1. SOAP vs Restful?

Rest, JSON, HTML ve XML gibi veri tiplerini desteklediğinden sadece XML tabanlı SOAP'a göre daha esnek ve kullanışlıdır.

Soap, isteklerini oluşturmak için W3C standartı olan bir WSDL dili kullanmak zorundadır. Rest'te buna gerek yoktur istekler URI üzerinden oluşturulabilir. Bir dile ihtiyaç duymadığından REST'in kullanımı ve tasarlanması daha kolaydır.

Güvenlik açısından SOAP için hazır fonksiyonlar bulunmaktadır ve daha kompleks oluşu onu bu konuda daha avantajlı yapar.

Her ikisi de HTTP protokolünü kullanır ve server-client arasındaki haberleşmeyi sağlar. Ancak REST için HTTP şarttır SOAP ise TCP/IP STMP gibi protokollere de destek verebilir.

Cachleme isteği için REST basit HTTP metodunu kullanır ve SOAP'a göre daha kolaydır. SOAP, bu işlem için karışık XML requestleri yapmaktadır.

2. Difference between acceptance test and functional test?

Functional test, ortamdan ve yan etkilerden bağımsız olarak sadece bir fonksiyonun test edilmesini sağlar. Acceptance test ise yazılımdan beklenen işlemlerin yapılıp yapılamadığını kontrol eder.

3. What is Mocking?

Mocking(mocklama), popüler yazılım metodolojisi olan **TDD** ve özelde birim testlerinin (unit test), test ettikleri sistemi izole etmede kullandığı yöntemlerden biridir. Bu yöntemler, geniş anlamıyla **test dublörleri (test double)** olarak tanımlanabilir. Test dublörleri, test edilen sistemin bağımlı olduğu diğer birimlerin yerini tutar. Bu izolasyona birim testlerinde ihtiyaç duyulmasının temelde iki sebebi vardır:

1. Birim testleri, genelde test ettikleri sistemin kendisi ile ilgili varsayımları doğrulamak için yazılır.
2. Test dublörleri, davranış ve kullanım şekillerine göre çeşitlenir. Bunlardan en çok kullanılanları dummy, fake, stub, spy ve mock'tur denebilir. Bu çeşitliliğe sebep olan genel faktörler, bu dublörlerin beklenen işi yapıp yapmadığı ve yaparken nasıl bir davranış gösterdiği ile ilgilidir.

4. What is a reasonable code coverage % for unit tests (and why) ?

Genel olarak %80 kapsamı hedeflemek iyi bir hedefitr. Daha yüksek bir yüzdeye ukaşmaya çalışmak, yeterli fayda sağlamak için gerekli olmasa da maliyetlidir.

5. HTTP/POST vs HTTP/PUT?

POST kaynağa veri göndermek için kullanılır. PUT ise aynı kaynağa aynı adres ile erişilir ve eğer içerik var ise gelen veriler ile değiştirilir, eğer içerik yok ise yeni içerik yaratılır. Kısaca PUT veri güncellemek için kullanılır.

6. What are the Safe and Unsafe methods of HTTP?

Safe methods: **GET, HEAD, OPTIONS**

Unsafe methods: **PUT, DELETE, POST**

7.How does http Basic Authentication work?

1. Bir client, korunan bir kaynağa erişim ister.

2. Web sunucusu, kullanıcı adı ve parola isteyen bir iletişim kutusu döndürür.

3. İstemci, kullanıcı adı ve parolayı sunucuya gönderir.

4. Sunucu, belirtilen alanda kullanıcının kimliğini doğrular ve başarılı olursa istenen kaynağı döndürür.

8. Define RestTemplate in Spring?

RestTemplate, istemci tarafı HTTP erişimi için merkezi Spring sınıfıdır. Kavramsal olarak, JdbcTemplate, JmsTemplate ve Spring Framework ve diğer portföy projelerinde bulunan diğer çeşitli şablonlara çok benzer. Bu, örneğin, RestTemplate oluşturulduktan sonra iş parçacığı için güvenli olduğu ve işlemlerini özelleştirmek için geri aramaları kullanabileceğiniz anlamına gelir.

Tanımlamaları da şu şekildedir;

HTTP	RestTemplate
DELETE	`delete(String, String...)`
GET	`getForObject(String, Class, String...)`
HEAD	`headForHeaders(String, String...)`
OPTIONS	`optionsForAllow(String, String...)`
POST	`postForLocation(String, Object, String...)`
PUT	`put(String, Object, String...)`

9. What is the difference between @Controller and @RestController?

Controller'lar class düzeyinde @Controller anotasyonu ile tanımlanıyor. Bu da aslında class düzeyinde ekstradan @Component anotasyonunu yazmaktan bizi kurtarıyor ve bu sınıfın bir Controller olduğunu belirtiyor. @Controller ile oluştuğumuz endpoint'ler içerisinde Model class'ı ile objeleri Thymeleaf gibi template engine üzerinden göstermemizi sağlıyor

RestController ise @Controller ve@ ResponseBody'nin birleşiminden oluşan bir streotype. Dolaylı olarak da @Controller üzerinden @Component anotasyonunu da barındırıyor. @Controller

anotasyonunun aksine @RestController datanın kendisini JSON veya XML formatı ile direkt olarak sunabiliyor.

10. What is DNS Spoofing? How to prevet?

DNS Spoofing, sahte verilerin DNS çözümleyicinin önbelleğine girdiği ve ad sunucusunun yanlış bir IP adresi döndürmesine neden olan bir siber saldırıdır. Başka bir deyişle, bu tür saldırılar, alan adı sunucularındaki güvenlik açıklarından yararlanır ve trafiği gayri meşru web sitelerine yönlendirir.

Bu saldırıları önlemek için DNSSEC kurulumu yapılabilir.

11. What is content negotiation?

Content negotiation, kullanıcı aracısının kullanıcı için hangi gösterimin en uygun olduğunu (örneğin, hangi belge dili, hangi görüntü formatı veya hangisi) belirlemesine yardımcı olmak için bir kaynağın farklı temsillerini aynı URI'ye sunmak için kullanılan mekanizmadır.

12. What is statelessness in RESTful Web Services?

REST mimarisine göre, bir RESTful Web Hizmeti, sunucuda bir istemci durumu tutmamalıdır. Bu kısıtlamaya **statelessness** denir. Bağlamını sunucuya iletmek istemcinin sorumluluğundadır ve ardından sunucu, müşterinin daha sonraki isteğini işlemek için bu bağlamı saklayabilir. Örneğin, sunucu tarafından sürdürülen oturum, istemci tarafından geçirilen oturum tanımlayıcısı tarafından tanımlanır.

13. What is CSRF attack? How to prevent?

CSRF attack, bir saldırganın bir kurbanı kendi adına eylemler gerçekleştirmesi için kandırdığı bir saldırıdır. Saldırının etkisi, kurbanın sahip olduğu izinlerin düzeyine bağlıdır. Bu tür saldırılar, bir web sitesinin kullanıcıya tamamen güvendiği gerçeğinden yararlanır ve bu, kullanıcının gerçekten söylediği kişi olduğunu onaylar.

14. What are the core components of the HTTP request and HTTP response?

GET -> GET yöntemi, belirli bir URI kullanılarak verilen sunucudan bilgi almak için kullanılır. GET kullanan istekler yalnızca verileri almalı ve veriler üzerinde başka bir etkisi olmamalıdır.

HEAD -> GET ile aynıdır, ancak yalnızca durum satırını ve başlık bölümünü aktarır.

POST -> HTML formlarını kullanarak örneğin müşteri bilgileri, dosya yükleme vb. gibi verileri sunucuya göndermek için bir POST isteği kullanılır.

PUT -> Hedef kaynağın tüm mevcut temsillerini yüklenen içerikle değiştirir.

DELETE -> Bir URI tarafından verilen hedef kaynağın tüm mevcut temsillerini kaldırır.

CONNECT -> Belirli bir URI tarafından tanımlanan sunucuya bir tünel oluşturur.

OPTIONS -> Hedef kaynak için iletişim seçeneklerini açıklar.

TRACE -> Hedef kaynağa giden yol boyunca bir mesaj geri döngü testi gerçekleştirir.