Prueba técnica

1. **JavaScript**: Explique la diferencia entre &&, || y ??.

2. **JavaScript**: Escriba una función en Javascript que tome un array de strings y retorne el string más largo.

3. **JavaScript**: Para la siguiente función:

```
function exampleFunction(a, b, c) {
  if (a > b) {
    if (c > a) {
      return c;
    } else {
      return a;
    }
  } else {
    if (c > b) {
      return c;
    } else {
      return b;
    }
  }
}

exampleFunction(10, 15, 20)
```

   I. Documente qué hace *exampleFunction* y cómo funciona.

   II. Calcule su complejidad.

   III. Proponga una mejora. Se tomarán en cuenta factores como legibilidad y complejidad.

4. **JavaScript/Typescript**: Para la siguiente función:

```
function exampleFunction2(y) {
  var z = 0;
  for (var i = 0; i < y.length; i++) {
    if (y[i] % 2 === 0) {
      z += y[i];
    }
  }
  return z;
}
```

I. Documente qué hace *exampleFunction2* y cómo funciona.

II. Proponga una mejora. Se tomarán en cuenta factores como legibilidad y complejidad.

III. Implemente TypeScript, justificando los cambios y documentando.

5. **JavaScript**: Considere el siguiente código:

```
function fetchData() {
  let data;
  fetch('https://api.example.com/data')
    .then(response => response.json())
    .then(json => data = json);
  return data;
}

console.log(fetchData());
```

La función *fetchData* pretende hacer una petición a una API y mostrar la respuesta por consola, pero fallará. Explique por qué fallará y proponga una solución.

6. **MongoDB**: Considere una colección de MongoDB llamada “pedidos” con la siguiente estructura:

```
{
  "_id": ObjectId("123"),
  "customerId": ObjectId("456"),
  "items": [
    {
      "productId": ObjectId("789"),
      "quantity": 2
    },
    {
      "productId": ObjectId("012"),
      "quantity": 1
    }
  ],
  "orderDate": ISODate("2022-01-01T00:00:00Z")
}
```

I. Proponga mejoras para optimizar la estructura de la base de datos.

II. Proponga estrategias de indexación, explicando sus casos de uso y las queries donde se usarán. *Éstas estrategias pueden aplicarse sobre la estructura original o sobre la estructura que implemente las mejoras propuestas.*

7. **GraphQL**: Estás trabajando sobre una arquitectura de microservicios, donde cada servicio posee su propia base de datos y su propia REST API. Se te encarga proveer un endpoint para una API unificada usando GraphQL. Describa los pasos que tomaría para su implementación, y los cambios a realizar en cada servicio.