enoca java challenge

1- SAP Commerce (Hybris) nedir? Hangi amaçlarla kullanılır? Kullandığı teknolojiler nelerdir? Kısaca açıklayınız.

Hybris, müşteri deneyimi ve yönetiminde kullanılan ve SAP tarafından geliştirilmiş bir e ticaret platformudur. Çoğunlukla Java temellidir ve frontend tarafında JavaScript kullanılır. SAP ERP gibi tek bir ürün değil, uçtan uca müşteri etkileşimi deneyimi sağlayacak bir ürün grubudur. Ayrıca, SAP'nin SAP ERP ve SAP CRM gibi diğer arka uç çözümleriyle entegre edilebilir.

2- Birbirinden bağımsız iki platformun birbiriyle haberleşmesi nasıl sağlanabilir? Örneğin, X platformu Java ile yazılmış olsun, Y platform u C# ile. Bu iki platformun bir biri ile iletişim halinde request-response ilişkisi kurması gerekiyor. Bu yapıyı nasıl sağlarız? Bu iletişim sırasında güvenlik nasıl sağlanır?

Bu konuda aklıma ilk gelen kavramlar RESTful ve SOAP. RESTful ile HTTP protokolü üzerinden yine HTTP metotlarıyla ve durum kodlarıyla gelen isteklere XML, JSON olarak cevap verilebilir. SOAP da aynı şekilde HTTP protokolünün yanı sıra TCP/IP protokolünü de kullanabilir. SOAP XML haricinde hiçbir biçimde veri alışverişine izin vermez.
RESTful'un bir diğer artısı hata ayıklama aşamasındaki kolaylığı. RESTful ile HTTP hataları döndürülür ancak SOAP için bir hata ayıklama aracına ihtiyaç vardır.

Bir RESTful web servisi ile iki platform arasında request- response ilişkisi kurmak isteyelim. Bu ilişkinin request tarafı client, response tarafı server yapımız olur. Bu yapıda client, server tarafındaki veri kaynağı ile ilgili hiçbir şey bilmediği gibi, server da client hakkında bilgi sahibi değildir.

Her ikisi de HTTPS destekler. HTTPS, HTTP protokolüne SSL sertifikası eklenerek oluşturulur. Bu sertifika internet sitelerinin metinlerle kurduğu bağlantının şifrelenmesidir. HTTPS, HTTP'nin iletim katmanının güvenli hale getirilerek gizliliğin sağlanması ve istemcinin sunucunun kimliğini

doğrulaması ile çalışır. HTTPS protokolü HTTP veri paketini SSL ile şifreleyerek verilerin gizliliği, güvenliğini ve bütünlüğünü sağlar. Kısaca istemci sunucuya istek gönderdiğinde, sunucudan dönen yanıt şifrelenir ve güvenlik ihlali olduğunda dahi bu veri okunamaz. HTTP'de ise böyle bir güvenlik söz konusu değildir. Veriler açık şekilde gönderilir ve alınır.

3- SOLR Nedir? Kullanım alanlarını araştırınız. Kurumsal bir projede kullanılabilecek iki farklı kullanım alanı örneği veriniz.

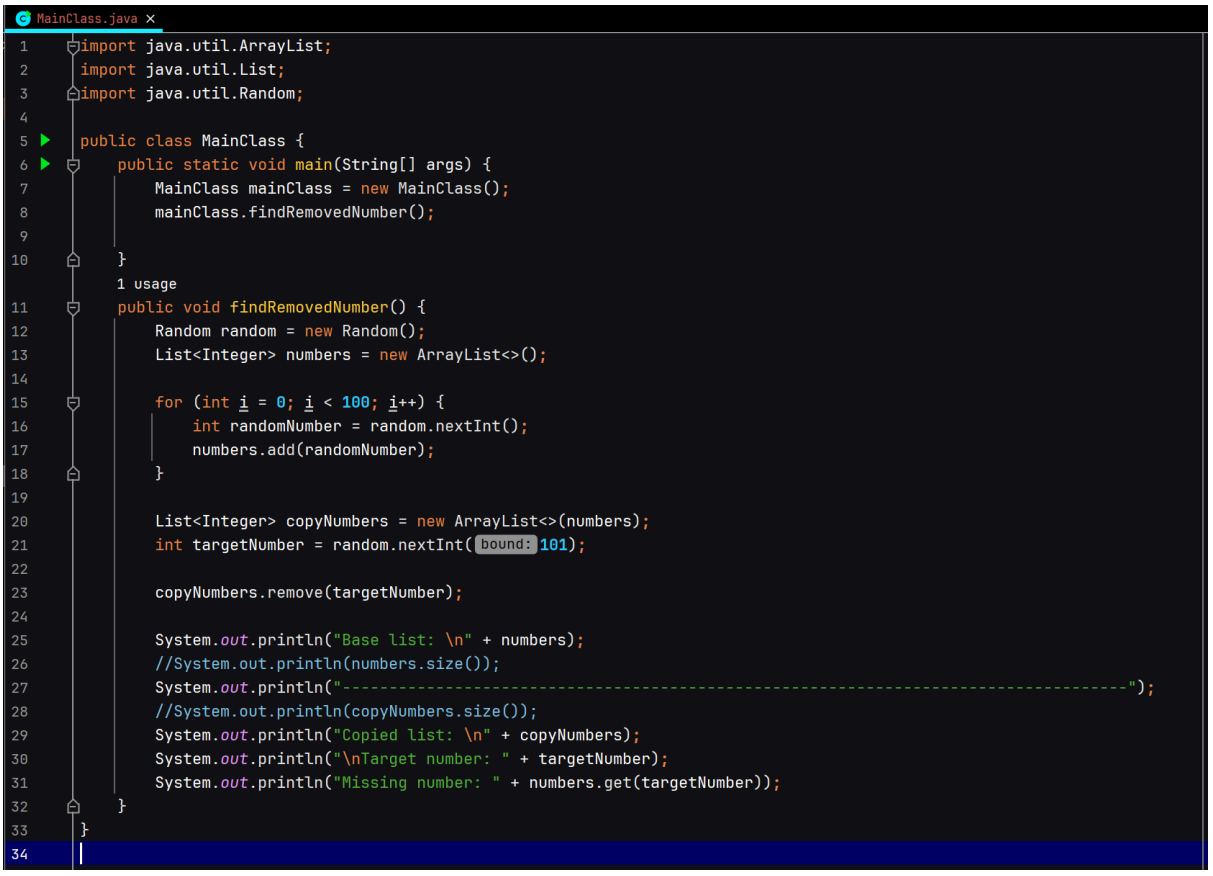
SOLR temel olarak gücünü indeksleme mekanizması sağlayan bir Java kütüphanesi olan Apache Lucene'den alan açık kaynaklı arama platformudur. Metin tabanlı verilerin aranması, dizinlenmesi ve analiz edilmesi için kullanılır. Verilerin replikasyonu ve çoklu core'lara izin vermesi gibi özelliklerle bir arama motorundan fazlasıdır. Solr aynı zamanda HTTP istekleriyle sorgular yapabileceğiniz kullanıma hazır bir web uygulamasıdır.

Bu teknolojiyi daha önce kullanmadım ancak araştırdığımda şu şekilde bir kullanım avantajı olduğunu gördüm.
`Person` gibi veritabanı varlıkları isim, adres ve email gibi alanlardan oluşabilir. Bu dokümanlar ise koleksiyonlarda saklanırlar. Koleksiyonları, geleneksel veritabanlarındaki tablolar gibi düşünebiliriz. Tek ve en önemli fark, geleneksel veritabanlarının aksine bir "Person" varlığı, birden fazla adres bilgisini aynı dökümanda barındırabilir. Geleneksel veritabanında ise `Adres` adında yeni bir tablo açıp "Person" ile ilgili ilişkiyi sağlamak gerekir.
Örnek bir Person dokümanı aşağıdaki şekilde oluşturulabilir:

```
Person {
 "Id": "1333425",
 "first_name": "Francis",
 "middle_name": "J.",
 "last_name": "Underwood",
 "address": ["1600 Pennsylvania Ave NW, Washington, DC 20500", "1609 Far St. NW, Washington, D.C., 20036"],
 "phone": ["202-456-1111", "202-456-1414"]
}
```

4- Aşağıdaki algoritma için uygun çözümü üretin.

● Java'da 100 adet random sayıya sahip bir liste oluşturun.
● Daha sonra bu listenin bir kopyasını oluşturun.
● 0 ile 100 arasında rastgele bir sayı üretin.
● Kopya listedeki bu random sayının olduğu indisteki değeri silin.
● Şimdi elinizde iki adet liste var ve kopya listede orjinal listeye göre bir eleman eksik.
● Hangi elemanın eksik olduğunu bulan bir metot oluşturun.



```java
import java.util.ArrayList;
import java.util.List;
import java.util.Random;

public class MainClass {
    public static void main(String[] args) {
        MainClass mainClass = new MainClass();
        mainClass.findRemovedNumber();

    }
    1 usage
    public void findRemovedNumber() {
        Random random = new Random();
        List<Integer> numbers = new ArrayList<>();

        for (int i = 0; i < 100; i++) {
            int randomNumber = random.nextInt();
            numbers.add(randomNumber);
        }

        List<Integer> copyNumbers = new ArrayList<>(numbers);
        int targetNumber = random.nextInt( bound: 101);

        copyNumbers.remove(targetNumber);

        System.out.println("Base list: \n" + numbers);
        //System.out.println(numbers.size());
        System.out.println("----------------------------------------------------------------------------------------");
        //System.out.println(copyNumbers.size());
        System.out.println("Copied list: \n" + copyNumbers);
        System.out.println("\nTarget number: " + targetNumber);
        System.out.println("Missing number: " + numbers.get(targetNumber));
    }
}
```

```
Base list:
[1144387339, -1180654955, -858791216, 1103314256, -1242281658, 883481248, -843656474, -996207321, 791436575, 751689468, -1546422118, -1292820499,
 1673199835, 1456805021, 872321890, -1138967141, 362200246, 2112362663, 1996086274, -989176911, -1629908441, 1447184487, 40378230, -1541160890,
 907078504, -1441115002, -831413228, 1885434464, -1474396363, 2075480955, 1969700082, -1252666491, 1717382355, 1953761089, -1023889532, 936885905,
 -781642669, -662549942, 448352138, 783794122, 958033687, -186918990, 1140323624, 906508849, -2059998059, -925310280, 563982678, -1288565222,
 1213568859, 1755853161, 169403514, -1852533505, -877820334, 440643229, -74409134, -66517360, -1622679994, -2050513113, 368465431, 479107853,
 1519511461, -68934073, 1162517983, -1444544808, 1976843398, 77800803, 751504450, -1472730299, -1176481043, 1161134838, 1711712576, -682886987,
 1888872269, -1339820856, 1598014305, -1013842462, -889844915, -1352636065, 1364239902, -795766661, 1620171982, 243764405, 1220203135, 813202163,
 209660205, 1416193066, 1574870533, -21349483, 232620229, -1595198021, 1607687113, 1480581339, 1234861751, 650391156, 842429412, -1547118848,
 -255047444, -893813725, 1657408785, 1846704852]
--------------------------------------------------------------------------------
Copied list:
[1144387339, -1180654955, -858791216, 1103314256, -1242281658, 883481248, -843656474, -996207321, 751689468, -1546422118, -1292820499, 1673199835,
 1456805021, 872321890, -1138967141, 362200246, 2112362663, 1996086274, -989176911, -1629908441, 1447184487, 40378230, -1541160890, 907078504,
 -1441115002, -831413228, 1885434464, -1474396363, 2075480955, 1969700082, -1252666491, 1717382355, 1953761089, -1023889532, 936885905, -781642669,
 -662549942, 448352138, 783794122, 958033687, -186918990, 1140323624, 906508849, -2059998059, -925310280, 563982678, -1288565222, 1213568859,
 1755853161, 169403514, -1852533505, -877820334, 440643229, -74409134, -66517360, -1622679994, -2050513113, 368465431, 479107853, 1519511461,
 -68934073, 1162517983, -1444544808, 1976843398, 77800803, 751504450, -1472730299, -1176481043, 1161134838, 1711712576, -682886987, 1888872269,
 -1339820856, 1598014305, -1013842462, -889844915, -1352636065, 1364239902, -795766661, 1620171982, 243764405, 1220203135, 813202163, 209660205,
 1416193066, 1574870533, -21349483, 232620229, -1595198021, 1607687113, 1480581339, 1234861751, 650391156, 842429412, -1547118848, -255047444,
 -893813725, 1657408785, 1846704852]

Target number: 8
Missing number: 791436575

Process finished with exit code 0
```