## 1 – IOC and DI means ?

Nesneler arası bağımlılıkların kontrol altına alınması gerekir. Bağımlılıkların uygulama tarafından kontrol edilmesi yerine bu kontrolün kullanılan uygulama framework'üne devredilmesi olayına Inversion of Control denir.

Dependecy Injection uygulayarak bir sınıfın bağımlı olduğu nesneden bağımsız olarak hareket edilmesi sağlanır v kod üzerinde olası geliştirmelere karşı değişiklik yapma ihtiyacı ortadan kaldırılır.

## 2 – Spring Bean Scopes ?

Bean: Spring IoC container tarafından yönetilen nesnelere denir. Metod düzeyinde kullanılır. Eğer bir metod @Bean ile tanımlıysa Spring tarafından yönetilen bir Bean ürettiğini belirtmiş olur.

Scope: Bir kapsam alanı gibi düşünülebilir. 5 çeşit Scope kavramı vardır.

Singleton : Bir Bean oluşturduğumuzda varsayılan olarak er zaman Singleton Scope kulanılır. Uygulama yaşam döngüsü oyunca aynı olacak bir sigleton nesnesi oluşturur. Her inject edildiğinde aynı instance döndürür. Varsayılan olduğu için ayrıca belirtilmeye gerek yoktur.

Protype : Her request geldiğindeyei bir instance döndürür. Her inject edildiğinde yeni bir obje oluşturur. Kullanıldığında belirtilir.

Request : HTTP isteği geldiği zaman oluşturulur. API isteği yanıtı geri gönderene kadar nesnenin aynı instance'ı alınır ancak yeni request geldiği zaman yeni instance gönderilir.

Session : HTTP isteği geldiği zaman oluşturulur. Kullanıcı oturumu kapandığında yeni kullanıcı oturumu için nesnenin yeni bir instance'ı alınır.

Global Session : Bir HTTP'nin yaşam döngüsünde tek bir Bean'ın tanımını kapsar.

## 3 – What Does @SpringBootApplication?

Uygulamanın giriş metodunu belirtir. Bir ya da daha fazla @Bean yönetimi bildiren ve ayrıca otomatik yapılandırmayı ve bileşen taramayı tetikleyen bir yapılandırma sınıfını işaretlemek için kullanılır.

## 4 – Why Spring Boot over Spring?

Spring Boot kodu kısaltmaya ve Spring uygulamasını kolayca çalıştırmaya odaklanmıştır. En temel amacı birim-entegrasyon testleri için varsayılan bir kurulum sağlayarak geliştirme süresini azaltmak ve verimi arttırmaktır. Spring Boot servis uçları REST kullanımı sunar ve güçlü bir REST yapılandırmasına sahiptir. XML yapılandırmasına gerek yoktur.

## 5 – What is Singleton and where to use it?

Singleton Design Pattern nesne oluşturma mantığı üzerine yoğunlaşmıştır. Bir class'tan tek bir nesne üretilmesi ve sistemde anlık olarak bu nesnenin bulunması istenen durumlarda Singleton yaklaşımı tercih edilir.

## 6 – Explain @RestController annotation in Sprint boot?

Restful web servisleri yapmak için kulanılır. Bu anotasyon sınıf düzeyinde kullanılır ve sınıfın istemci tarfından yapılan istekleri işlemesine izin verir.

## 7 – What is the primary difference between Spring and Spring Boot?

Spring ile çalışırken bir web sunucusu üzerinde çalışmak gerekir ancak Spring Boot içine eklenmiş sunucu ile gelir ve ekstra kurulum gerektirmez.

Spring Boot starter paketleri ile birlikte proje tipine göre hazır kütüphaneler ile gelir. Springte ise core, security, web, data tarafları manuel olarak eklemek gerekir.

## 8 – Why to use VCS?

Versiyon Kontrol Sistemleri (Version Control Systems) depo (repository) içindeki değişiklikleri sürümler halinde tutabilmemize ve bu değişiklikleri kolayca yönetebilmemize imkan sunan sistemlerdir.

Ekip olarak geliştirilen bir üründe aynı modüller üzerinde paralel olarak geliştirme yapmamız gerektiğinde bu sistemler çok daha önem ifade eder.

## 9 – What are SOLID Principles ? Give sample usages in Java?

Nesne yönelimli programlamanın 5 prensibinin baş harflerinin oluşturduğu bir kısaltma olan SOLID ile projenin gelecekte yapılacak geliştirmelere rahatlıkla adapte olabilmesi, geliştirme yaparken yazılmış yerleri değiştirmeden yeni eklemelere açık olması, anlaşılır kod yazılması, değişiklik yapılırken bir yerlerde istemeden çıkan hataların önüne geçilmesi amaçlanmıştır.

- *Single Responsibilit Principle :* Metod veya sınıf sadecetek bir iştn sorumlu olmalı. Örneğin bir üye olma metodu var ve içinde mail de atılabiliyorsa bunlar ayrı metodlarda yazılmalı.
- *Open / Closed Principle :* Bir metod veya sınıfa yeni davranışlar eklenmesine açık ancak temel yapısının değişikliğe kapalı olması prensibidir. Örneğin bir bankada var olan kredilere ek yeni bir kredi türü eklenebilmeli ancak bankanın temel sistemi değiştirilemez olmalı.
- *Liskov Substitution Principle :* Bu prensibin amacı türeyen alt sınıfların üst sınıfın tüm özelliklerini ve metodlarını aynı işlevi gösterecek şekilde kullanabilme ve kendine ait yeni özellikler alabilmesidir.Örneğin bir hayvan sınıfımız olsun bu sınıftan türeyen kuş sınıfı olsun. Hayvan sınıfındaki özellikleri kullanabilirken kuş sınıfında ekstra ötme metodu olabilir.
- *Interface Segregation Principle :* Bir interface içinde tüm özellikleri toplamak yerine her interface özelleştirilmeli ve onunla bağdaşan sorumlulukları gereksiz bir sorumluluk kalmayacak şekilde sınıfa implement edilmesini isteyen prensiptir. Örneğin bir kargo uçağı sınıfımız olsun. Bu sınıfta yangın söndürme gibi alakasız metod olmamalı.
- *Dependency Inversion Principle :* Bu prensipte üst seviyeli sınıfların alt seviyeli sınıflara bağımlılığı ortadan kaldırılarak her iki sınıfında abstract kavramlara bağımlı olması hedeflenir. Örneğin bir yedekleme sınıfımız olsun önce bir interface ile yazma metodu oluşturulur ve yedekleme istenen farklı yerlerde de rahatlıkla kullanılabilir. Bunun için yedekleme interface'i kullanılacak sınıfa implemet edilmeli ve yazma metodu override edilerek kullanılır.

## 10 - What is RAD model?

Hedefi en kısa zamanda ve düşük maliyetle mümkün olan en az insan kaynağı ile projenin isteklerinin adım adım ortaya çıkrıldığı bir prototip geliştirme yöntemidir. Bir RAD projesi sonucunda netleşmiş sistem konsept dökümanı, netleşmiş istemler ve süreçler lstesi, istemler ışığında geliştirilmiş prototip sistem olarak üç ana çıktı elde edilir.

## 11 - What is Spring Boot starter ? How is it useful?

Manuel ekleme için gereken bağımlılıkların sayısını azaltmaya yardımcı olan bir grup bağımlılık. Yaygın olan başlatıcılar;

Spring – boot – starter

Spring – boot – starter – data – jpa

Spring – boot – starter – web

Spring – boot – starter – security

## 12 – What are the Spring Boot Annotations?

Annotation ( Anatosyan / notasyon ): Development anında IDE veya compiler tarafından ya da run-time anında framework tarafından yorumlanan ifadelerdir.

*Kısıtlamaları ve kullanımını tanımlayan: @Deprecated @Override @NotNull*

*Bir öğenin çalışma yapısını belirtenler: @Entity @TestCase @WebService*

*Bir öğenin davranışını belirtenler: @Statefull @Transaction*

*Bir öğenin nasıl işleneceğini belirtenler: @Column @XmlElement*

*@Bean:* Bir metodun spring tarafından yönetilen bir Bean üretiğini belirtir.
*@Service:* Belirtilen sınıfın bir servis sınıfı olduğunu belirtir.
*@Repository:* Veritabanı işlemlerini gerçekleştirme yeteneği olan repository sınıfını belirtir.
*@Configuration:* Bean tanımlamaları gibi tanımlamalar için bir Bean sınıfı olduğunu belirtir.

*@Controller:* Requestleri yakalayabilme yeteneği olan bir web controller sınıfını belirtir.
*@RequestMapping:* Conroller sınıfının handle ettiği HTTP requestlerinin path eşleştirmesini yapar.
*@Autowired:* Constructo, değişken ya da setter metodlar için dependecy injection işlemini gerçekleştirir.
*@SpringBootApplication:* Spring Boot autoconfiguration ve component taramasını aktif eder.

## 13 – What is Spring Boot dependency management?

İlgili Sping Boot sürümüne göre gerekli tüm bağımlılıkları tek bir yerde toplamaya yarar. Spring Boot sürümü belirtilebilir veya değiştirilebilir.Sürüm değiştirildiğinde eklenen bağımlılıklarını tüm sürümleri otomatik olarak güncellenir. 'Multı-Module' projeleri için faydalı olan farklı Spring Boot kitaplıkları sürümlerinin çakışmasını önler.

## 14 - What is Spring Boot Actuator?

Projenin çalışma durumu, trafik durumu, veritabanı durumu gibi bilgilere endpointler yardımıyla kolay erişmeyi sağlar. Aktuatöün sunduğu tüm özelliklere ihtiyaç duyulmadığı zaman application.properties'ten ilgili özellik değeri false yapılır.