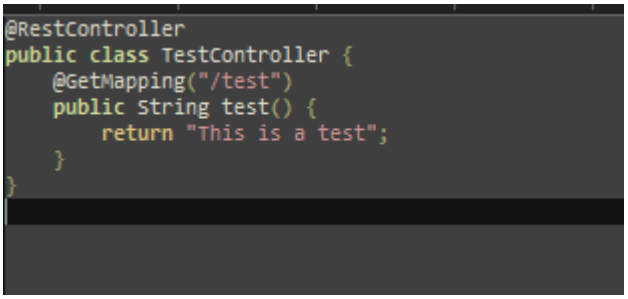
Java Spring Boot 3.

A continuación, se presenta un conjunto de preguntas sobre el lenguaje Java y el framework Spring
Boot. Por favor, proporciona una respuesta para cada una en tus propias palabras.

- ¿Qué es Spring Boot?
  Es un framework de Java que permite crear aplicaciones web y microservicios, es un herramienta sencilla y rápida, la ventaja principal es la automatización de varios aspectos de los proyectos y dependencias.
- ¿Cuál es la diferencia entre Spring y Spring Boot?
  La configuración automática en Spring boot, mientras que en Spring las dependencias el servidor y demás configuraciones deben ser manualmente realizadas, Spring boot lo automatiza.
- ¿Cuál es el propósito de Maven en Java?
  Automatizar proyectos de Java, donde podemos compilar, ejecutar y probar, muy util y facil de usar con proyectos.
- De un ejemplo de cómo utiliza comúnmente Maven.
  mvn clean install
  mvn spring-boot:run
  Por ejemplo con estos comandos y con un archivo de configuracion XML podemos ejecutar nuestro proyecto de Java.
- ¿Qué es un repositorio?
  Es un espacio donde podemos guardar las dependencias que son bibliotecas externas.
- ¿Qué es un controlador?
  Es una clase donde se gestionan las solicitudes HTTP y se definen las rutas, se definen si son GET, POST…
  Ej:



```java
@RestController
public class TestController {
    @GetMapping("/test")
    public String test() {
        return "This is a test";
    }
}
```

- ¿Qué es un dto ?
  Data Transfer Object es una clase que se usa para transportar datos de una capa a otra en la aplicacion, se podria decir que es similar a los objetos en POO
- ¿Por qué es recomendable emplear hibernate?
  Porque facilita la interacción entre los objetos de Java y las bases de datos relacionales.
- ¿Puedo retornar contenido renderizable desde Spring Boot?
  Si, por ejemplo un JSON o archivos PDF, imágenes…
- ¿Cómo puedo documentar mis controladores?
  Con Swagger es una buena manera de agregar una documentación apropiada y fácil de leer.

- ¿Qué es JPA?
  Es una especificación que define cómo se pueden mapear objetos Java a las tablas de las bases de datos.
- ¿Qué es la inyección de dependencias?
  Es un patrón que permite entregar dependencias a los componentes del sistema sin necesidad de que los objetos creen estas, es decir que un objeto no necesita conocer cómo se implementó una dependencia porque solo necesita trabajar con una interfaz para poder trabajar.
- De un ejemplo de cómo usar o cuándo usar la inyección de dependencias.

  Por ejemplo tenemos una clase A que necesita ciertos métodos o propiedades de la clase B, con la inyección de dependencias no es necesario crear objetos para poder utilizar los métodos de la clase B, esto es muy bueno por el desacoplamiento.

```java
//Sin inyeccion de dependencias
public class PC {
    private Component component;

    public PC() {
        this.component = new Component(); // Es necesario crear el objeto
    }

    public void TurnOn() {
        component.someAction();
    }
}
//Con inyeccion de dependencias
public class PC {
    private Component component;

    public PC(Tool tool) {
        this.tool = tool; // Se entrega directamente el objeto sin necesidad de crearlo
    }

    public void TurnOn() {
        component.someAction();
    }
}
```

React JS

- ¿Utiliza JS ReactJS?
  Si, se usa para construir interfaces de usuario dinámicas.
- ¿Qué es Node JS?
  Es un entorno de ejecución para JavaScript para automatizar tareas y usar la herramienta npm
- ¿Cuál es el propósito de npm?
  Es el gestor de paquetes de Node.js. Sirve para instalar, actualizar y administrar librerías y herramientas que se usan el proyecto, como React, Axios, Tailwind, etc.
- ¿Cuál es el propósito de vite o webpack?
  Son herramientas que preparan el código para producción, ambos ayudan a que la app cargue más rápido y funcione correctamente en el navegador..

  Webpack: agrupa, transforma y optimiza archivos JS, CSS, imágenes, etc.

Vite: es más moderno y rápido, usa ES Modules y tiene recarga instantánea en desarrollo.

- Proporcione un ejemplo de cómo se emplearía comúnmente npm.
  npm install react
  npm start
  estos comandos se usan para instalar react y ejecutar el servidor de desarrollo
- En el contexto de npm, ¿qué es un paquete?
  Un paquete es una colección de código reutilizable que puedes instalar con npm. Por ejemplo, react, axios, lodash, etc. Cada paquete tiene su versión, dependencias y documentación.
- ¿Dónde se almacena el árbol de librerías de un proyecto en el contexto de ReactJS?
  En la carpeta node_modules/, se crea cuando se ejecuta npm install. Allí se guardan todas las dependencias del proyecto.
- ¿Qué es un hook?
  Un hook es una función especial de React cuya funcionalidad permite usar características de estados (useState) o efectos (useEffect) en componentes funcionales.
- ¿Cuál es el ciclo de vida de un componente en React?
  Montaje: cuando el componente se crea (componentDidMount)

  Actualización: cuando cambia el estado o props (componentDidUpdate)

  Desmontaje: cuando se elimina (componentWillUnmount)

- ¿Puedo emplear el paradigma de POO para desarrollar en ReactJS?
  Si como por ejemplo herencia, encapsulamiento y abstracción.
- ¿Cuándo se usa el hook useMemo?
  Se usa para memorizar valores calculados y evitar cálculos innecesarios en cada renderizado. Es útil cuando tienes funciones costosas o listas grandes que no cambian frecuentemente.

Interacción Navegador Servidor

- Cuando un protocolo de comunicación como HTTP tiene el sufijo "s", ¿qué significa? (Por ejemplo, http://mydomain.com/path se convierte en https://mydomain.com/path).
  "s" en HTTPS significa "Secure". Es la versión segura de HTTP, que utiliza TLS (Transport Layer Security) y es una autenticación del servidor.
- Cuando las aplicaciones involucran tanto el navegador web como el servidor, ¿la ejecución del código ocurre únicamente en el servidor?
  No, se ejecuta tanto en el servidor como en el navegador.

- Cuando cierras una pestaña en tu navegador, ¿es posible almacenar información que persista hasta que regreses a esa página? En caso afirmativo, ¿cómo?

  Si, por ejemplo se puede usar localStorage para guardar datos o las cookies.

- Al interactuar entre el navegador y el servidor, comúnmente utilizamos solicitudes HTTP con métodos específicos (GET, POST, PUT, DELETE, etc.). ¿Se utilizan estos métodos al transmitir contenido como imágenes, sonido, vídeo o archivos? En caso afirmativo, ¿cuál es el método más utilizado?

  Si, el método GET es el más utilizado, el método POST también se usa para cargar archivos.

- ¿Qué lenguajes se utilizan comúnmente en el navegador?

  JavaScript, HTML, CSS

- ¿Se pueden utilizar los lenguajes comúnmente usados en el navegador en el lado del servidor?

  Con NodeJS se puede usar JavaScript en el servidor.

- ¿Se pueden utilizar lenguajes del lado del servidor en el navegador?

  No, el servidor interpreta y envía los resultados al cliente.

Docker

¿Qué es docker?

Es una plataforma de contenedores que permite empaquetar aplicaciones junto con todas sus dependencias en unidades llamadas contenedores.

- Seleccione la opción que no corresponde con un comando de Docker válido.

  -docker shutdown no corresponde

- ¿Dos contenedores pueden compartir las mismas librerías?
  Si se usan volúmenes compartidos si pueden compartir las librerias
- ¿Qué diferencia hay entre una máquina virtual y un contenedor?

  Los contenedores son ideales para microservicios y despliegues rápidos, mientras que las VMs son útiles cuando se necesita un entorno completamente aislado

- ¿Pueden dos contenedores desplegar un servicio en el mismo puerto interno?
  Sí, pueden usar el mismo puerto interno, como el 8080, pero eso mientras no se expongan al mismo puerto externo del host.