1.] What is Spring ?
ans-: Spring is a light weight container that maintain life cycle of beans.and it is a factory of all
kinds of beans.

1.1] How you implement Spring in your application ?
ans-: By adding spring dependency --
Spring- core ,spring-context, spring -orm, spring -web, spring- webmvc .

2.] How bean is configure ?
ans-: Bean is configured in .xml file in bean tag(<bean>)in which we defined class and id
and <property> tag in bean tag.

3.] How many types of Container ?
ans-: There are two types of container --
1-BeanFactory Container.
2-ApplicationContext Container.

4.] What is BeanFactory Container ?
ans-: BeanFactory is light weight container and used for enterprise (desktop) application.
Find in package (org.springframework.beans.factory.BeanFactory).

5.] What is ApplicationContext Container ?
ans-: ApplicationContext Container is the child interface of BeanFactory Container.
It is heavy Weight container .and used for Web applications.

6.] Which Container You used in your Application ?
ans-: ApplicationContext Container.

7.] Why you use ApplicationContext Container ?
ans-: Because it is used for Web applications and it provide all the facility of BeanFactory
and also some additional facility of its own.(like support i18n).

8.] How many key implementation classes of ApplicationContext Container ?
ans-: There are 3 key implemenatation classes of ApplicationContext Container--
1 - ClassPathXmlApplicationContext .
2 - FileSystemXmlApplicationContext .
3 - XmlWebApplicationContext .
4- WebApplicationContext.
5- AnnotationConfigApplicationContext.

9.] What is IOC ?
ans-: Inversion Of Control Provides DI(Dependency Injection).
Dependency Injection can be done in two ways--
1 - Constructor ,called constructor injection .
2 - Setter method , called setter injection.
The container provide injection at runtime .

9.1] Why Use DI ?Advantage of DI ?
ans-: To achieve loose coupling and IOC container injects object dependencies and gives you the ready to use object.

10.] How many Scope of Beans ?
ans-:
1 - Singleton => Single object instance will be created.
2 - Prototype => New instance created on every request.
3 - Request => Instance is created on per HTTP request.and kept in HttpRequest object.
4 - Session => Instance is created on per HTTP session.and kept in HttpSession object.
5 - Global Session => Instance is created as per Global HTTP session.(Portlet Applications)

11.] What is AutoWiring ?
ans-: It enables implicit bean dependency injection.
Autowiring can be done by two ways --
1 - XML Congiguration.
2 - Annotations . (Annotations can be applied on attributes, setter methods and constructors).

----------11.1] Why use @autowire annotation ?
ans-: If @autowire annotation is used on class then class inject all dependency .

-------12.] Autowired(required=false) what it mean?
ans-: it means mandatory .If it is true then it throw exception.

||||||13.] Define Modes of Autowiring ?
ans-: There are 4 modes of autowiring --
1 - autowire="no" => means autowire is disabled.
2 - autowire="byName" => means autowiring is resolved by property Name.(Setter Method is used).
3 - autowire="byType" => means autowiring is resolved by property Type.(Constructor is used).
{By Default autowire is byType}
4 - autowire="constructor" =>means autowiring is applied on constructor.

14.] What is autowire-candidate ="false" ?
ans-: Excluding a bean from being autowired. means bean can't be inject in another bean.

-------15.] What is primary bean ?
ans-: When we define two or more beans of the same type and these beans are auto wired by byType in
some other bean then you can define primary bean.
In <bean primary="true">tag.

------16.] How you enable Annotation based configuration ?
ans-: By adding <context:component-scan base-package="in.co.rays.project0.*" />
in dispatcher-servlet.xml .

17.] Which Architecture you used in your project ?
ans-: Layered Architecture.

18.] How many layer in this architecture ?
ans-: There are 4 layers --
1 - Presentation Layer .
2 - Buisness Layer .
3 - Data Layer .

4 - Integration Layer .

19.] Which framework used on Which layer ?Components of each layer ?
ans-:
 1 - Presentation Layer .(Tiles ,JSTL , Spring MVC)
2 - Buisness Layer .(Spring AOP)
3 - Data Layer .(Spring ORM ,Spring DAO)
4 - Integration Layer .

-----20.] What is Spring DAO ?
ans-: It provides a consistent way to use data access frameworks like jdbc,hibernate,JPA,JDO etc.
DAO's are defined with the help of @Repository annotation.
Spring DAO translates ORM specific exceptions into DataAccessException(unchecked exception class)
or into its subclass ,using @Repository annotation.

21.1] How implement Spring DAO ?OR which annotation is used on DAO ?
ans-: @ Repository .

21.] Annotated classes are scanned by which tag ?
ans-:   <context:component-scan base-package="in.co.rays.project0.*" />

22.] How you configure DCP?
ans-: By configure DataSource and SessionFactory in hibernate-servlet.xml.

```
<bean id="dataSourceC3P0"
class="com.mchange.v2.c3p0.ComboPooledDataSource"
        destroy-method="close">
        <property name="driverClass" value="com.mysql.jdbc.Driver" />
        <property name="jdbcUrl"
value="jdbc:mysql://localhost:3306/ors_project0" />
        <property name="user" value="root" />
        <property name="password" value="root" />
        <property name="acquireIncrement" value="2" />
        <property name="minPoolSize" value="1" />
        <property name="maxPoolSize" value="5" />
        <property name="maxIdleTime" value="60" />
    </bean>



    <bean id="sessionFactory"

class="org.springframework.orm.hibernate4.LocalSessionFactoryBean">
        <property name="dataSource">
            <ref bean="dataSourceC3P0" />
        </property>

        <property name="hibernateProperties">
            <props>
                <prop key="hibernate.dialect">
                    org.hibernate.dialect.MySQLDialect
                </prop>
                <prop key="hibernate.show_sql">true</prop>
                <prop key="hibernate.hbm2ddl.auto">update</prop>
            </props>
```

```
        </property>

        <property name="annotatedClasses">
            <list>

                <value>in.co.rays.project0.DTO.UserDTO</value>
                <value>in.co.rays.project0.DTO.RoleDTO</value>
                <value>in.co.rays.project0.DTO.StudentDTO</value>
                <value>in.co.rays.project0.DTO.CollegeDTO</value>
                <value>in.co.rays.project0.DTO.MarksheetDTO</value>
                <value>in.co.rays.project0.DTO.CourseDTO</value>
                <value>in.co.rays.project0.DTO.SubjectDTO</value>
                <value>in.co.rays.project0.DTO.FacultyDTO</value>
                 <value>in.co.rays.project0.DTO.TimeTableDTO</value>

           </list>
                 </property>

    </bean>
```

23.] Have you use Spring JDBC in your project ?What is Spring JDBC?
ans-: No, Spring JDBC provides an abstract way to communicate with database using plain old JDBC objects.
Spring JDBC use jdbcTemplate to communicate with DB.

-----24.] What is Row Mapper ?
ans-: It is an Interface.jdbcTemplate uses this to map a row of Resultset to an object.

-----25.] What is Spring ORM ?
ans-: It integrates ORM frameworks hibernate,JPA and JDO for data access object,transaction and
resource management implementation.
It converts ORM exceptions to DataAccessException.
It provides declarative transactions with help of Spring AOP.

----26.] What is Contextual sessions?
ans-: Hibernate itself manages one current session per transaction in contextual sessions.

27.] Define CurrentSession?Why you use CurrentSession in your application ?
ans-: CurrentSession which is bound to the lifecycle of transaction and will be automatically closed
when transaction ends.

28.] Define Service class ?
ans-: It contains buisness logic.define by @Service annotation.It does transaction handling.
Transactions are handled by Spring AOP.
Transaction handling is called declarative transaction handling.

29.] How many ways to apply transactions in service classes ?
ans-:
1- XML Configuration .
2- @Transactional annotation.

30.] Define attribute of Transaction ?

ans-:
1- propogation.Required => If transaction exist then use it otherwise create its new transaction.
2- propogation.Required New =>It create new transaction and suspend current transaction it exist.
3- propogation.Supported => If any transaction exist then use it otherwise doesn't use.
4- propogation.Not Supported => If transaction exist then doesn't use it .or if not exist then doesn't
create it.
5- propogation.Mandatory => use Current transaction .if no transaction exist then throw exception.
6- propogation.Never => If there is a transaction exist then throw exception.if not exist then doesn't use it.

------31.] What is Spring AOP ?
ans-: AOP enable modularization such as transaction management that work accross multiple types and
objects.

------32.] Key elements of AOP(Aspect Oriented Programming)?
ans-:
1- JoinPoint => JoinPoint always represents a method execution.
2- Advice => It is the action taken by an aspect at a particular JoinPoint.(@Aspect)
3- Pointcut => An expression that identifies join points.

-----33.] How You configure Spring AOP? How Enable annotation?
ans-: In hibernate-servlet.xml--

```
   <bean id="hibernateTransactionManager"

class="org.springframework.orm.hibernate4.HibernateTransactionManager">
        <property name="sessionFactory" ref="sessionFactory" />
    </bean>

    <!-- enables annotation based transaction -->

    <tx:annotation-driven transaction-
manager="hibernateTransactionManager" />
```

------34.] How you configure Email in your project ?
ans-: by configure some in dispatcher-servlet.xml --
1---

```
 <!-- Mail Sender Start -->
    <bean id="mailSender"
class="org.springframework.mail.javamail.JavaMailSenderImpl">

        <property name="host" value="smtp.gmail.com" />
        <property name="port" value="587" />
        <property name="protocol" value="smtp" />
        <property name="username" value="webmaster@sunrays.co.in" />
        <property name="password" value="PA$$1234" />

        <property name="javaMailProperties">
            <props>
                <prop key="mail.smtp.auth">true</prop>
```

```
                <prop key="mail.smtp.starttls.enable">true</prop>
                <prop key="mail.smtp.debug">false</prop>
            </props>
        </property>
    </bean>
    <!-- Mail Sender End -->
```

2---
Inject Email bean in Service class --

```
private JavaMailSenderImpl mailSender;
```

3--
Send Email by write in service class method --

```
                                        HashMap<String, String> map = new
HashMap<String, String>();
            map.put("login", dto.getLogin());
            map.put("password", dto.getPassword());

            String message =
EmailBuilder.getUserRegistrationMessage(map);

            MimeMessage msg = mailSender.createMimeMessage();

            try {
                  MimeMessageHelper helper = new MimeMessageHelper(msg);
                  helper.setTo(dto.getLogin());
                  helper.setSubject("Registration is successful for ORS
Project SUNRAYS Technologies.");
                  // use the true flag to indicate the text included is
HTML
                  helper.setText(message, true);
                  mailSender.send(msg);
```

----------xxxxxxxxxxxxxxxx----------

40.]What is Spring MVC ?
ans-: The spring web MVC framework provides model-view-controller architecture
that can be used to develop flexible and lossely
coupled web applications .

41.] What is Front Controller ?

ans-: Main Controller performs session checking and logging operations before calling any application
controller. It prevents any user to access
  application without login.

42.] What is Model ?
ans-: Model is a carrier that carry data between controller and view.Model is a map object that
contains data in form of key and value pairs .

43.] What is View ?
ans-: View is responsible for UI(User Interface).and view contains Presentation Logic.View gets data
from model using keys and displays them using JSTL
and other spring tags.

44.] What is Controller ?How you make Controller ?
ans-: Controller is a java class that contains navigation logic.By @Controller annotation .

45.] What is Service ?
ans-: Service performs buisness operations with the help of DAO and handle Transactions.

46.] What is DAO ?
ans-: DAO is implemented in ORM or JDBC and perform database CRUD operations .

47.] What is DIspatcher Servlet ?
ans-: It is a main servlet that handles all the HTTP requests and responses.

48.] How you configure Dispatcher Servlet in your application ?
ans-: By configure in Web.xml --

```
 <servlet>
        <servlet-name>dispacher</servlet-name>
        <servlet-class>
            org.springframework.web.servlet.DispatcherServlet
        </servlet-class>
        <load-on-startup>0</load-on-startup>
    </servlet>

    <servlet-mapping>
        <servlet-name>dispacher</servlet-name>
        <url-pattern>/</url-pattern>
    </servlet-mapping>
```

**NOTE =>

-------org.springframework.web.servlet.DispatcherServlet ->

(**.servlet.DispatcherServlet) is uses WebApplicationContext Container and WebApplicationContext container
in child container of ApplicationContxt container.WebApplicationContext initialize the ApplicationContext
container.one applicaton may have multiple (**.servlet.DispatcherServlet) WebApplicationContext containers
but ApplicationContext Container one for one Application.ApplicationContext is common for all WebApplicationContext containers.

49.] What is load on startup 0 ?
ans-: When servlet is initialized ,the spring will load the application-context from a file named disaptcher-servlet.xml.

50.] Can we give custom file name and location to Dispatcher -servlet.xml ?
ans-: yes, By adding listener 'ContextLoaderListener' in web.xml .

```
<context-param>
```

```
    <param-name>contextConfigLocation</param-name>
    <param-value>/WEB-INF/sunilos-servlet.xml</param-value>
    </context-param>

    <listener>
    <listener-class>
    org.springframework.web.context.ContextLoaderListener
    </listener-class>
    </listener>
```

51.] How you enable MVC based annotation ?
ans-: <mvc:annotation-driven /\>

52.] How to enable auto-discovery or scan annotation ?
ans-: <context:component-scan base-package="in.co.rays.project0.*" />

53.] How many Key model maps provided by spring ?
ans-: Two key model maps provided by spring--
1- org.springframework.ui.Model (Model is Interface) .
2- org.springframework.web.servlet.ModelAndView (ModelAndView is Class) .

|||||||54.] What is the difference between Model and ModelAndView ?
ans-: Model -> Model is received as method argument .and Model return String value(view name) .
ModelAndView -> Model object is created with view name and returned from method .

55.] Define View Resolver ?
ans-: View Resolver is an interface that render view.means it resolves view by name, name is
returned by controller .

------56.] How view resolver resolve view ?
ans-: Controller return logical name of view.view-resolver explicitly resolves view by name.
If controller does not return view then view is resolved implicitly by view resolver .
(It directly converts a logical view name into url) .

-------57.] why you put .jsp file under \WEB-INF ?
ans-: To restrict direct access of JSP from URL .

||||||||58.] Implementation classes of View Resolver ?
ans-: There are 7 classes --

1- UrlBasedViewResolver -> It can resolves any kind of views that include tiles,velocity,free marker etc .
2- InternalResourceViewResolver -> It wraps jsp/servlet or resource of same web app to forward or
include.makes bean available
as request attributes to be accessed by EL in jsp .(It should be last in the chain,
it resolve view name that may not actually exists) .
3- AbstractTemplateViewResolver .
4- VelocityViewResolver -> It resolves velocity template view .
5- FreeMarkerViewResolver -> It resolves free marker template view .
6- XmlViewResolver -> It resolves view defined in /WEB-INF/views.xml file .

7- ResouceBundleViewResolver -> It resolves views defined in views.properties file .

59.] Which View Resolver you used in your project ?
ans-: UrlBasedViewResolver .

60.] How you configure View Resolver ?
ans-: By config in dispatcher-servlet.xml-
It uses Request Dispatcher to forward a user request to a view .

```
<bean id="viewResolver"

class="org.springframework.web.servlet.view.UrlBasedViewResolver">
   //you can set different view classes to generate different views .
        <property name="viewClass">
            <value>
                org.springframework.web.servlet.view.tiles2.TilesView
            </value>
        </property>
    </bean>
```

61.] How can we forward request ?
ans-: by using prefix return "forward:/UserList" in controller . (By default forward) .

62.]How can we redirect request ?
ans-: by using prefix return "redirect:/UserList" in controller .

63.] What is @RequestMapping ?Why we use it ?
ans-: @RequestMapping is used to map controller and also its method with the URL .
ByDefault all HTTP methods are bound.
To map controller with url write @RequestMapping annotation before class.
To map controller methods with url write @RequestMapping annotation before methods.

64.] Variants of @RequestMapping for different HTTP methods ?
ans-: @GetMapping OR can be read as -
@RequestMapping(method=RequestMethod.GET) .
@PostMapping , @PutMapping , @DeleteMapping .

65.] Why use @RequestParam annotation ?
ans-: It is used to bind request parameters with method parameters or a form bean(POJO)
object in a controller .

66.] Why use @ModelAttribute annotation ?
ans-: It is used to bind request parameters to a bean object and store bean into model object.

67.] How bind request parameter ?
ans-: when parameter name and argument name are same -
public String submit(@RequestParam String userId)

when parameter name and argument name are different -
public String submit(@RequestParam("login") String userId)

68.] Why write @RequestParam(name="id",required=false) ?

ans-:It's for optional parameter.required=false means it does not throw any exception .

-----69.]What is Data Binding ?
ans-: slide 47.

70.] Why use @PathVariable annotation ?
ans-: It binds method parameters with URI template variables .

71.] Define Validations and its types ?
ans-: Validations validates the form elements entered by user.
There are 2 types of validations --
1 -Buisness Validation. => That check from database .
2- Input Validation . => That does not check from database .

There are 2 types of input validation --
1- Programmatic(Manual) validation => Validation code are written by programmer .
2- Declarative validation => Validations are declared using XML configuration or annotations.
Spring support both type of validations .

||||||72.] How to perform declarative input validation in your project ?
ans-: By Config in dispatcher-servlet.xml --(By basic and hibernate validations)

```
 <!-- JSR-303 -->
    <bean id="validator"

class="org.springframework.validation.beanvalidation.LocalValidatorFactoryBean">
        <property name="validationMessageSource" ref="messageSource" />
    </bean>
```

73.] How to apply validation on a bean ?
ans-: @Valid annotation is used to apply validation on a bean .If validation is failed, error
messages are set into BindingResult object.
call hasErrors() method of BindingResult to check if there is any validation error and take corrective path .

74.] How to display error messages on View ?
ans-: <sf:errors path="firstName"></sf:errors>

75.] How to display Buisness validation on view ?
ans-: By using JSTL core tag on View .

```
<c:if test="${not empty error }">
${error}
</c:if>
```

-------76.] Define @RequestAttribute , @CookieValue , @RequestHeader ?
ans-: @RequestAttribute => It can be used with method argument to access an object from HttpServletRequest .
@CookieValue => It can be used with method argument to access a value from cookie .

@RequestHeader => It allows a method parameter to be bound to a request header .

77.] What is Preload Data ?
ans-: Data load at the time of Html page loading .
When @ModelAttribute is applied on a method of controller then this method will be invoked before
calling any method
 mapped with @RequestMapping annotation .

78.] How you apply preload in your application ?And how you display it on view ?
ans-: By overriding preload method on controller -

```
                public void preload(Model model) {

            List<RoleDTO> roleList = roleService.search(null);
            model.addAttribute("roleList", roleList);

      }
```

To display preload data on view -

```
<sf:select path="roleId" cssClass="form-control" >
   <sf:option value="0">------------Select------------</sf:option>
   <sf:options items="${roleList}" itemValue="id" itemLabel="roleName">
</sf:options>  </sf:select>
```

79.] How can we inject Container objects ?
ans-: Container objects HttpServletRequest ,HttpServletResponse and HttpSession can be injected to a
controller's method using method arguments.

80.] What is Internationalization(i18n) and Localization (L10n)?
ans-: i18n & L10n are means of supporting multi-languages for multiple countries or multiple regions within
a country without changing code in the app.
Messages files are created for each language and kept in root class path.(Files are created by 2 character
language code say "en" for english) .
labels,Messages and error messages are stored in message files as key and value pairs.
keys are same all across the files.values are labels and messages .

-------81.] How you configure i18n & L10n ?
ans-: By config 3 classes in dispatcher-servlet.xml -
1- ReloadableResourceBundleMessageSource .
2- CookieLocaleResolver .
3- LocaleChangeInterceptor .(It intercepts each request and look for request parameter "lang"
 (?lang=hi or ?lang=en)to change the locale).
(selected locale is saved in cookies for future subsequent requests) .

```
  <!-- Defines the message resources -->
    <bean id="messageSource"

class="org.springframework.context.support.ReloadableResourceBundleMessag
eSource">
```

```
        <property name="basename" value="classpath:messages" />
        <property name="defaultEncoding" value="UTF-8" />
    </bean>

    <bean id="localeResolver"

class="org.springframework.web.servlet.i18n.CookieLocaleResolver">
        <property name="defaultLocale" value="en"></property>
        <property name="cookieName" value="myAppLocaleCookie"></property>
           <property name="cookieMaxAge" value="3600"></property>
    </bean>

    <!-- har bar check krta h before request and before response   -->

    <mvc:interceptors>
        <bean id="localeChangeInterceptor"

class="org.springframework.web.servlet.i18n.LocaleChangeInterceptor">
            <property name="paramName" value="lang" />
        </bean>
    </mvc:interceptors>
```

-------82.] How to view Locale on jsp ?
ans-: < s:message code="label.login"></s:message>

------83.] How to get Locale messages on controller ?
ans-: By using MessageSource object .

```
@Autowired
      private MessageSource messageSource;
      String msg = messageSource.getMessage("message.success", null,
locale);
```

BeanFactory is the root interface of Spring IoC container.

85.] Features of Spring framework ?
ans-: Spring is a light-weight framework for the development of enterprise-ready applications.
 It provides following features:

=>Dependency Injection
=>Declarative transaction management using Spring AOP.
=>Spring MVC web application and RESTful web service framework
=>Foundational support for JDBC, JPA, JMS
=>Mailing facilities.

----------Any Spring bean declared in the DispatcherServlet’s application context that implements
HandlerExceptionResolver
 will be used to intercept and process any exception raised in the MVC system and not handled by
a Controller