|  |  |
| --- | --- |
| **Spring** | **SpringBoot** |
| Spring is an open-source lightweight framework widely used to develop enterprise applications.  The most important feature of the Spring Framework is dependency injection.  To run the Spring application, we need to set the server explicitly.  It helps to create a loosely coupled application.  Developers have to define dependencies manually in the pom.xml file.  It helps to create a stand-alone application. | Spring Boot is built on top of the conventional spring framework, widely used to develop REST APIs.  The most important feature of the Spring Boot is Autoconfiguration  Spring Boot provides embedded servers such as Tomcat and Jetty etc.  It helps to create a stand-alone application.  pom.xml file internally handles the required dependencies.  It provides support for the in-memory database such as H2. |

---------------------------------------------------------------------------------------------------------------------------------

|  |  |
| --- | --- |
| **JPA** | **Hibernate** |
| JPA is a specification and defines the way to manage JPA is a specification and defines the way to manage.  JPA uses javax.persistence package.  JPA uses EntityManagerFactory interface(& also to create/read/delete operation) to get the entity manager to persist objects.  JPA uses JPQL (Java Persistence Query Language) as Object Oriented Query language for database operations. | Hibernate is an implementation of JPA, it is an ORM tool to persist Hibernate is an implementation of JPA, It is an ORM tool to persist.  Hibernate uses org.hibernate package.  Hibernate uses SessionFactory interface(& also to create/read/delete operation) to create session obj which is then used to persist objects.  Hibernate uses HQL (Hibernate Query Language) as Object Oriented Query language for database operations. |

---------------------------------------------------------------------------------------------------------------------------------

|  |  |
| --- | --- |
| **BeanFactory** | **ApplicationContext** |
| **Basic IoC Container**: It's the fundamental interface for Spring's IoC, providing the core functionality for dependency injection.  **Lazy Loading**: BeanFactory loads beans on-demand (lazy loading) when they are first requested, which can improve startup time for large applications.  **Lightweight**: It has a smaller memory footprint compared to ApplicationContext, making it suitable for resource-constrained environments or when you only need basic DI capabilities.  BeanFactory might be preferred in specific scenarios where you need a lightweight container or want to optimize startup time by leveraging lazy loading. | **Enhanced IoC Container**:  It extends the BeanFactory interface and adds additional enterprise-level features.  **Eager Loading**:  Beans are typically initialized eagerly at startup, making them readily available but potentially increasing startup time.  **Additional Features**: Provides support for:     \* Message resource handling     \* Event publication     \* Application-layer specific contexts (e.g., WebApplicationContext for web applications)     \* Integration with Spring AOP  ApplicationContext is the most commonly used container in Spring applications due to its richer feature set. |