**Spring Data Jpa vs spring data:**

Spring Data provides several modules such as Spring Data JPA, Spring Data MongoDB, Spring Data Redis, etc.

Spring Data JPA, on the other hand, is a specific module within Spring Data that provides integration with the Java Persistence API (JPA). It simplifies the use of JPA by providing default implementations for JPA interfaces such as JpaRepository and providing query creation from method names.

**How will you create an immutable class in java?**  
You can create immutable class in java by implementing below points:  
  
1. Make the class final so it can not be extended(inherited)  
2. Make all fields private so one can not access them from outside the class.  
3. Do not provide setter methods for the variables.  
4. Declare all mutable fields as final so that it's value can be assigned only once.

Working of spring mvc:

Spring MVC is a framework to develop a web application using Java technology by following an MVC design pattern i.e. Model View Controller. This provides a clear separation of functionality or concern, thus making it easier to develop a Java-based web application.

Working od mvc:

In summary, here is the flow of an HTTP request in Java application created using the Spring MVC framework:  
  
1) The client sends an HTTP request to a specific URL  
  
2) DispatcherServlet of Spring MVC receives the request  
  
2) It passes the request to a specific controller depending on the URL requested using @Controller and @RequestMapping annotations.  
  
3) Spring MVC Controller then returns a logical view name and model to DispatcherServlet.  
  
4) DispatcherServlet consults view resolvers until actual View is determined to render the output  
  
5) DispatcherServlet contacts the chosen view (like Thymeleaf, Freemarker, JSP) with model data and it renders the output depending on the model data  
  
6) The rendered output is returned to the client as a response

Web.xml use:

In Spring MVC, web.xml can be used to define servlets, filters, and listeners, map servlets to URL patterns, and define initialization parameters for servlets and filters. It is also used to configure security settings, error pages, and session management.

IOC :

Inversion of Control and Dependency Injection is a core design pattern of Spring framework. IOC and DI design pattern is also a popular [design pattern interview question in Java](http://javarevisited.blogspot.sg/2012/06/20-design-pattern-and-software-design.html). As the name suggest Inversion of control pattern Inverts responsibility of managing the life cycle of the object e.g. creating an object, setting their dependency etc from application to a framework, which makes writing Java application even more easy.

Spring bean lifecycle:

In Spring Framework, a bean is an object that is instantiated, assembled, and managed by the Spring IoC container. The Spring container manages the entire life cycle of the bean, from instantiation to destruction.

The bean life cycle in Spring Framework can be divided into several phases:

1. Bean Instantiation: The container creates an instance of the bean by invoking its constructor or factory method.
2. Dependency Injection: The container sets the values of the bean's properties and dependencies.
3. Bean Post-processing: The container applies any additional processing to the bean, such as adding custom behavior or changing its configuration.
4. Initialization: The bean is initialized by calling its initialization method if any.
5. Using the Bean: The bean is used by the application.
6. Destruction: The container releases any resources held by the bean, by calling its destruction method if any.

Spring Sequrity:

What is the delegating filter proxy in Spring Security?

The Delegating Filter Proxy is a design pattern used in Spring Security that allows filters to be added to the filter chain dynamically at runtime. It acts as a proxy for the actual filter, which can be configured as a bean in the Spring context.

When a request is received by the Spring Security filter chain, it passes through a chain of filters, each of which performs a specific action on the request. The Delegating Filter Proxy intercepts the request and delegates it to the actual filter, which can be a custom implementation provided by the application.

Using the Delegating Filter Proxy, filters can be added to the filter chain at runtime without modifying the web.xml file or the Spring Security configuration.