AHMET BERA KANSU

Q1- Java dünyasındaki framework'ler ve çözdükleri problemler nedir? Kod Örneklendirini de içermelidir.

Spring Framework: Spring, Java uygulamalarının geliştirilmesi ve yönetilmesi için kapsamlı bir çözüm sunar. Bağlantı yönetimi, veritabanı işlemleri, güvenlik, iş akışı yönetimi gibi birçok alanı kapsar. Ayrıca, Spring Boot, Spring projelerinin hızlı bir şekilde başlatılmasını ve yapılandırılmasını sağlar.

```
// Spring ile basit bir Restful Web Servis örneği
@RestController
public class HelloWorldController {
    @RequestMapping(value = "/hello", method = RequestMethod.GET)
    public String helloWorld() {
        return "Hello, World!";
    }
}
```

Hibernate: Hibernate, Java nesnelerini ilişkisel veritabanlarıyla eşlemek için kullanılır. Nesne ilişkisel eşleme (ORM) sağlar, böylece veritabanı işlemleri için SQL sorguları kullanmak yerine Java nesneleri kullanılabilir.

```
// Hibernate ile basit bir entity ve ilişkilendirme örneği
@Entity
public class Product {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    private String name;
    // Diğer alanlar ve getter/setter metotları...
}
@Entity
public class Order {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    @OneToMany
    private List<Product> products;
    // Diğer alanlar ve getter/setter metotları...
}
```

Quarkus: Java tabanlı mikro servislerin hızlı bir şekilde geliştirilmesi için optimize edilmiş bir framework'tür. Özellikle Kubernetes, Docker ve serverless gibi modern uygulama geliştirme ve dağıtım senaryolarına odaklanmıştır.

```
@Path("/hello")
public class MyResource {
    @GET
    @Produces(MediaType.TEXT_PLAIN)
    public String hello() {
        return "Hello, Quarkus!";
    }
}
```

Q2- SOA - Web Service - Restful Service - HTTP methods kavramlarını örneklerle açıklayınız.

SOA (Service-Oriented Architecture - Hizmet Odaklı Mimarlık):

SOA, yazılım sistemlerini birbiriyle bağlayan, hizmetlerin modüler bir şekilde oluşturulmasını ve kullanılmasını sağlayan bir mimari yaklaşımdır. Hizmetler, genellikle işlevsellik veya yeteneklere dayalı olarak tanımlanır ve farklı uygulamalar arasında veri ve işlem akışını kolaylaştırır.

Örnek: Bir bankacılık uygulaması, müşteri bilgilerini almak, hesap bilgilerini görmek veya para transferi yapmak gibi farklı işlevlere sahip olabilir. Her bir işlev, bir hizmet olarak sunulabilir ve diğer uygulamalar bu hizmetlere erişebilir.

Web Service (Web Hizmeti):

Web servisleri, farklı platformlardaki uygulamalar arasında iletişim kurmak için kullanılan bir yazılım bileşenidir. Web servisleri, HTTP protokolü üzerinden erişilebilen ve genellikle XML veya JSON gibi belirli veri formatlarını kullanarak iletişim kuran hizmetlerdir.

Örnek: Bir hava durumu servisi, bir web sitesi veya mobil uygulama tarafından kullanılarak kullanıcılara güncel hava durumu bilgilerini sağlayabilir.

RESTful Service (RESTful Hizmet):

RESTful hizmetler, Representational State Transfer (Temsil Durumu Aktarımı) kavramına dayanan bir web hizmeti türüdür. RESTful hizmetler, kaynakları benzersiz URI'lerle temsil eder ve HTTP protokolünü kullanarak bu kaynaklara erişim sağlar. HTTP metodları (GET, POST, PUT, DELETE vb.) kullanılarak kaynaklar üzerinde işlemler gerçekleştirilir.

Örnek: Bir blog platformu, RESTful bir hizmet aracılığıyla makaleleri, yorumları ve kullanıcı profillerini yönetebilir. Makale oluşturma (POST), makaleyi güncelleme (PUT), makaleyi silme (DELETE) gibi işlemler HTTP metodları aracılığıyla gerçekleştirilebilir.

HTTP Methods (HTTP Metodları):

HTTP protokolü, bir sunucu ile bir istemci arasındaki iletişimi sağlar. Bu iletişim, HTTP metotları olarak adlandırılan belirli eylemler aracılığıyla gerçekleşir. En yaygın kullanılan HTTP metotları şunlardır:

- GET: Belirtilen URI'deki kaynağı almak için kullanılır. (Örneğin, web sayfası veya dosya)
- POST: Belirtilen URI'ye yeni veri göndermek veya kaynak oluşturmak için kullanılır. (Örneğin, bir form gönderme)
- PUT: Belirtilen URI'deki mevcut kaynağı değiştirmek veya güncellemek için kullanılır.
- DELETE: Belirtilen URI'deki kaynağı silmek için kullanılır.
- PATCH: Belirtilen URI'deki kaynağın bir kısmını güncellemek için kullanılır.
- OPTIONS: Belirtilen URI'nin desteklediği HTTP metotlarını almak için kullanılır.

Örnek: Bir kullanıcı bir web sitesine girdiğinde, tarayıcı GET isteği gönderir ve web sitesinin ana sayfasını alır. Kullanıcı bir form doldurup gönderdiğinde, tarayıcı POST isteği gönderir ve sunucuya form verilerini iletir.