Instrucciones:

Entrega las respuestas a las preguntas en un documento de texto con el mismo formato, el código debe de ser adjunto en un zip ordenado por lenguajes o en su caso el link al repositorio o al snippet que lo contiene. Para nada debe de enviarse código en un documento de texto. En caso de que alguno no pueda ser resuelto explicar el porqué y continúa con otro ejercicio.

Tiempo máximo de entrega: 3 días

JavaScript

1. escribir un programa que dada una cadena regresa la reversa de esa cadena en un nuevo objeto, ej: ver el link. No se pueden usar las funciones nativas de arreglos o cadenas como “reverse” o “map”. Se debe explicar la complejidad del programa.
2. escribe un programa que dada una cadena regresa la reversa de esa cadena en el mismo objeto, ej: ver el link. No se pueden usar las funciones nativas de arreglos o cadenas como “reverse” o “map”.
3. Implementa una función que “fold” en un json. La función debe de sumar las hojas si son numeros y si la llave es diferente de “bar”. En caso de que alguna hoja no sea un número se debe de lanzar un error. Debe de regresar el resultado en el mismo objeto. Ej: ver el link.
4. escribe un programa que dado una n>=1 calcule f(x)=1+...+n. La complejidad debe de ser constante, explicar porque.

Opcional

1. escribe los mismos programas de javascript solicitados pero en el lenguaje de tu preferencia de la lista:
    a. java
    b. python
    c. elixir

Bases de datos

1. Diferencias fundamentales entre una SQL y una NoSQL, si quieres con ejemplos con dbs concretas (MariaDB, MongoDB, Cassandra, etc)
2. Dada la relación con los siguientes atributos
    a. usuarios
        i. nombre
        ii. apellidos
        iii. curp
        iv. rfc
        v. dirección (calle, colonia, cp, municipio, estado, país, etc); más de una.
        vi. nacionalidad; más de una
        vii. sexo/género
        viii. username
        ix. password
        x. correo electrónico
    b. sesiones de los usuarios

        i. fecha de creación
        ii. expiración
        iii. tipo
    c. proyectos
        i. nombre
        ii. tipo
        iii. tags
        iv. encargados (usuarios)
        v. participantes (usuarios)
        vi. fecha de creación
        vii. fecha de actualización
    d. tareas de los proyectos
        i. nombre
        ii. descripción
        iii. tareas relacionadas
        iv. encargado (usuario)
        v. fecha de creación
        vi. fecha de vencimiento
        vii. fecha de cancelación (opcional)
        viii. fecha de actualización
    e. entregas por tareas
        i. fecha de entrega
        ii. descripción
        iii. calificación

    normaliza como creas conveniente y sugiere cambios para una base de datos relacional
3. Describe la relación anterior usando una base de datos NoSQL
4. Explica que es DDL
5. Explica que es DML
6. Que es un ODM y que diferencias tiene de un ORM
7. Describe funciones de agregación/agrupación
8. Que significa “High Partition Tolerance” en bases de datos SQL
9. Que significa “High Availability” en bases de datos NoSQL como MongoDB

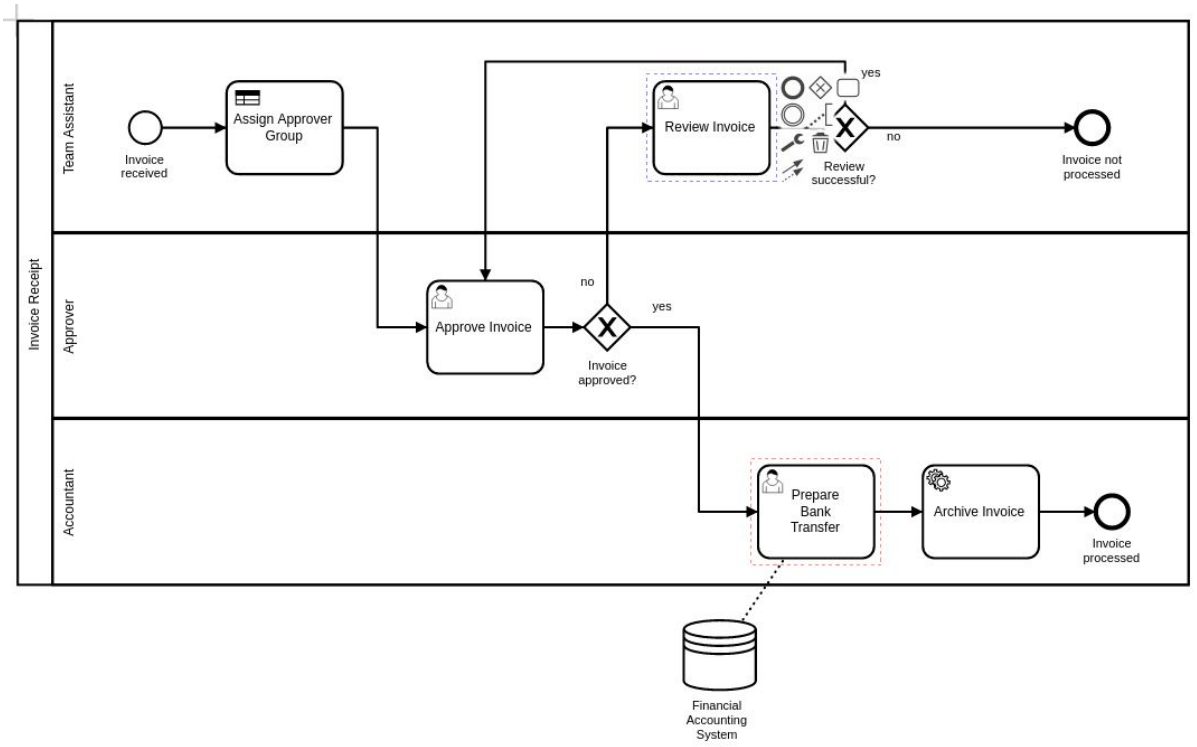
Dev

1. Generar un API RESTful para la base de datos que diseñaron (SQL o NoSQL).
    a. El lenguaje es abierto pero se debe de usar un framework tanto para generar los endpoints como para acceder a la base de datos.
    b. Se deben de incluir pruebas.
    c. Se debe de incluir las siguientes función de agregación/agrupación.
        i. promedio de la calificación de las entregas de los usuarios asignados a cada tarea
        ii. agrupar proyectos por tipo y dar el mínimo y el máximo de las calificaciones de las entregas
    d. Se debe de definir las versiones a usar
2. Sobre el desarrollo anterior se debe de incluir
    a. una propuesta de despliegue; dado tu framework y tu base de datos elegida proporcionar una receta de despliegue así como flujo de mantenimiento

b. un diagrama del mismo; de todos los componentes y la comunicación de los mismos

Testing

1- Generar pruebas unitarias para cada uno de los programas usando la suite de pruebas que creas necesario
2- Dado el siguiente flujo detalla los casos de prueba a realizar



Preguntas

1- ¿Cuál es la diferencia entre validación y verificación?
2- ¿Cómo aseguras que un código tiene suficiente cobertura de pruebas?
3- ¿Que es una prueba funcional?
4- ¿Cuál es la diferencia entre pruebas de integración y pruebas unitarias?
5- ¿Cuál es la diferencia entre “Quality Assurance” y “Quality Control”?
6- ¿Cuáles son las pruebas de benchmark?
7- Como borrar los últimos cambios sin commitear
8- Menciona un patrón de diseño que hayas utilizado y describe cómo ayudó en el proyecto
9- ¿Cómo describes un antipatrón de diseño?