pipeline de RM), de los cuales el banco posee el código fuente.
- Son repositorios que permiten conectarnos a repositorios externos. Estos repositorios por defecto ya se encuentran configurados.
- Permiten conectar repositorios externos e internos y las librerías que se cargan a través de ellos quedan publicadas en los repositorios internos de cada tecnología.

73. **En el equipo desean implementar un mecanismo que permita realizar la revisión de los cambios y que tenga una aprobación por parte de un miembro del equipo para garantizar la integración de los nuevos cambios en el repositorio. ¿Cuál práctica le recomendarías para cumplir con el requerimiento?**

- Continuous integration con una ejecución manual por parte de un miembro del equipo.
- Notificar a un miembro del equipo de los nuevos cambios e indicarle la rama que tiene el nuevo código fuente.
- Un practica de pair programming donde los miembros del equipo den el visto bueno del código de su paraje de trabajo.
- Un practica de Code Review apoyándose de pull request para la integración del código fuente entre ramas.

74. **¿Qué es una estrategia de branching en los repositorio de Git?**

- Son los trigger automáticos para la ejecución del pipeline cuando se realiza un nuevo commit.
- Es una serie de reglas y estándares de nombramiento para el uso de las ramas y las integraciones del código fuente en el repositorio.
- Son las políticas que valida los commit para cumplir con la calidad del código en los repositorios.
- Es la definición del los tipos y tamaños de los archivos en los repositorios.

DevOps CT 15 preguntas

75. ¿Cuáles de los siguientes describen los principios FIRST? (Varias Respuestas)

- Fast
- Isolated
- Repeteable, Self Validating
- Timely

76. ¿Cuándo no se debe hacer una prueba automatizada?

- La aplicación solo va a tener un paso a producción
- La aplicación no tiene adherencia con los frameworks
- No se aprecia un costo - beneficio positivo
- Todas las anteriores

77. ¿Cuáles de los siguientes son pasos del principio AAA? (Varias Respuestas)

- Arrange
- Act
- Acceptance
- Assert
- Approve
- Todas las anteriores.

78. La siguiente definición a que tipo de prueba hace referencia: "Son pruebas ligeras que validan la funcionalidad de un objeto de código y su lógica de negocio, son la mano derecha del desarrollador y normalmente son las que mayor cantidad de casos tienen construidos"

- Pruebas de Aceptación
- Pruebas Unitarias
- Pruebas E2E
- Pruebas de humo o smockTest

79. ¿Qué es T.D.D?

- Técnica de desarrollo que consiste en crear pruebas unitarias que le mostrarán al desarrollador el código que debe escribir
- Técnica de pruebas que verifican el funcionamiento del software.
- Es una estrategia de verificación de la implementación de las buenas prácticas de programación.
- Es una herramienta para hacer pruebas de penetración.

80. ¿Cuál framework se utiliza para poner en práctica BDD?

- Screenplay
- Gherkin
- Cucumber
- Page Object Model

81. ¿Qué es ATDD?

- Desarrollo guiado por pruebas de software
- Pruebas de aceptación
- Desarrollo orientado a pruebas de Aceptación
- Desarrollo con arquitectura de Aceptación

82. ¿Cuáles son los beneficios de usar T.D.D? (Seleccione DOS)

- Obtener software tolerante al cambio
- Probar los apliance que están externos al software
- Economizar en su mantenimiento
- Recortar el tiempo de desarrollo

83. ¿Cuáles de los siguientes son principios usados para crear pruebas unitarias? (Seleccione DOS)

SOLID	FIRST	BUILDER	AAA

84. ¿Cuáles de las siguiente herramientas permite realizar análisis estático de código en el pipeline de integración continua? Seleccione dos opciones.

- SonarQube.
- Azure DevOps.
- Kiuwan.
- OpenShift.

85. Según la pirámide de Cohn, ¿cuál porción de pruebas debe tener mayor relevancia e impacto en el desarrollo de software?

- Pruebas E2E.
- Pruebas de GUI.
- Pruebas unitaras.
- Pruebas de integración.

86. ¿Cuáles de los siguientes son pasos del Ciclo T.D.D? (Varias Respuestas)

- Definir los features
- Agregar Pruebas y Ejecutar Pruebas
- Hacer cambios al código en caso de necesitarlo
- Ejecutar el código
- Todas las anteriores.

87. Un equipo de desarrollo de software debe consumir un Web Services desde la prueba E2E que valida el escenario más crítico del negocio. Dicho Web Services es muy inestable en el ambiente de QA. ¿Qué debería hacer el equipo para tener completitud en la prueba E2E?

- No debe hacer la prueba E2E. Desde las pruebas E2E no se deben consumir Web Services.
- Consumir la instancia del Web services que se encuentra en el ambiente productivo.
- Pedirle a EMMA virtualizar el Web Services.
- La prueba E2E debe omitir el consumo del web services.
- La prueba E2E debe acceder directamente a la base de datos.

88. ¿Cuál es el enfoque de las pruebas de aceptación de forma automática en los pipeline de release y cómo se diseñan?

- Se enfocan en verificar si el sistema es "apto para el uso". Se diseñan a partir de las definiciones de requerimientos, casos de uso y de los procesos de negocio definidos.
- Se enfocan en verificar si el sistema es "altamente disponible". Se diseñan a partir de requisitos no funcionales de la aplicación.
- EL principal enfoque es validar las unidades de código y pueden ser diseñadas con técnicas como TDD.
- El enfoque de cualquier tipo de pruebas es validar la integración de los sistemas. Se diseñan a partir de requisitos funcionales y de arquitectura.

89. ¿Cuál es el propósito de BDD?

- Plasmar la funcionalidad de la aplicación en términos del negocio.
- Desarrollar la aplicación de acuerdo a las pruebas.
- Desarrollar pruebas automatizadas para bases de datos
- Ninguno de los anteriores

Docker y contenedores 10 preguntas

90. ¿Cuál de los siguientes comandos se utiliza para detener un contenedor en ejecución?

- kubectl killall container_id
- docker kill -9 image_pid
- docker stop container_id
- kubectl stop pod_id
- la 3 y 4 son correctas

91. ¿Cuál de los siguientes comandos debería ejecutar para ver todos los contenedores en ejecución en Docker?

- docker ps
- docker all
- docker list
- La 2 y 3 son correctas.

92. Un Dockerfile es:

- El formato de almacenamiento de las imágenes de contenedores Docker
- El formato de almacenamiento y representación de los contenedores Docker
- El archivo de texto que contiene la descripción del despliegue de los contenedores en entornos Cloud Native
- El archivo de texto que contiene todos los comandos para crear una nueva imagen

93. _____ es un servicio alojado en la nube que proporciona capacidades de almacenamiento y distribución para imágenes de contenedores públicas y privadas.

- Docker Cloud Orquestrator
- Docker Swarm Unified Repository
- Prisma Cloud Repo
- Elastic Container Registry
- Docker Compose Registry

94. Es cierto afirmar con respecto a contenedores Docker:

- Docker es una tecnología basada en la virtualización de contenedores a través del concepto de Hypervisor, generando un aislamiento completo, lo cual permite simular un sistema operativo completamente diferente.
- Docker proporciona un muy alto nivel de aislamiento entre contenedores, pero presenta cierto costo en el uso de recursos ya que requiere virtualización del sistema operativo de cada contenedor.
- Docker es un sistema de orquestación de contenedores que se ejecutan en entornos Cloud Native principalmente en AWS Elastic Container Service.
- Docker hace uso de capacidades nativas del Kernel de Linux para separar y aislar procesos sin requerir virtualización.
- La 1 y 3 son correctas.

95. En entornos productivos y pre-productivos, cuando se desea ver los logs de una aplicación desplegada a través de kubernetes podemos:

- Usar el comando docker logs especificando el id de la imagen del contenedor en Kubernetes.
- Consultar los logs en CloudWatch en caso que un extractor de logs esté configurado apropiadamente.
- Usar el comando kubectl kubectl logs -f pod_name.
- Usar el comando docker stats y docker logs sobre la instancia maestra del orquestador junto con el id de la imagen.
- La 2 y 3 son correctas

96. ¿Cuáles son los pasos del ciclo de vida del contenedor Docker?

- Build, pull, run
- CI, CD, Release
- Scheduling, deployment, replacement
- Todos los anteriores

97. Cuando se usan contenedores Docker podemos afirmar:

- Se crear una instancia virtualizada nueva a nivel de SO por cada contenedor en ejecución para garantizar el aislamiento a través del Hypervisor y los namespaces.
- Los contenedores comparten siempre el mismo rango de puertos, es decir si un contenedor en ejecución usa el puerto 8080, ningún otro contenedor en el mismo host puede usar el puerto 8080 ya que no se puede generar un mapeo diferente en el Host.
- Los contenedores no comparten los puertos, es decir si un contenedor usa el puerto 8080, otro contenedor en el mismo host puede usar el mismo puerto 8080, solo difiere el mapeo si se desea exponer en el Host.
- Todos los contenedores se ejecutan como procesos del mismo SO, sin virtualización, se usan Linux cgroups y namespaces para garantizar aislamiento.
- La 3 y la 4 son correctas

98. ¿Qué significa el comando FROM en un Dockerfile?

- El sistema operativo del host
- La imagen en la que se basa el contenedor
- El sistema operativo invitado
- El autor del actual contenedor

99.

En Docker, ¿cómo se pueden enumerar tanto los contenedores que se encuentran en funcionamiento como los que están parados?

- docker system ps -a
- docker run -a
- docker ps -a
- docker system run -a

Seguridad en el desarrollo de SW 9 preguntas

100. La definición:
"Debilidad de un sistema de información, componente tecnológico, procesos o personas que pueden ser explotadas por una o más amenazas"
¿A cual de los siguientes términos corresponde?

- Ataque
- Amenaza
- Vulnerabilidad
- Ninguna de las anteriores

101. ¿Cuál de los siguientes no corresponde a un principio de diseño seguro?

- Mediación completa
- Segregación de funciones
- Defensa a profundidad
- Economía de mecanismos
- Cadena segura
- Diseño abierto

102. Jose decide implementar el principio de diseño oscuro en la construcción de una aplicación financiera. Desafortunadamente sus diseños se han hecho públicos en la organización donde trabaja, ¿consideras que al seguir este principio la aplicación podría tener algún riesgo de seguridad?

- Sí, a pesar de tener un principio de diseño oscuro, podría tener otra vulnerabilidad no identificada.
- No, el hecho de tener aplicado este principio garantiza que el diseño por si mismo no tiene riesgos de seguridad al ser publicado.
- Sí, el hecho de no tener aplicado un principio de diseño abierto ya representa un riesgo de seguridad.
- No, con la aplicación de este principio de diseño se garantiza la seguridad por cualquier áspecto del diseño.

103.

Principio de desarrollo seguro para limitar los privilegios a una persona o proceso para para completar una tarea por un tiempo determinado:

- Segregación de funciones
- Menor privilegio
- Economía de mecanismos
- Economía de privilegios
- Mecanismos no compartidos.

104.

Los mecanismos de seguridad no deben añadir dificultad al acceso de un recurso, o de ser necesario se debe ocultar la complejidad introducida por los mecanismos de seguridad.
Dada esta afirmación, ¿A cuál de los siguientes principios de diseño seguro corresponde?

- Eslabón más débil
- Adaptabilidad psicológica
- Componentes existentes
- Mediación social
- Aceptación de experiencia

105.

"Eventualmente un sistema fallará, pero deberá hacerlo de forma segura. La pérdida de funcionalidad no debe implicar pérdida de la seguridad."
¿A cuál de los siguientes principios de diseño seguro corresponde la anterior afirmación?

- Diseño seguro
- Safe failover
- Fallo seguro
- Defensa profunda
- Eslabón fuerte

106. **Cuando se trata de datos, ¿cual de los siguientes mecanismos podríamos considerar defensivos? (Seleccione dos)**

- Segmentación de red
- Antivirus
- ACL
- Cifrado
- Programas de aprendizaje para usuarios
- Túneles IPSec

107. **¿Cuál de los siguientes principios de diseño establece que la concesión de permisos a una entidad no debe basarse únicamente en una sola condición?**

- Mecanismo menos común
- Separación de privilegios
- Seguridad total
- Defensa en profundidad
- Ninguno de los anteriores

108. **¿Cuál de las siguientes definiciones se ajusta mejor al pricipio "Eslabón más fuerte"?**

- Un ataque típicamente intentará inicialmente vulnerar los componentes más débiles del sistema.
- La resistencia de un software dependerá en gran medida de la seguridad de los principales componentes del sistema.
- Los componentes más debiles del sistema estarán a salvo siempre y cuando se haga un diseño seguro de los principales componentes.
- Ninguna de las anteriores
- "Eslabón más fuerte" no corresponde a un principio de desarrollo seguro.

Comunicación 7 preguntas

109. ¿Cuál de los siguientes métodos HTTP debería ser de naturaleza idempotente?

OPTIONS	DELETE	POST	HEAD

110. ¿Cuál de los siguientes componentes de la solicitud HTTP indica métodos HTTP como GET, POST, DELETE, PUT, etc.?

- VERB
- URI
- HTTP Version
- Request Header

111. ¿Cuál de los siguientes componentes de la respuesta HTTP contiene metadatos para el mensaje de respuesta HTTP como pares clave-valor?

- Status/Response Code
- HTTP Version
- Response Header
- Response Body

112

¿Cuál de los siguientes códigos de estado HTTP significa SIN CONTENIDO, cuando el cuerpo de la respuesta está vacío, por ejemplo, una solicitud DELETE?

200	201	204	304

113

¿Qué método de clase RestTemplate realiza una operación HTTP HEAD?

- headForHeaders (String, Object ...)
- getForObject (String, Class, Object ...)
- postForLocation (String, Object, Object...)
- postForObject (String, Object, Class, Object...)

114

¿Cuál de los siguientes no es un método HTTP?

CREATE	POST	PUT	OPTION

115

¿Cuál de las siguientes directivas del encabezado de control de caché de la respuesta HTTP indica que el recurso no se puede almacenar en caché?

- Public
- Private
- no-cache/no-store
- max-age