Apellido y Nombre: ……... **Legajo:** ……
Docente con quien cursó la materia: ……...

Contexto

En esta ocasión, a ***2Diseños*****,** consultora de software*,* le toca hacer un desarrollo interno. Ya que el sistema de integración continua que contratan no se adapta a todos los tipos de proyectos que utilizan, decidieron crear el suyo propio.

Se pide contemplar una solución que **permita ejecutar la construcción de un componente**. Esta construcción podrá ser ejecutada por decisión del usuario, o en horarios determinados definidos en la configuración del componente.

La construcción de un componente tiene las siguientes etapas:

1. Descargar el código fuente de un origen de datos.
2. Generar el componente.
3. Verificar que el componente funcione adecuadamente.

Para cada componente deberá especificarse:

Origen de datos:

Existen múltiples orígenes de datos: *FTP, Git, Local* y podrían agregarse más en el futuro. Actualmente no debemos realizar las implementaciones de estos orígenes porque lo realizará otro equipo. Los orígenes de datos necesitan de la siguiente información

- FTP: IP, puerto, usuario y contraseña.
- Git: URL del repositorio, usuario y contraseña.
- Local: ruta al directorio local.

Generación y verificación:

- Lenguaje (obligatorio). La generación en todos los lenguajes tiene tres tareas bien definidas: descarga de dependencias, generación y verificación. Las tareas de generación y verificación son similares en todos los lenguajes, pero la forma de conseguir las dependencias varía. A continuación se detalla el proceso de descarga de dependencias para algunos de los lenguajes que queremos soportar:
    - Para Java, usa Maven y estos son los pasos:
        - Leer las dependencias del "pom.xml" en el directorio raíz del componente.
        - Descargar el código fuente de cada dependencia (.java)
        - Compilar y empaquetar cada dependencia (.class/.jar)
        - Guardar cada .jar en el directorio “/usuario/.m2”
    - Para Javascript, usa NPM y estos son los pasos:
        - Leer las dependencias del archivo "package.json" en el directorio raíz del componente.
        - Crear el directorio "node_modules" en el directorio raíz del componente.
        - Descargar cada dependencia dentro del directorio "node_modules".

- Debe permitir agregar comandos extra a realizar en la etapa de generación (opcionales). Es importante que estos comandos se realicen en el orden definido por el usuario. En el caso de que todos los comandos sean satisfactorios procede a la etapa de verificación. Ejemplo:

```
mvn generate:docs
./deploy_artifacts.sh
```

Resultado de la operación:

- Tiene que registrarse a qué hora se ejecutó la construcción del componente y si se pudo ejecutar satisfactoriamente o no. Es satisfactoria la construcción cuando todas las etapas del proceso fueron ejecutadas satisfactoriamente.
- Se podrán realizar varias acciones con el resultado de la construcción de un componente, como envío por mail, notificación por twitter, etc. Esta configuración es opcional. Se debe poder configurar bajo qué condiciones queremos realizar dichas acciones:
    - Siempre.
    - Siempre que la creación sea satisfactoria.
    - Siempre que la creación falle.
    - Cuando la creación empieza a fallar.
    - Cuando la creación vuelve a funcionar correctamente.
    - Cuando la creación cambie de estado.

Solución técnica

Se deberá proponer una solución que contemple los siguientes aspectos:

A) Arquitectura

Proponer una arquitectura del sistema a alto nivel y comunicarla. Deberá especificar los componentes de alto nivel, los sistemas externos y la comunicación entre componentes.

B) Modelo de Dominio

Modelar el dominio con el paradigma de orientación a objetos. Comunicar, empleando diagramas, código y prosa, contemplando todos los requerimientos del caso. Si considera apropiado utilizar patrones de diseño, indicarlos y justificar su uso.

C) Persistencia

Utilizando un DER, código anotado y/o prosa, se pide explicar cómo persistir el modelo de objetos del punto anterior en una base de datos relacional, indicando claramente:

- Qué elementos del modelo es necesario persistir.
- Las claves primarias, las foráneas y las restricciones (constraints) según corresponda.
- Estrategias de mapeo de herencia utilizadas, si fueran necesarias.
- Las estructuras de datos que deban ser desnormalizadas, si corresponde.

Condiciones de aprobación: Para aprobar debe sumar como mínimo 60 puntos y no menos del 50 % en cada uno. (A) otorga 20 puntos, (B) 40 puntos y (C) 40 puntos.

Explicar supuestos y justificar decisiones de diseño.