**SPRİNG DEPENDENCY**

ACTUATOR: Uygulamanın çalışma zamanı durumu ve davranışı hakkında bilgi sağlayan birçok özellik sunar.

DEVELOPER TOOLS: Geliştirme sürecini kolaylaştıran araçlar içerir.

GraalVM Native Support: GraalVM ile uyumluluğu sağlar, Java uygulamalarının native olarak derlenmesine olanak tanır. GraalVM, Oracle tarafından geliştirilen ve Java Virtual Machine (JVM) üzerinde çalışan çok dilli bir sanal makinedir. GraalVM, Java, Scala, Kotlin, Groovy gibi JVM dilleri yanı sıra JavaScript, Python, Ruby ve R gibi dilleri de destekler. Java uygulamalarının native olarak derlenmesi, Java kodunun Java Sanal Makinesi (JVM) üzerinde çalıştırılmak yerine, doğrudan platformun makine diline derlenerek çalıştırılmasını ifade eder.

Graphql DGS Code Generation: GraphQL API'lerini oluşturmak için kullanılan bir çerçevedir. Geleneksel GraphQL API geliştirme sürecinde, genellikle şema (schema) tanımlama, sorgu işleme, veri kaynağı entegrasyonu gibi adımlar manuel olarak yapılır. Ancak DGS, bu süreci otomatikleştirir ve geliştiricilere daha hızlı bir şekilde GraphQL API'leri oluşturma imkanı sağlar.

Spring Boot DevTools: Geliştirme sırasında otomatik yeniden yükleme ve diğer geliştirici araçları sağlar.

Lombok: Getter, setter, equals, hashCode gibi tekrarlayıcı kodları otomatik olarak üretir.

Spring Configuration Processor: Yapılandırma sınıflarını işlemek için yardımcı araçlar sağlar.

Docker Compose Support: Docker Compose ile Spring Boot uygulamalarının entegrasyonunu sağlar.

Spring Modulith: Spring tabanlı monolitik uygulamaları oluşturmayı kolaylaştırır.

WEB: Web uygulamaları geliştirmek için gereken araçlar.

Spring Web: Web uygulamaları oluşturmak için temel Spring desteği.

Spring Reactive Web: Reaktif web uygulamaları oluşturmak için destek sağlar. Senkron web uygulamalarının aksine hızlı, esnek ve ölçeklenebilir uygulamalar geliştirmeyi amaçlar.

Spring for GraphQL: GraphQL API'leri oluşturmak için destek sağlar.

Rest Repositories: Spring Data REST ile entegre bir şekilde RESTful web servisleri oluşturmayı sağlar.

Spring Session: HTTP oturum yönetimi için destek sağlar.

Spring Web Services: SOAP tabanlı web servisleri oluşturmak için destek sağlar.

Spring HATEOAS: HATEOAS (Hypermedia as the Engine of Application State) prensibine uygun RESTful servisler oluşturmayı sağlar. HATEOAS, RESTful API tasarımının bir prensibidir. İstemci bir kaynağa ulaştığında, bu kaynağın üzerinde yapabileceği işlemler ve bu işlemleri gerçekleştirebileceği diğer kaynaklar da istemciye sunulur. Bu sayede, istemci, API'yi daha dinamik bir şekilde keşfedebilir ve etkileşimde bulunabilir.

TEMPLATE ENGINES: Web sayfaları oluşturmak için farklı şablon motorları sağlar.

Thymeleaf: Web sayfalarını oluşturmak için Java tabanlı bir şablon motoru.

Apache Freemarker: Apache Freemarker şablon motoru.

SECURITY: Güvenlikle ilgili araçlar ve kütüphaneler.

Spring Security: Uygulama güvenliği için kapsamlı bir çözüm.

OAuth2 Client: OAuth 2.0 istemci desteği sağlar.

OAuth 2.0, kullanıcıların farklı uygulamalara erişim yetkilerini kontrol etmek için kullanılan bir kimlik doğrulama ve yetkilendirme protokolüdür.

OAuth2 Authorization Server: OAuth 2.0 yetkilendirme sunucusu oluşturmayı sağlar.

Spring LDAP: LDAP entegrasyonu için destek sağlar. (Lightweight Directory Access Protocol), kullanıcı kimlik doğrulama, yetkilendirme ve dizin hizmetleri için kullanılır. Genellikle kullanıcı bilgilerinin (kullanıcı adları, şifreler, roller vb.) merkezi bir şekilde yönetilmesini ve dağıtılmasını sağlar.

SQL:

JDBC API: Java Database Connectivity Application Programming Interface, Java programlarının veritabanlarına erişmesini sağlayan bir API'dir. JDBC, Java programlarının farklı veritabanlarına bağlanmasını, sorgu çalıştırmasını ve sonuçları işlemesini sağlar.

JDBC (Java Database Connectivity): JDBC, Java programlarının veritabanlarına erişimini sağlayan bir Java API'sidir. JDBC, veritabanı sorgularını çalıştırmak, sonuçları almak ve işlemek için kullanılır.

Spring Data JPA: Java Persistence API, JPA, nesne ilişkisel eşleme (ORM) için bir standarttır ve ilişkisel veritabanlarına erişimi kolaylaştırır.

Spring Data JDBC: Spring Data JDBC, Spring Data projesinin bir parçasıdır ve JDBC kullanarak veritabanlarına erişimi sağlar.

Spring Data R2DBC: Reactive Relational Database Connectivity, R2DBC, asenkron olarak veritabanı işlemlerini gerçekleştirmeyi destekler.

H2 Database: Hafif bir SQL veritabanı motorudur. H2, Java tabanlıdır ve genellikle testlerde ve küçük ölçekli uygulamalarda kullanılır.

Mysql Driver: MySQL veritabanına Java programları tarafından erişmek için kullanılan bir JDBC sürücüsüdür.

Postgresql Driver: Postgresql veritabanına Java programları tarafından erişmek için kullanılan bir JDBC sürücüsüdür.

Oracle Driver: Oracle veritabanına Java programları tarafından erişmek için kullanılan bir JDBC sürücüsüdür.

NOSQL:

Spring Data Redis: Redis veritabanına erişimi kolaylaştıran Spring Data modülüdür.

Spring Data MongoDB: MongoDB veritabanına erişimi kolaylaştıran Spring Data modülüdür.

Spring Data Reactive Redis: Reactive Programming prensiplerini kullanarak Redis veritabanına erişimi sağlayan Spring Data modülüdür.

Reactive Programming: Reactive Programming, veri akışlarını ve olayları kolayca işleyebilmek, hata durumlarını ele almak ve sistemler arasında verimli bir iletişim sağlamak için kullanılır.

Spring Data Elasticsearch: Elasticsearch arama motoruna erişimi sağlayan Spring Data modülüdür.

Spring Data Couchbase: Couchbase NoSQL veritabanına erişimi sağlayan Spring Data modülüdür.

MESSAGING:

Spring Integration: Uygulamalar arasında veri akışını kolaylaştıran bir entegrasyon çerçevesidir.

Spring for RabbitMQ: RabbitMQ mesaj kuyruğu sistemine entegrasyonu kolaylaştıran Spring modülüdür.

RabbitMQ: RabbitMQ, bir mesaj kuyruğu yazılımıdır. Dağıtılmış sistemler arasında veri iletimini kolaylaştırmak için kullanılır. Üreticilerden mesajları alır, bunları kuyrukta saklar ve tüketicilere ileterek iletişim sağlar.

Spring for Apache Kafka: Apache Kafka mesaj sistemiyle entegrasyonu kolaylaştıran Spring modülüdür.

Websocket: Websocket, çift yönlü, gerçek zamanlı iletişim sağlayan bir iletişim protokolüdür.

I/O:

Spring Batch: Toplu işleme (batch processing) uygulamaları geliştirmeyi kolaylaştıran Spring Batch çerçevesidir.

Toplu İşleme (Batch Processing): Toplu işleme, büyük veri setlerinde işlemleri gruplar halinde işleyerek performansı artıran bir işleme modelidir. Örneğin, büyük miktardaki verilerin toplanması, işlenmesi ve analiz edilmesi gibi işlemler toplu işleme modeliyle gerçekleştirilir.

Validation: Veri doğrulama işlemlerini kolaylaştıran bir Java API'sidir.

Java Mail Sender: Java Mail API'yi kullanarak e-posta göndermeyi sağlayan bir Spring modülüdür.

OPS:

Spring Boot Actuator: Spring Boot uygulamalarının çalışma zamanı bilgilerini ve metriklerini toplamayı sağlayan bir modüldür.

Sentry: Uygulama hatalarını izlemeyi ve raporlamayı sağlayan bir hata izleme platformudur.

OBSERVABILITY:

Datadog: Uygulama izleme ve performans yönetimi için bir platformdur.

Dynatrace: Uygulama performansını izlemeyi ve analiz etmeyi sağlayan bir gözlem platformudur.

Influx: Ölçümleri ve olayları depolamak ve sorgulamak için bir zaman serisi veritabanıdır.

Graphite: Metrikleri izlemek ve görselleştirmek için bir zaman serisi veritabanıdır.

TESTING:

Spring REST Docs: RESTful API'ler için belgelendirme otomatikleştirmeyi sağlayan bir Spring modülüdür.

Testcontainers: Docker konteynerlerini kullanarak entegrasyon testleri oluşturmayı sağlayan bir kütüphanedir.

SPRING CLOUD:

Cloud Bootstrap: Spring Cloud uygulamalarını başlatmak ve yapılandırmak için kullanılan bir modüldür.

Function: Serverless uygulamalar geliştirmek için Spring Cloud Function kullanılır.

Serverless uygulamalar, geleneksel sunucu tabanlı uygulama geliştirme modelinden farklı bir yaklaşımı temsil eder. Bu modelde, uygulama kodu, altyapı yönetimi ile ilgilenmeden çalıştırılır. Serverless uygulamalar, bulut bilişim sağlayıcıları tarafından yönetilen işlev hizmetlerine dayanır. İşlev hizmetleri, belirli bir olay veya istekle tetiklenen ve belirli bir işlevi gerçekleştiren parçalardır.

Serverless modelde, geliştirici, kendi kodunu yazarak uygulamanın mantığını oluşturur. Ardından, bu kodu bir işlev hizmetine yükler. İşlev hizmetleri, uygulamanın çalıştırılacağı zaman ve kaynakları otomatik olarak yönetir. İşlev hizmetleri, gerektiğinde otomatik olarak ölçeklendirilir ve sıfır bakım gerektirir. Bu sayede, geliştiriciler altyapı ile ilgilenmek zorunda kalmadan hızlı bir şekilde uygulama geliştirebilir ve dağıtabilirler.

SPRING CLOUD CONFIG:

Config Client: Spring Cloud Config sunucudan yapılandırma bilgilerini almayı sağlayan bir modüldür.

Config Server: Spring Cloud Config, uygulama yapılandırma bilgilerini depolamak ve sunmak için kullanılır.

SPRING CLOUD ROUTING:

Gateway: Spring Cloud Gateway, API yönlendirme işlemlerini sağlayan bir modüldür.