Dirección de
Tecnologías de la Información
-Senacyt-

Evaluación

Desarrollador de Sistemas de Información

Confidencial

GOBIERNO de GUATEMALA
DR. ALEJANDRO GIAMMATTEI

SECRETARÍA NACIONAL DE CIENCIA Y TECNOLOGÍA

Evaluación de conocimientos teóricos y prácticos

Nombre: José Carlos Soberanis Ramírez

Correo electrónico: jcsru13@gmail.com

Github: enlace

Dirección de Tecnologías

"La Dirección de Tecnologías de la información de la Senacyt es el órgano responsable de planificar, organizar, dirigir, supervisar y controlar los recursos y sistemas informáticos de la Secretaría, estableciendo normas, estándares, políticas y metodologías relacionadas con las tecnologías de la información, comunicaciones e infraestructura, así como dar apoyo tecnológico para el desarrollo de las Tecnologías de la Información y Comunicación a Nivel Nacional. "

Evaluación teórica -50pts-

Instrucciones

A continuación, se presenta una serie de enunciados los cuales tiene como objetivo medir su capacidad de toma de decisión, así como conocer su punto de vista en el desarrollo de sistemas informáticos. Todos los enunciados de selección múltiple deben justificar su respuesta.

Parte 1: Scrum -2pts cada una-

1. **¿Qué es Scrum?**

    a. Una metodología de desarrollo de software

    b. Una metodología ágil de proyectos

    c. Un marco de trabajo

    d. Otro

Justifique su respuesta: Scrum se suele 'categorizar' como una metodología ágil, sin embargo, Scrum es un marco de trabajo dado que hace referencia al conjunto de prácticas, actividades, criterios para resolver cierto problema, una metodología es más una guía general, y entre las metodologías ágiles suelen ser aquellas que son flexibles durante el desarrollo.

2. **Analice y responda: Se planificaron 6 sprints, pero en el transcurso del desarrollo del producto, nos damos cuenta de que el sprint número 5 ya no genera valor, ¿qué propone usted?**

    a. Cancelar el sprint

    b. Realizar el sprint porque ya fue planificado

    c. Consultar a la parte interesada.

    d. Consultar al Project Manager

    e. Otro

Justifique su respuesta: Como desarrollador Jr. que se encontrará bajo la supervisión del project manager, no me corresponde consultar la parte interesada ni de tomar la decisión final de cancelar o realizar el sprint (esa es una decisión de equipo), aunque apoyaría la idea de cancelar el sprint y seguir con el 6to.

3. **¿Cuál de los siguientes actores no deben de existir en Scrum?**

    a. Scrum Master

    b. Product Owner

    c. Project Manager

d. Developer team

e. Stakeholder

f. Ninguna de las anteriores

Justifique su respuesta: Todos los actores mencionados forman parte de Scrum

4. **Estamos realizando el sprint 2, pero nos damos cuenta de mejoras que podemos incluir en el sprint 3, ¿lo podemos hablar en medio del sprint 2 o hasta el Sprint Planning 3?**

   a. Sí, se puede hablar en el 2, es el Sprint Refinement

   b. No, eso le corresponde al Sprint Planning 3

   c. Lo decide el Scrum Master

   d. Lo decide el Product Owner

Justifique su respuesta: Considero que si se puede hablar en el 2, en los daily scrums, sin embargo no a fondo, por lo que considero que se puede mencionar en el 2° pero debería hablarse a profundidad en el Sprint planning 3

5. **El objetivo principal de scrum es:**

   a. Entregas ágiles y continuas

   b. Desarrollo de software

   c. Hablar más con el cliente o interesado

   d. Generar valor vía el desarrollo de un producto

   e. Otro

Justifique su respuesta: Los mencionados son todas características de Scrum, sin embargo, las entregas constantes en períodos cortos (habiendo utilizado un método de priorización) ayudan a brindar valor rápidamente al desarrollo del producto, garantizando así que este cumpla con lo que desea el cliente.

Parte 2: Desarrollo de software -4pts cada una-

6. **Es el momento de subir el código a control de cambios, usted está en la rama de desarrollo local y debe ser trasladado a producción, pasando por ambiente de pruebas. Indique cuál es la secuencia de instrucciones ideal para validar la integridad del sistema y el control de versiones.**
   R//

   1. Realizar el commit correspondiente a la rama en la que estoy trabajando

   2. Realizar un pull request a la rama del ambiente de pruebas, en esta parte lo más probable es que una persona se encargue de hacer un coding review para verificar

que el código cumpla con los estándares, una vez realizado esto (asumiendo que se encuentra correcto se integra a la rama de pruebas)

3. Una vez, pasadas las pruebas en el ambiente de pruebas se haría un merge hacia la rama de producción.

7. **Se tenía un sistema antiguo en un servidor, dicho servidor dejó de funcionar y con ello, por el modelo monolítico que tenía, hubo perdida de datos, ¿cómo se hubiera evitado este fallo?**
   R// Teniendo dos servidores y utilizando un balanceador de carga, de esta manera si cualquiera de ambos servidores por alguna razón fallase, aún se tendría el sistema activo, aunque probablemente a menor rendimiento. Sin embargo, podría depender de a qué nivel se desee, teniendo hasta arquitecturas complejas de redundancia para garantizar la alta disponibilidad.

8. **Se tiene un API que debe de traer 1 millón de datos aproximadamente, la consulta puede tardar 1 minuto, usted propone una llamada**:

   a. Sincrónica

   b. Asincrónica

Justifique su respuesta: Las llamadas síncronas tienden a esperar a que el sistema responda para realizar una acción, de ser asíncrona se pueden realizar acciones mientras se obtiene la información, dependiendo la situación puede que se requiera cargar esta información antes de realizar otra acción, sin embargo, por norma general, propondría asincrona.

9. **Una versión publicada en producción presenta inestabilidad, ¿qué acciones toma usted?**

R// Al estar usando un controlador de versiones, realizaría un 'rollback' hacia una versión estable del sistema.

10. **¿Tiene algún beneficio una tipificación de un lenguaje o es indiferente?**

R// Siempre es más legible un código de un lenguaje tipificado así como la identificación de errores, sin embargo, no suele ser un 'factor' que influya mucho en la decisión de qué lenguaje usar para cierto proyecto, dado que suele depender de muchos otros aspectos.

Parte 3: Bases de datos -4pts cada una-

11. **Defina con sus propias palabras qué es un modelo maestro-detalle.**

R// Suelen ser un par de tablas relacionadas en las que podríamos decir que se encuentra una relación 1:*, en estas se conoce a una que generalmente 'contiene' a varias de las otras, un ejemplo general de esto son las facturas, donde podríamos tener una tabla factura (No., cliente, fecha, etc) y una tabla detalle_factura (id_producto, descripcion, cantidad) en donde detalle_factura depende de la tabla factura.

12. **Un sistema necesita la carga de archivos, tales como expedientes, fotografías, etc, ¿cuál es la mejor opción para guardar los archivos?**

a. En el sistema de archivos del servidor

b. En una base de datos relacional

c. En una base de datos documental

d. En otro servidor

e. Otro

Justifique su respuesta: Personalmente considero que depende del tamaño de los archivos que se vayan a cargar, de cierta forma puede ser viable almacenar en el sistema de archivos, almacenar archivos en la base de datos tiende a ser problemático también debido a la conversión del archivo binario a un campo tipo BLOB o similar pero sería viable con archivos un poco más grandes.

16. **¿Una hoja de cálculo es una base de datos?**

a. Sí

b. No

Justifique su respuesta: Una hoja de cálculo tiende a confundirse con una base de datos, esto no es así, una base de datos está optimizada a nivel del almacenamiento de información, hecha para la fácil escritura y lectura de datos. Una hoja de cálculo posee una interfaz gráfica lo cual nos facilita manipular la información en primera instancia pero no está optimizada a más bajo nivel sino para un fácil uso.

17. **¿En qué casos utilizaría usted una vista?**

R// En tablas que no tiendan a cambiar constantemente, o que no es muy relevante tener la información más actualizada en todo momento, de esta forma podemos optimizar las consultas.

18. **¿Qué tipos de bases de datos conoce o ha trabajado, describa con sus propias palabras una breve reseña y mencione algunas tecnologías?**

R//

- Postgresql: Sistema de base de datos en el que más he trabajado, es open source, siendo una de las bases de datos (personalmente) más flexible, teniendo incluso la posibilidad de almacenar campos con información en tipo Json (aunque no sea una buena práctica del todo).
- Mysql: Sistema de base de datos relacional, que considero tiene una sintaxis más simple que postgresql, sin embargo tienen muchas similitudes dado que ambos se basan en SQL, considero que es una base de datos bastante fácil para comenzar al igual que su hermano open source Mariadb.
- MongoDb: Sistema de base de datos nosql basado en documentos, tienden a ser bases de datos muy flexibles, lo cual dependiendo del sistema puede ser muy provechoso, aunque también tiende a ser confuso o ciertamente problemático si no se tiene mucha experiencia con ellas.

19. **Mencione 5 motores de bases de datos NoSQL y en qué casos los utilizaría.**

- **MongoDb (Orientado a documentos)** -> Estos suelen almacenar de manera similar a los clave-valor, sin embargo, almacenan estructuras más complejas en formatos tipo xml, json, etc.
- **Redis (Clave-valor)** -> Cuando se requiera acceder a información muy básica (sin una estructura lógica compleja) de manera muy rápida.
- **Neo4j (De grafo)** -> Funciona también similarmente a las clave-valor, aunque estas admiten estructuras más complejas, así como relaciones entre los diferentes datos.
- **ScyllaDb (Columnar)** -> Es muy similar a un sistema de base de datos relacional, a diferencia de que en este una columna posee un solo valor, mientras que en las nosql, se tiene una 'familia' de valores, podría usarse en situaciones en las que se requiera una estructura más definida algo que en el resto no suele tenerse (a excepción de grafos, aunque este podría ser más complejo)

20. Explique la diferencia entre INNER JOIN, LEFT JOIN y RIGHT JOIN.

Un inner join hace referencia a una consulta que tiene en cuenta ambos lados de las tablas que se desean unir, en este caso solo mostrará la información que haga 'match' en ambos lados (tablas).

Un Left join es similar, con la excepción de que mostrará toda la información de la tabla de la izquierda, en conjunto con los datos que hagan 'match' de la derecha.

Un right join funciona exactamente igual que el left join, pero mostrando toda la información de la tabla derecha, en conjunto con los datos que hagan 'match' de la izquierda, puede hacerse una misma consulta utilizando left o right join si se conoce bien cómo es que estas trabajan.

Evaluación práctica -50pts-

Instrucciones

A continuación, se presenta una serie de enunciados los cuales tiene como objetivo medir su nivel de conocimiento técnico y de desarrollo de software. Para lo cual se le solicita crear un ambiente de desarrollo local basado en PHP, MySQL/MariaDB, y otras tecnologías que se indican en el enunciado para resolver los escenarios planteados.

Así también, todo el código utilizado deberá subirlo a una carpeta GitHub para su evaluación en la fecha y horario indicado.

Deberá incluir este archivo, así como cualquier documentación que considere oportuna, no olvide notificar a Recursos Humanos el enlace a su carpeta en GitHub para programar una reunión de 20 dónde se le pedirá presentar la solución desarrollada.

Serie I – 10 pts.

Se tiene un registro de personas y se necesita el almacenamiento de sus datos personales como nombres, apellidos, teléfono, dirección, fecha de nacimiento y otro dato personal de relevancia, estas personas pueden tener o no uno o varios logros académicos de los cuales pueden ser licenciatura, maestría, doctorado o también diplomados, certificaciones entre otros tipos de logros académicos, el logro académico debe de tener registrado año, título del programa, institución de la que lo obtuvo y otra información que sea necesaria y pertinente. De igual manera se desea que se registre sus áreas de desempeño de las personas, por ejemplo: ciencias de la computación, psicología, ciencias económicas, entre otros, recordando que pueden tener varias áreas de desempeño. Estas personas registradas deben de contar con usuario y contraseña del sistema.

Se desea que se construya una base de datos relacional del anterior enunciado. No olvide incluir el diagrama ER.

Serie II – 15 pts.

Se necesita que realice un *frontend* en una librería o framework basada en JQuery que tenga lo siguiente:

- Login para las personas registradas
    - Actualización de datos personales
    - Actualización de datos académicos
    - Actualizar área de desempeño
    - Agregar o quitar logros académicos
    - Agregar o quitar área de desempeño

- Una vista (**no** se necesita un login para acceder) en la cual se pueda visualizar las personas registradas, así como el último grado académico que han conseguido con su respectiva información de dicho grado académico.
- Una vista (**no** se necesita un login para acceder) en la cual se pueda visualizar las áreas de desempeño registradas, así como el número de personas que tienen esa área de desempeño.

Serie III – 25 pts.

Se necesita que realice un *backend* de todo el manejo de base de datos (inserciones, actualizaciones, consultas, entre otros) y lógica del negocio del escenario planteado en los incisos anteriores por medio de APIs Rest en Node JS.

Aspectos que evaluar

- Modelo relacional creado con sentencias SQL
- Uso adecuado de llaves primarias, foráneas e índices (este último si es necesario)
- Uso adecuado de framework o librería JQuery, si lo necesita puede apoyarse de PHP.
- El acceso a datos, manejo de estos, lógica de negocio y demás debe de ser desarrollado **netamente** en Node JS y API Rest.
- Es libre utilizar los módulos de Node JS que necesite para crear las rutas de las APIs Rest o demás módulos.
- Para consumir las APIs Rest puede hacerlo desde JavaScript o PHP.