**7) What are the commands to run and stop Spring Boot executable jar file?**

You need to open cmd or shell window command and use

java -jar

Example

$ java -jar myproject-0.0.1-SNAPSHOT.jar

To stop use ctrl+C

**8) How can you change JDK version in Spring Boot?**

To change the JDK version in Spring Boot, you can overwrite it by adding a java. version property tag as given:

1.8

**9) What is the process that you need to follow to run Spring Boot application on the custom port?**

In order to run a Spring Boot application, you require to put server.port properties in application.properties. For example, server.port=8050

**10) What is Spring Boot starter? How is it useful?**

Spring Boot has many starters. They are a set of convenient dependency descriptors. Starter allows you to include these descriptors in your pom.xml.

For example, If you want to work with Spring MVC, you can include “spring–boot–starter–web” as a dependency in pom.xml.

**11) Can you use Spring Boot with applications which are not using Spring?**

No, it is not possible as Spring Boot is limited to Spring application only.

**12) What is the name of the configuration file which you can use in Spring Boot?**

The configuration file used in Spring Boot projects is called application.properties. It is an important file which allows you to override your default configurations.

**13) What is DevTools in Spring Boot?**

Spring Boot DevTools helps you to increase the productivity of the developer. So, you don't require to redeploy your application every time you make the changes. It allows the developer to reload changes without the need of restarting of the server.

**14) What are the important features of Reactjs?**

Important features of Reactjs are:

* Web Development
* Spring Application
* Application occasions and listeners
* Admin highlights
* YAML Support
* Type-safe Configuration
* Externalized Configuration
* Properties Files
* Logging and Security

**15) What are the essential components of Spring Boot**

The important components of Spring Boot are:

* Spring Boot Starter
* Spring Boot autoconfiguration
* Spring Boot Actuator
* Spring Boot CLI

**16) How are properties defined? Where?**

You can define properties in the application.properties file exists in the classpath.

Example: configure default DataSource bean

database.host=localhost

**17) What is spring-boot-starter-parent?**

It is a special starter which makes Gradle or Maven dependency-management easy by adding jars to your classpath.

**18) How to enable HTTP/2 supports in Spring Boot?**

User can enable HTTP/2 support by using

server.http2.enabled configuration property.

**19) What is a Spring Boot Actuator?**

Spring Boot Actuator allows you to monitor and manage your application when you want to push it for the production. It helps you to control your application by using HTTP endpoints.

**20) What is the command to run Spring Boot application to custom port?**

In application.properties, add following property.

server.port = 8181

**21) How can you access a value defined in the application? What is properties file in Spring Boot?**

Use the @Value annotation to access the properties which is defined in the application - properties file.

@Value("${custom.value}")

private String customVal;

**22) What is the primary difference between Spring and Spring Boot?**

Spring is a web application development framework based on Java. On the other hand Spring Boot is an extension of the spring framework which eliminated the boilerplate configuration required for setup a Spring application.

**23) Explain Spring Boot Admin**

Spring Boot admin is a community project which helps you to manage and monitor your Spring Boot applications.

**24) How can you connect Spring Boot to the database using JPA?**

Spring Boot supports spring-boot-data-JPA start, which helps you to connect spring application with a relational database.

**25) Explain @RestController annotation in Spring Boot?**

The @RestController annotation helps you to add @ResponseBody and @Controller annotations to the class.

You can also import org.springframework.web.bind.annotation package in your file.

**26) Define the term Spring Initializer**

Spring initializer is a web application which can create an initial project structure for you.

**27) Explain Spring CLI**

Spring CLI is used for writing in Groovy Spring Boot application, which helps you to concise code.

**28) Where can you define properties in Spring Boot application?**

You can define properties of Spring Boot into a file called application.properties. It helps you to create this file manually, or you can use Spring Initializer to create this file.

**29) What are embedded containers support by Spring**

Spring Boot support the main three embedded containers:

1) Tomcat

2) Jetty

3) Undertow.

By default, it uses Tomcat as an embedded container.

**30) Explain thymeleaf in Spring Boot**

Thymelaf is a server-side Java template engine for a web application. It helps you to bring elegant natural templates to your web application.

**31) What are the Spring Boot properties?**

Spring Boot offers various properties which can be specified inside our project's application.properties file. It helps you to set values like a server-port number, database connection configuration, etc.

**32)  What is the main difference between JPA and Hibernate?**

The main difference between both of them is that JPA is a specification/Interface, whereas Hibernate is only JPA implementations.

**33) What is a shutdown in the actuator?**

A shutdown is an endpoint that helps application to be shut down properly. This feature is not enabled by default.

However, you can use it by setting command: management.endpoint.shutdown.enabled=true in your application.properties file.

**34) Is it possible to replace or override the Embedded Tomcat server in Spring Boot?**

Yes, it is possible to replace the Embedded Tomcat with any other servers by using the starter dependencies. For that, you can use spring-boot-starter-jetty or as a dependency for according you to your need.

**35) Can you disable the default web server in the Spring Boot application?**

Yes, we can disable the default web server by using application.properties to configure the web application type.

**36) How do you Add, Filter to an application?** There are three methods to add filter to Spring Boot application:

* By implementing Filter interface.
* Using FilterRegistrationBean.
* Using MVC controller.

**37) What are Spring Boot Starter Projects?**

Starters in Spring Boot are a set of convenient descriptors that are included in Spring Boot applications. It comes with a variety of Spring-related technology which makes the entire process of the application development much easier.

**38) What is @pathVariable?**

@PathVariable annotation helps you to extract information from the URI directly.

**39) What is Swagger2?**

Swagger is used to describing the structure of APIs. Swagger 2 is an open-source service provided in Spring Boot which makes it easier for the machines to find out the structure of APIs like RESTful Web services.

**40) What are different environments for enterprise application development?**

* Dev
* QA
* Stage
* Production

**41) What are the major differences between RequestMapping and GetMapping?**

RequestMapping can be used with GET, POST, PUT, and many other request methods using the method attribute on the annotation. Whereas GetMapping is only an extension of RequestMapping, which helps you to improve clarity on requests.

**42) How can you define properties in Spring Boot?**

You can define properties in Spring Boot with the help of the application.properties file which exists in a classpath of the application as follows.

**43) How to create a Spring Boot project using Maven?**

Use any of the following methods to create a project.

* Spring Initializr
* Spring Boot CLI
* Spring Starter Project Wizard

**44) What is the use of profiles in Spring Boot?**

Profiles are used to separate various parts of your spring application configuration and make it only available in certain environments.

**45) How to change tomcat HTTP port?**

To change the tomcat HTTP port, you have to change default HTTP property in application.properties file.

**46) What is LiveReload in Spring Boot?**

LiveReload is a spring-boot-devtools module that includes LiveReload server to trigger a browser refresh when a resource is changed. LiveReload server extensions are available freeware for Firefox, Chrome, and Safari.

**47) What are the major benefits of spring Externalized Configuration?**

Externalized Configuration helps to work with the same code in different environments. Developers can use YAML files, properties files, command-line arguments, and environment variables to externalize configuration.

**48) What do you mean by hot-swapping in Spring Boot?**

It is a way to reload the changes without restarting Tomcat, or Jetty server. Eclipse and Many other IDEs support bytecode hot swapping. If you make any changes that don’t affect the method signature, it should reload without side effect.

**49) Explain Auto-Configuration in Spring Boot.**

Auto-configuration is used to configure Spring application automatically based on dependencies of classpath parameter. It makes development faster and easier.

**50) What is the meaning of Aspect-Oriented Programming (AOP)?**

Aspect-Oriented Programming supplements Object-Oriented Programming that aims to increase modularity. AOP breaks program logic into various parts, which are called concerns.

**51) How to enable logging in Spring Boot?**

In order to enable debug logging, you can specify --debug while starting the application from the command prompt.

**52) Explain overriding default properties in Spring Boot application.**

Spring Boot has lots of properties that can be easily overridden by specifying them in application.properties.

**53) Explain Docker in Spring Boot.**

It is a tool designed to create, deploy, and run a project by using containers.

**54) Define ELK stack.**

The ELK Stack is made of three open-source products: 1) Elasticsearch, 2) Logstash, and 3) Kibana.

* Elasticsearch: It is a NoSQL database which is based on the open-source search engine called Lucene.
* Logstash: It is a data processing pipeline tool which accepts inputs from sources, performs different transformations, and exports the data to targets.
* Kibana: Kibana helps users to visualize data with graphs and chart in Elasticsearch.

**55)  How to handle exception in Spring Boot.**

Spring Boot provides a very useful way to handle exceptions using @ControllerAdvice annotation.

**56) Explain caching.**

Caching is a memory are that temporary stores frequently accessed data that is otherwise expensive to get or compute.

**57) What is Cross-Site Request Forgery attack?**

Cross-Site Request Forgery attack or one-click attack is an attack that forces other users to execute malicious commands on the application. CSRF attack specifically targets state-changing requests.

**58) Define apache freemarker.**

Freemarker is a Java-based template used to generate plain text, emails, HTML file, etc.

**59) What is mean by spring batch?**

Spring Boot Batch provides code reusability which is important when working with large numbers of records, including transaction management, logging, skipping, job processing statistics, and job restarts.

**60) Explain Apache Kafka.**

Apache Kafka is an open-source messaging platform. LinkedIn develops it. Apache Kafka enables the user to build distributed applications and handle real-time data feeds. Kafka is suitable for both offline and online messaging.

**61) Explain CORS in Spring Boot?**

CORS stands for Cross-Origin Resource Sharing is a  mechanism implemented by browsers and helps users to authorize cross-domain requests. This mechanism serves as an alternative to less secure and less powerful hacks of the kinds of IFrame or JSONP.

**62) Explain different types of dependency injection.**

There are two types of dependency injection in Spring Boot. They are as follows:

* Constructor based dependency injection: It is a technique in which one class object supplies the dependency of another object.
* Setter-based dependency injection: It is a dependency injection in which the framework injects the primitive and string-based values using setter method.

**63) What are the advantages of micro service?**

Following are the major advantages of micro service:

* It makes development fast and easy.
* Compatible with all container.
* Reduce production time.
* It's a lightweight model that supports a major business application.

**64) What is the default package in Spring Boot?**

A class without any package declaration is considered as a default package.

**65) Explain the difference between an embedded container and a WAR.**

The main difference between these two is:

Embedded containers help you to run Spring Boot application as a JAR from the command prompt without setting up any web server, while to run a WAR you need first to set up Tomcat.

**66) Explain Spring MVC.**

It is a traditional web application framework which helps you to build a web application. This framework is similar to the framework of Struts.

**67) What is the use of <set> tag?**

This tag is used to write to inject java set using XML.

**68)  What do you mean by aspect?**

It is a set of APIs which provides cross-cutting requirements.

**69) What is join point in Spring Boot?**

It is a program execution point like the handling of an exception or the execution of a method. In AOP, a join point is referred to as a method execution.

**70) How can you set active profile in Spring Boot?**

Follow the following methods to set an active profile in Spring Boot.

* Pass this profile as an argument when you launch the Spring Boot application.
* Set active the active profile in application.properties file.

**71) Is excluding package without using the basePackages filter is possible? How?**

Yes. It is possible to exclude package without using the basePackages filter by simply using the exclude attribute while using the @SpringBootApplication annotation.

**72) List out benefits of using the JavaConfig method.**

Following are the benefits of JavaConfig method.

* User can take benefit of object-oriented configuration.
* Spring Boot configuration improves the efficiency of web-based application by eliminating complex XML configuration.

**73) Explain steps to deploy an application on virtual machine.**

Below are the steps to deploy application on virtual machine.

* Install Java.
* Install the Application Server.
* Deploy the application war file.

**74) List out some of the Spring Boot Starters.**

Different Spring Boot Starters are as follows:

* Security
* Parent
* web
* Thymeleaf
* Freemarker

**1. What is spring boot?**

Spring Boot is basically, a microservice framework that is built on top of the spring framework.

Spring boot helps developers to focus more on convention rather than configuration as all the basic configuration will be handled by spring boot.

The main aim of Spring boot is to give you a production-ready application. So, the moment you create a spring-boot project, it is runnable and can be executed/deployed on the server.

It comes with features like autoconfiguration, auto dependency resolution, embedded servers, security, health checks which enhances the productivity of a developer.

**2. How to create spring-boot project in eclipse?**

One of the ways to create a spring boot project in eclipse is by using Spring Initializer.

You can go to the official website of spring and add details such as version, select maven or Gradle project, add your groupId, artifactId, select your required dependencies and then click on CREATE PROJECT.

Once the project is created, you can download it and extract and import it in your eclipse or STS.

And see your project is ready!

**3. How to deploy spring boot application in tomcat?**

Whenever you will create your spring boot application and run it, Spring boot will automatically detect the embedded tomcat server and deploy your application on tomcat.

After successful execution of your application, you will be able to launch your rest endpoints and get a response.

**4. What is the difference between spring and spring boot?**

Difference between Spring and Spring boot are as follows:

Spring –

Is a dependency injection framework.

It is basically used to manage the life cycle of java classes (beans). It consists of a lot of boilerplate configuration.

Uses XML based configuration.

It takes time to have a spring application up and running and it’s mainly because of boilerplate code.

Spring boot-

It is a suite of pre- configured frameworks and technologies which helps to remove boilerplate configuration.

Uses annotations.

It is used to create a production-ready code.

**5. What is actuator in spring boot?**

Actuator is one of the best parts of spring boot which consists of production-ready features to help you monitor and manage your application.

With the help of an actuator, you can monitor what is happening inside the running application.

Actuator dependency figures out the metrics and makes them available as a new endpoint in your application and retrieves all required information from the web. You can identify beans, the health status of your application, CPU usage, and many more with the actuator.

**6. How to change port in spring boot?**

The default port number to start your SpringBoot application is 8080.

Just to change the port number, you need to add server.port=8084c(your port number) property in your application.properties file and start your application.

**7. How to install spring boot in eclipse?**

The classic and preferred way to install STS in eclipse is:

Go to Eclipse IDE, click on “Help”->then go to Eclipse marketplace->and type Spring IDE and click on finish button.

**8. How to create war file in spring boot?**

To create a war file in spring boot you need to define your packaging file as war in your pom.xml(if it is maven project).

Then just do maven clean and install so that your application will start building. Once the build is successful, just go into your Target folder and you can see .war file generated for your application.

**9. What is JPA in spring boot?**

JPA is basically Java Persistence API. It’s a specification that lets you do ORM when you are connecting to a relational database which is Object-Relational Mapping.

So, when you need to connect from your java application to relational database, you need to be able to use something like JDBC and run SQL queries and then you get the results and convert them into Object instances.

ORM lets you map your entity classes in your SQL tables so that when you connect to the database , you don’t need to do query yourself, it’s the framework that handles it for you.

And JPA is a way to use ORM, it’s an API which lets you configure your entity classes and give it to a framework so that the framework does the rest.

**10. How to save image in database using spring boot?**

First configure mysql in your spring boot application.

Then you can map your entities with your db tables using JPA.

And with the help of save() method in JPA you can directly insert your data into your database

@RestController

@RequestMapping("/greatleasrning")

public class Controller {

@Autowired

private final GreatLearningRepository greatLearningRepository;

public Controller(GreatLearningRepository greatLearningRepository) {

}

In above case, your data which may be in JSON format can be inserted successfully into database.

@RequestMapping(method = RequestMethod.POST)

ResponseEntity<?> insert(@RequestBody Course course) {

greatLearningRepository.save(course);

return ResponseEntity.accepted().build();

}

}

**11. What is auto configuration in spring boot?**

AutoConfiguration is a process by which Spring Boot automatically configures all the infrastructural beans. It declares the built-in beans/objects of the spring specific module such as JPA, spring security and so on based on the dependencies present in your applications class path.

For example: If we make use of Spring JDBC, the spring boot autoconfiguration feature automatically registers the DataSource and JDBCTemplete bean.

This entire process of automatically declaring the framework specific bean without the need of writing the xml code or java config code explicity is called Autoconfiguration which is done by springBoot with the help of an annotation called @EnableAutoconfiguration alternatively @SpringBootApplication.

**12. How to change port number in spring boot?**

The default port number to start your Spring Boot application is 8080.

Just to change the port number, you need to add server.port=8084(your port number) property in your application.properties file and start your application.

**13. How to resolve whitelabel error page in spring boot application?**

This is quite common error in spring boot application which says 404(page not found).

We can mostly resolve this in 3 ways:

Custom Error Controller– where you will be implementing ErrorController interface which is provided by SpringFramework and then overriding its getErrorPath() so that you can return a custom path whenever such type of error is occurred.

By Displaying Custom error page– All you have to do is create an error.html page and place it into the src/main/resources/templates path. The BasicErrorController of of springboot will automatically pick this file by default.

By disabling the whitelabel error page– this is the easiest way where all you need to do is server.error.whitelabel.enabled property to false in the application.properties file to disable the whitelabel error page.

**14. How to fetch data from database in spring boot?**

You can use the following steps to connect your application with MySQL database.

1. First create a database in MySQL with create DATABASE student;

2. Now, create a table inside this DB:

CREATE TABLE student(studentid INT PRIMARY KEY NOT NULL AUTO\_INCREMENT, studentname VARCHAR(255));

3. Create a SpringBoot application and add JDBC, MySQL and web dependencies.

4. In application.properties, you need to configure the database.

spring.datasource.url=jdbc:mysql://localhost:3306/studentDetails

spring.datasource.username=system123

spring.datasource.password=system123

spring.jpa.hibernate.ddl-auto=create-drop

5. In your controller class, you need to handle the requests.

package com.student;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.jdbc.core.JdbcTemplate;

import org.springframework.web.bind.annotation.RestController;

@RestController

public class JdbcController {

@Autowired

JdbcTemplate jdbc;

@RequestMapping("/save")

public String index(){

jdbc.execute("insert into student (name)values(GreatLearnings)");

return "Data Entry Successful";

}

}

6. Run the application and check the entry in your Database.

**15. What is response entity in spring boot?**

Response Entity is basically an HTTP response which includes headers,status code and body of your response.

**16. How to use logger in spring boot?**

There are many logging options available in SpringBoot. Some of them are mentioned below:

Using log4j2:

import org.apache.logging.log4j.Logger;

import org.apache.logging.log4j.LogManager;

// [...]

Logger logger = LogManager.getLogger(LoggingController.class);

Using Lombok:

All you need to do is add a dependency called org.projectlombok in your pom.xml as shown below:

<dependency>

<groupId>org.projectlombok</groupId>

<artifactId>lombok</artifactId>

<version>1.18.4</version>

<scope>provided</scope>

</dependency>

Now you can create a loggingController and add the @Slf4j annotation to it. Here we would not create any logger instances.

@RestController

@Slf4j

public class LoggingController {

@RequestMapping("/logging")

public String index() {

log.trace("A TRACE Message");

log.debug("A DEBUG Message");

log.info("An INFO Message");

log.warn("A WARN Message");

log.error("An ERROR Message");

return "Here are your logs!”;

}

}

So, there are many such ways in spring boot to use logger.

**17. What is bootstrapping in spring boot?**

One of the way to bootstrap your spring boot application is using Spring Initializer.

you can go to the official website of spring and select your version, and add you groupID, artifactId and all the required dependencies.

And then you can create your restEndpoints and build and run your project.

There you go, you have bootstrapped your spring boot application.

**18. Spring boot introduced in which year?**

Spring boot was introduced in the year 2002.

**19. How to create jar file in spring boot?**

To create a jar file in spring boot you need to define your packaging file as jar in your pom.xml(if it is maven project).

Then just do maven build with specifying goals as package so that your application will start building.

Once the build is successful, just go into your Target folder and you can see .jar file generated for you application.

**20. How to handle exceptions in spring boot?**

To handle exceptions in spring boot, you can use @ControllerAdvice annotation to handle your exceptions globally.

In order to handle specific exception and send customized response, you need to use @ExceptionHandler annotation.

**21. What is dependency injection in spring boot?**

Dependency injection is a way through which the Spring container injects one object into another. This helps for loose coupling of components.

For example: if class student uses functionality of department class, then we say student class has dependency of Department class. Now we need to create object of class Department in your student class so that it can directly use functionalities of department class is called dependency injection.

**22. How to store image in MongoDB using spring boot?**

One of the way for storing image in MongoDB is by using Spring Content. And also you should have the below dependency in your pom.xml.

<dependency>

<groupId>com.github.paulcwarren</groupId>

<artifactId>spring-content-mongo-boot-starter</artifactId>

<version>0.0.10</version>

</dependency>

You should have a GridFsTemplate bean in your applicationContext.

@Configuration

public class Config

@Bean

public GridFsTemplate gridFsTemplate() throws Exception {

return new GridFsTemplate(mongoDbFactory(), mappingMongoConverter());

}

...

Now add attributes so that your content will be associated to your entity.

@ContentId

private String contentId;

@ContentLength

private long contentLength = 0L;

@MimeType

private String mimeType = "text/plain";

And last but not the least, add a store interface.

@StoreRestResource(path="greatlearningImages")

public interface GreatLearningImageStore extends ContentStore<Candidate, String> {

}

That’s all you have to do to store your images in mongoDb using Springboot.

**23. Which is the ui web framework that is built to use spring boot?**

The best UI web framework that can be used with springboot is JHipster.

With this you can generate your web-applications and microservices within less time.

**24. How to configure hibernate in spring boot?**

The important and required dependency to configure hibernate is:

spring-boot-starter-data-jpa

h2 (you can also use any other database)

Now, provide all the database connection properties in application.properties file of your application in order to connect your JPA code with the database.

Here we will configure H2 database in application.properties file:

spring.datasource.url=jdbc:h2:file:~/test

spring.datasource.driverClassName=org.h2.Driver

spring.datasource.username=test

spring.datasource.password=test

spring.jpa.database-platform=org.hibernate.dialect.H2Dialect

spring.h2.console.enabled=true

spring.h2.console.path=/h2-console

Adding the above properties in your application.properties file will help you to interact with your database using JPA repository interface.

**25. Why spring boot is used for microservices?**

In microservices, you can write code for your single functionality. You can use different technology stacks for different microservices as per the skill set.

You can develop this type of microservices with the help of Spring boot very quickly as spring boot gives priority to convention over configuration which increases the productivity of your developers.

**26. Mention the advantages of Spring Boot.**

Advantages of Spring Boot-

It allows convention over configuration hence you can fully avoid XML configuration.

SpringBoot reduces lots of development time and helps to increase productivity.

Helps to reduce a lot of boilerplate code in your application.

It comes with embedded HTTP servers like tomcat, Jetty, etc to develop and test your applications.

It also provides CLI(Command Line Interface) tool which helps you to develop and test your application from CMD.

**27. Explain Spring Actuator and its advantages.**

An actuator is one of the best parts of spring boot which consists of production-ready features to help you monitor and manage your application.

With the help of Actuator, you can monitor what is happening inside the running application.

Actuator dependency figures out the metrics and makes them available as a new endpoint in your application and retrieves all required information from the web. You can identify beans, the health status of your application, CPU usage, and many more with the actuator.

**28. Explain what is thyme leaf and how to use thymeleaf?**

Thymeleaf is a server-side java template engine which helps processing and creating HTML,XML, JavaScript , CSS, and text. Whenever the dependency in pom.xml(in case of maven project) is find, springboot automatically configures Thymeleaf to serve dynamic web content.

Dependency: spring-boot-starter-thymeleaf

We can place the thyme leaf templates which are just the HTML files in src/main/resources/templates/ folder so that spring boot can pick those files and renders whenever required.

Thymeleaf will parse the index.html and will replace the dynamic values with its actual value that is been passed from the controller class.

That’s it, once you run your Spring Boot application and your message will be rendered in web browsers.

**29. What is the need for Spring Boot DevTools?**

This is one of the amazing features provided by Spring Boot, where it restarts the spring boot application whenever any changes are being made in the code.

Here, you don’t need to right-click on the project and run your application again and again. Spring Boot dev tools does this for you with every code change.

Dependency to be added is: spring-boot-devtools

The main focus of this module is to improve the development time while working on Spring Boot applications.

**30. Can we change the port of the embedded Tomcat server in Spring boot?**

Yes, you can change the port of embedded Tomcat server in Spring boot by adding the following property in your application.properties file.

server.port=8084

**31. Mention the steps to connect Spring Boot application to a database using JDBC**

Below are the steps to connect your Spring Boot application to a database using JDBC:

Before that, you need to add required dependencies that are provided by spring-boot to connect your application with JDBC.

Step 1: First create a database in MySQL with create DATABASE student;

Step 2: Now, create a table inside this DB:

CREATE TABLE student(studentid INT PRIMARY KEY NOT NULL AUTO\_INCREMENT,

studentname VARCHAR(255));

Step 3: Create a springBoot and add JDBC,mysql and web dependencies.

Step 4: In application.properties, you need to configure the database.

spring.datasource.url=jdbc:mysql://localhost:3306/studentDetails

spring.datasource.username=system123

spring.datasource.password=system123

spring.jpa.hibernate.ddl-auto=create-drop

Step 5: In your controller class, you need to handle the requests.

package com.student;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.jdbc.core.JdbcTemplate;

import org.springframework.web.bind.annotation.RestController;

@RestController

public class JdbcController {

@Autowired

JdbcTemplate jdbc;

@RequestMapping("/save")

public String index(){

jdbc.execute("insert into student

(name)values(GreatLearnings)");

return "Data Entry Successful";

}

}

Step 6: Run the application and check the entry in your Database.

Step 7: You can also go ahead and open the URL and you will see “Data Entry Successful” as your output.

**32. What are the @RequestMapping and @RestController annotation in Spring Boot used for?**

The @RequestMapping annotation can be used at class-level or method level in your controller class.

The global request path that needs to be mapped on a controller class can be done by using @RequestMapping at class-level. If you need to map a particular request specifically to some method level.

Below is a simple example to refer to:

@RestController

@RequestMapping("/greatLearning")

public class GreatLearningController {

@RequestMapping("/")

String greatLearning(){

return "Hello from greatLearning ";

}

@RequestMapping("/welcome")

String welcome(){

return "Welcome from GreatLearning";

}

}

The @RestController annotation is used at the class level.

You can use @RestController when you need to use that class as a request handler class.All the requests can be mapped and handled in this class.

@RestController itself consists @Controller and @ResponseBody which helps us to remove the need of annotating every method with @ResponseBody annotation.

Below is a simple example to refer to for use of @RestController annotation:

@RestController

@RequestMapping(“bank-details”)

public class DemoRestController{

@GetMapping(“/{id}”,produces =”application/json”)

public Bank getBankDetails(@PathVariable int id){

return findBankDetailsById();

}

}

Here, @ResponseBody is not required as the class is annotated with @RestController.

**33. How can we create a custom endpoint in Spring Boot Actuator?**

By using @Endpoint annotation, you can create a custom endpoint.

**34) What do you understand by auto-configuration in Spring Boot and how to disable the auto-configuration?**

AutoConfiguration is a process by which Spring Boot automatically configures all the infrastructural beans. It declares the built-in beans/objects of the spring-specific module such as JPA, spring-security, and so on based on the dependencies present in your application’s classpath.

For example: If we make use of Spring JDBC, the spring boot autoconfiguration feature automatically registers the DataSource and JDBCTemplete bean.

This entire process of automatically declaring the framework-specific bean without the need of writing the XML code or java-config code explicitly is called Autoconfiguration which is done by spring-boot with the help of an annotation called @EnableAutoconfiguration alternatively @SpringBootApplication.

1. You can exclude the attribute of @EnableAutoConfiguration where you don’t want it to be configured implicity in order to disable the spring boot’s auto-configuration feature.

2. Another way of disabling auto-configuration is by using the property file:

For example:

spring.autoconfigure.exclude=

org.springframework.boot.autoconfigure.mongo.MongoAutoConfiguration,

org.springframework.boot.autoconfigure.data.MongoDataConfiguration,

In the above example, we have disabled the autoconfiguration of MongoDB.

**35. Can you give an example for ReadOnly as true in Transaction management?**

Yes, example for ReadOnly as true in Transaction Management is:

Suppose you have a scenario where you have to read data from your database like if you have a STUDENT database and you have to read the student details such as studentID, and studentName.

So in such scenarios, you will have to set read-only on the transaction.

**36. Mention the advantages of the YAML file than Properties file and the different ways to load**

YAML file in Spring boot.

YAML gives you more clarity and is very friendly to humans. It also supports maps, lists, and other scalar types.

YAML comes with hierarchical nature which helps in avoiding repetition as well as indentations.

If we have different deployment profiles such as development, testing, or production and we may have different configurations for each environment, so instead of creating new files for each environment we can place them in a single YAML file.

But in the case of the properties file, you cannot do that.

For example:

spring:

profiles:

active:

-test

---

spring:

profiles:

active:

-prod

---

spring:

profiles:

active:

-development

**37. What do you understand by Spring Data REST?**

By using Spring Data Rest, you have access to all the RESTful resources that revolves around Spring Data repositories.

Refer the below example:

@RepositoryRestResource(collectionResourceRel = "greatlearning", path = "sample")

public interface GreatLearningRepo extends CustomerRepository< greatlearning, Long> {

}

Now you can use the POST method in the below manner:

{

“Name”:”GreatLearning”

}

And you will get response as follow:

{

“Name”:”GreatLearning”

}

\_\_\_\_\_\_\_\_\_\_

{

"name": "Hello greatlearning "

"\_links": {

"self": {

"href": "<a href="http://localhost:8080/sample/1">http://localhost:8080/ greatlearning /1</a>"

},

" greatlearning ": {

“href": "<a href="http://localhost:8080/sample/1">http://localhost:8080/ greatlearning /1</a>"

}

}

In the above, you can see the response of the newly created resource.

**38. How do you Configure Log4j for logging?**

Spring Boot supports log4j2 for logging configuration. what all you have to do is you have to exclude log backfile and include log4j2 for logging.

You can do it by using the spring starter projects.

**39. What do you think is the need for Profiles?**

The application has different stages-such as the development stage, testing stage, production stage and may have different configurations based on the environments.

With the help of spring boot, you can place profile-specific properties in different files such as

application-{profile}.properties

In the above, you can replace the profile with whatever environment you need, for example, if it is a development profile, then application-development.properties file will have development specific configurations in it.

So, in order to have profile-specific configurations/properties, you need to specify an active profile.

**40. What are the steps to add a custom JS code with Spring Boot?**

All you have to do is just create a static folder under src/main/resources and place all your static content in this folder.

And your custom JS code will be added with spring boot.

**41. What is the name of the default H2 database configured by Spring Boot?**

H2 database is an in-memory database configured by SpringBoot.

The default name of the H2 database that is configured by spring boot is testdb.

**42. How to call servlet in spring boot?**

Servlet can be configured by using implementing or extending WebAppInitializer interface, AbstractDispatcherServletInitializer abstract class, and AbstractAnnotationConfigDispatcherServletInitializer abstract class

**43. what is the HTTP methods that can be implemented in spring boot rest service?**

Spring boot rest service helps in CRUD (create,retrieve,update,delete)operations. So based on that most commonly used HTTP methods are GET, POST, PUT, DELETE, and PATCH are the methods that can be implemented in spring boot rest services.

**44. How to debug spring boot application in eclipse?**

Debugging your spring boot application is as simple as same as debugging your Java application.

All you have to do is place debug point within your application, and right-click-> click on debug as java application or spring boot application.

**45. How to insert data in mysql using spring boot?**

First configure mysql in your spring boot application.

Then you can map your entities with your db tables using JPA.

And with the help of save() method in JPA, you can directly insert your data into your database.

@RestController

@RequestMapping("/greatleasrning")

public class Controller {

@Autowired

private final GreatLearningRepository greatLearningRepository;

public Controller(GreatLearningRepository greatLearningRepository) {

this. greatLearningRepository = greatLearningRepository;

}

In the above case, your data which may be in JSON format can be inserted successfully into the database.

@RequestMapping(method = RequestMethod.POST)

ResponseEntity<?> insert(@RequestBody Course course) {

greatLearningRepository.save(course);

return ResponseEntity.accepted().build();

}

}

**46. How to create a login page in spring boot?**

You can create a simple and default login page in spring boot, you can make use of Spring security. Spring security secures all HTTP endpoints where the user has to login into the default HTTP form provided by spring.

We need to add spring-boot-starter-security dependency in your pom.xml or build.gradle and a default username and password can be generated with which you can log in.

**47. What is the main class in spring boot?**

Usually in java applications, a class that has a main method in it is considered as a main class. Similarly, in spring boot applications main class is the class which has a public static void main() method and which starts up the SpringApplicationContext.

**48. How to use crud repository in spring boot?**

In order to use crud repository in spring boot, all you have to do is extend the crud repository which in turn extends the Repository interface as a result you will not need to implement your own methods.

Create a simple spring boot application which includes below dependency:

spring-boot-starter-data-jpa, spring-boot-starter-data-rest

And extend your repository interface as shown below:

package com.greatlearning;

import java.util.List;

import org.springframework.data.repository.CrudRepository;

import org.springframework.data.rest.core.annotation.RepositoryRestResource;

@RepositoryRestResource

public interface GreatLearning extends CrudRepository<Candidate, Long>

{

public List<Candidate> findById(long id);

//@Query("select s from Candidate s where s.age <= ?")

public List<Candidate> findByAgeLessThanEqual (long age);

}

**49. How to run spring-boot jar from the command line?**

In order to run spring boot jar from the command line, you need to update you pom.xml(or build.gradle) of your project with the maven plugin.

<build>

<plugins>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

</plugin>

</plugins>

</build>

Now, Build your application and package it into the single executable jar. Once the jar is built you can run it through the command prompt using the below query:

java -jar target/myDemoService-0.0.1-SNAPSHOT.jar

And you have your application running.

**50. What is Spring Boot CLI and how to execute the Spring Boot project using boot CLI?**

Spring Boot CLI is nothing but a command-line tool which is provided by Spring so that you can develop your applications quicker and faster.

To execute your spring boot project using CLI, you need first to download CLI from Spring’s official website and extract those files. You may see a bin folder present in the Spring setup which is used to execute your spring boot application.

As Spring boot CLI allows you to execute groovy files, you can create one and open it in the terminal.

And then execute ./spring run filename.groovy;

**51. How to ignore null values in JSON response spring boot?**

In order to ignore null values in JSON response, you can use @JsonIgnore annotation.

The field can be ignored while reading JSON into Java objects as well as while writing Java objects into JSON.

**52. what is the rest controller in spring boot?**

The @RestController annotation is used at the class level.

You can use @RestController when you need to use that class as a request handler class.All the requests can be mapped and handled in this class.

@RestController itself consists @Controller and @ResponseBody which helps us to remove the need of annotating every method with @ResponseBody annotation.

Below is a simple example to refer to for use of @RestController annotation:

@RestController

@RequestMapping(“bank-details”)

public class DemoRestController{

@GetMapping(“/{id}”,produces =”application/json”)

public Bank getBankDetails(@PathVariable int id){

return findBankDetailsById();

}

}

Here, @ResponseBody is not required as the class is annotated with @RestController.

**53. How to handle 404 error in spring boot?**

Consider a scenario, where there are no stockDetails in the DB and still, whenever you hit the GET method you get 200(OK) even though the resource is not found which is not expected. Instead of 200, you should get 404 error.

So to handle this, you need to create an exception, in the above scenario “StockNotFoundException”.

GetMapping("/stocks/{number}")

public Stock retriveStock(@PathVariable int number)

{

Stock stock = service.findOne(number);

if(Stock ==null)

//runtime exception

throw new StockNotFoundException("number: "+ number);

return stock;

}

Now, create a Constructor from Superclass.

Right-click on the file -> Go to Source ->And generate constuctors from superclass-> and check the RuntimeException(String)-> and generate.

And add an annotation called @ResponseStatus which will give you 404(not found) error.

package com.greatlearning;

import org.springframework.http.HttpStatus;

import org.springframework.web.bind.annotation.ResponseStatus;

@ResponseStatus(HttpStatus.NOT\_FOUND)

public class StockNotFoundException extends RuntimeException

{

public StockNotFoundException(String message)

{

super(message);

}

}

Now, you can hit the same URL again and there you go, you get a 404 error when a resource is not found.

**54. How to do pagination in spring boot?**

The process of dividing your data into small and suitable chunks is Pagination.

One can achieve pagination by using PagingAndSortingRepository which is an extension of crudRepository.

**55. What are spring boot annotations?**

Spring Boot annotations are nothing but the supplemental data/information about the program. It is not directly part of the program and hence does not affect the compilation or execution. In other words, it is kind of the metadata that supplies us with the data of the program. Some important Spring Boot Annotations are:

Spring Framework Stereotype Annotations

Core Spring Framework Annotations

**56. Which is the spring boot latest version?**

The latest version of spring boot is 2.4.0. It came out with a lot of dependency upgrades, java 15 support and much more.