1) Tell us about yourself

-> Can you tell us about yourself or can you talk about your work experience while walking us through your résumé, no matter what the question is and how it is asked, this icebreaking question is a great opportunity to market our skills and all the great work we have been doing over the years.

-> It is like fishing where we will leave several technical baits hoping that the interviewer will be caught in one or most of them and ask us questions from those topics as the interview progresses.

-> This is the very first question we will be responding to, and it conveys a lot about our communication, confidence and experience.

-> So we should prepare six to 12 points and practice those points 30 to 40 times, at least in the next lecture.

-> I will share some important points from my about me that I have been using over the years that will help you.

-> And you can use it as a template if you want to.

----

2) About Me Preparation Template

-> I'll walk you through some of the important points from my about me, which I have been using over the years, I'll always start by saying I have started my career as a Java developer in 2002.

-> Since then, I have worked across domains ranging from ERP to CRM banking to finance, project management to health care and travel to gaming.

-> This point tells them my overall experience and also my experience across various domains, which can be very important for certain roles.

-> Next, I move on to the technology side where I state while working across the Java technology full stack, I have designed and developed applications using various spring boot modules and also frontend java script frameworks like Angular and React.

-> So depending on whether it is a full stack position or if it is a backend position, you can change this point slightly.

-> And based on your experience, whether you are a junior or senior developer etc, you can add more technologies here if you want to.

-> Next, I have worked closely with the product owners and business analysts to get the requirements right and with the architects to implement the non-functional requirements like security, scalability, reliability, etc., This point makes you a real good professional at the same time.

-> It also showcases that you are not just a simple developer, but you can interpret the requirements
and also, if required, you can work with the architects to implement the non functional requirements.

-> Again, if you are a junior developer, you can take out the nonfunctional side of things if you are
not into it yet.

-> Next, we'll move on to the devops side of things where I state
I have also worked with devops engineers to set up the CI/CD pipeline for our project using tools
like Maven,Docker, Jenkins' and Kubernetes.

-> So these are all the bits we are leaving them so that they can ask us more questions on these
later on.So we should be thorough in those topics.

-> In some cases, we may not have the devops team. We ourselves will set up this whole thing.
So depending on your project, you can change this point accordingly.

-> But the key here, state your devops experience and your CI/CD experience and the tools you have
used.

-> I have also worked with various aws components, such as EC2 , S3, EKS and so on, to deploy our apps to
the cloud.
You can modify this to the cloud you have worked with pretty much every job description has a cloud
requirement today.

-> So with which our cloud you are familiar with, you can use that here instead of aws so you can replace
it with Azure GCP, etc..

-> Some projects that do not need cloud, they might be using Kubernetes for their deployment or they
are still migrating to the cloud.

-> In such cases, you can skip it if you do not have that experience.
But if you do have that experience, please state all that in detail.

-> Moving on to the agile side of things, I have participated in several Agile conferences and played
a key role in implementing Scrum and other agile practices like tdd pair programming, etc. So depending
on your experience, you can state your agile experience here.

-> You can say how you have followed the scrum ceremonies, the planning meetings and all that.
You can put it into one point, which is very important.

-> As soon as you get onto a team, they expect you to immediately become agile and be a part of their
agile process that they are following last and very important.

-> I am a continuous learner.
I have recently learned creating serverless projects using AWS lambdas.

-> You can state whatever you have learned recently as this last point because teams need continuous learners
as the industry evolves, as new things evolve.

-> And also, I also like sharing what I learn and I do it through my YouTube channel and blog.
This has helped me tremendously over the years.

-> Not many people share from what I see in the industry, so when you love to share.
People love it because they need mentors.

-> Once you get onto the team, they allow you to share what you are learning on a continuous basis and
also learning from others as well.

-> So whatever you are doing, please add those points at the end.

-> And if you do not have a blogger YouTube channel yet, please do create one right away so you can use
this as a template by downloading it from the resources section of this lecture, but create one of
your own because you are unique.

----

3) Your recent project

-> Many were asked to explain about your most recent project.

-> Always remember that it is another great opportunity to highlight and sell the skills that you have
used on your recent project for the position you are being interviewed for.

-> Start with the high level use case of the problem you are trying to solve.
For example, if you are working on a flight reservation and checkin application, we can tell them
that our application allows the end users to book their flight tickets and check into the flights online
or through their mobile applications.

-> You then draw the high level architectural diagram, which will show them the various components of
your application.

-> Feel free to use the whiteboard if it is a Face-To-Face interview.
If there is a whiteboard available, grab the whiteboard with their permission, of course, and draw
the high level architectural diagram and show the flow in between these components.

-> If a whiteboard is not available, use a white sheet of paper.
If it is a virtual meeting, feel free to share your screen and paint the picture.
This will leave a lasting impression on them.

-> As you showcase your presentation skills, you can then talk a little more about how you have implemented
these components.

-> If it is completely a Java based full stack development, talk about it.

If the backend is in Java and the Frontend is developed using other JavaScript frameworks, talk about them as well.

-> You can then jump into the layered architecture of your Java back end typically will have the data access layer services layer where the business logic leaves the presentation layer and the integration layer where you can use restful web services or messaging.

-> You can also talk about each of these layers and the technologies that you have used in these layers. This is very important and this is where you can highlight those technologies that are really required for the job on hand.

-> For the data access layer typically will use spring data jpa with hibernate for the restful web services layer.

-> We would have use springboot web or frameworks like Jersey and the list goes on and on. So we have a lot of technologies which fall into these layers.

-> Again, this is where your responsibilities on your resume under your current project will come in. So go through all those responsibilities and highlight those portions which are required for the current job opening, which you are being interviewed for.

-> Don't stop with just the development side of things. Talk on how you how written unit tests using unit test frame works like Junit, Mockito, etc., how we have reported the test coverage of those unit tests, the servers to which you have deployed your application to.

-> If you did that, if you have used messaging, talk about the messaging provider or the broker you have used and if you have developed the front end using Angular react etc do talk about that as well.

-> Once you are done with the development side of things, move on to the devops side of things. This is where you highlight your skills, working with one of the build tools like Maven, Gradle, etc. Talk about how you how Dockerized your project, how you have created the docker file or the docker compose file for the application components within your architecture.

-> And if the job requirement needs kubernetes, highlight that as a container orchestration tool.

-> How you have depolyed your containers to the kubernetes cluster if it needs aws or if your last project has used one of these clouds, highlight that as well.

-> Keep the conversation interactive Don't let it be a one side affair.

-> You can always stop and ask them if they want to know more details of any of these as you mention them, especially you can ask them if they want to hear more about how you have

implemented these layers at
a very low level.

-> If they say yes, you can walk them through the classes and interfaces which you typically create in
a Java application, starting with the model classes, then the data access layer classes, as the service layer
classes, the restful and Web controllers, the utility classes in your application, the validators
views and many more.

-> This will give them a lot of confidence that you are a very hands on guy and you are ready to start
coding whenever it is required.

-> Also, do talk about how you have implemented the non-functional requirements or the ILities, as
we call them, the security, scalability, reliability, adaptability when it comes to your project.

-> If not, they will ask you these questions later on anyways.
They will ask you how did you secure your recent project or what are the security best practices?
How do you ensure that your application is scalable?
And all those questions will come in throughout the interview.

----

4) Resume Preparation Tips and Template

-> In this lecture, I will give you some résumé preparation tips and also share a resume template
I have been using over the years in the USA.

-> Before we go ahead and do that, launch your Web browser and search for Java developer resumé sample.
You can even be specific here.

-> You can say Junior java developer, Senior Java developer, java lead architect and so on.
Depending on where you are in your career, go to the images part and you will see hundreds and thousands
of resumes you can use as a reference.

-> There is no hard and fast rule.
You can be as creative as you can, but make sure that your resume talks to the recruiter or the interviewer.

-> Here is a template I have been using over the years.

-> I always start with my name and my contact information so that they can reach out to me immediately.
This YouTube channel I mentioned right up top has been helping me not only qualify for the interviews,
but to clear the interview rounds very easily because the recruiters and the interviewers already know
my technical knowledge and communication skills even before they talk to me.
So I suggest you create your own YouTube channel, your own blog, and share it right on top of your
resume.

-> This visa status is important in certain countries and for certain roles you can mention your visa status,
    especially in the US.

-> Certain positions need certain immigration status that will help you with that.

-> Next, you can directly move on to the project experience and talk about your proje->ct and the roles
    and responsibilities.

-> But I personally like sharing my overall experience right on top of my resume.
    This is like a mini about me where I summarize my overall experience.
    Very first line tells them that I'm a senior Java consultant with 18 plus years of IT experience and I
    have experience with them devops tools and aws cloud components.

-> This point makes me a good fit for several positions which are out there today.

-> Then I also show them that I have not only the development experience, but deployment and client
    supporting experience as well.

-> I summarize my object oriented design experience and Hands-On experience.
    Several recruiters and interviewers want to make sure that you have a hands on experience, not just
    theoretical knowledge, and they want to ensure that you haven't lost touch with your hands on work.

-> Then I tell them that I have worked closely with the architects and also the product owners in implementing
    behavior driven development BDD.

-> So depending on the project and the methodologies you have been following, you can mention that here,
    again, depending on whether you are a junior consultant, senior consultant, you can tweak these points
    and feel free to remove whatever you want to from within this template and add to it as required.

-> Here I mention I summarize my Devops and AWS experience, and I also tell them that I have experience leading complex
    projects as well.

-> I summarize this portion with my uml, design pattern experience following that, you can
    talk about your education qualification and all the certifications you how should be on the very first
    page of your resume.

-> This table here on the second page is optional, but I like having it here because several recruiters
    will do a quick control F and search for a particular skill because there are not technical, they are
    not expected to be technical.

-> So they do search for skills like Spring, Springboot, etc. So it will help them.

-> If you have a table like this, they can quickly look at this table to see if you have all the tools
    and requirements they have on hand and then you can move on to the project

experience where you can mention
the client the location, then the role you have played there, the project name if you want to,
and summarize the environment, which is very important.

-> It tells them that the environment you are currently working on and they can see if that fits into that
environment they are in right now.

-> So this is where talking to the recruiter, going through the job description will help you.
And you should highlight all the technologies that are expected from within the job description or what
the recruiter sees within your environment and within your roles and responsibilities.

-> Even when you talk about your overall experience right here, ensure that whatever is the position you
are working with right now, you highlight all those requirements in these responsibilities here.

-> And also, when you go talk about your latest project, do mention about all the important tools,
you know, and what they are expecting.

-> Under the roles and responsibilities, I start by telling them that I have designed and developed.
rest ful API, which is required for pretty much every microservice today using the spring framework
tools.

-> I also tell them about my frontend experience with Angular and react.

-> And as a senior consultant, you are expected not only to just implement and develop an application,
but also work with non-functional requirements.

-> So I showcase all that here how i configured hibernate fetch and plus strategies to improve the performance of
my application by configuring, caching, fine-tuning, the various layers of the application and so
on.

-> In this point, I summarize the unit testing and integration testing tools that have worked with which
they definitely will look for.

-> I also tell them that i have dockerize the application and deployed to aws using various components.
And my kubernative's experience, is also summarized in the last point.

-> So depending on the Cloud you have worked with depending on the role you are applying for, you can pick all
these points.

-> You can then quickly summarize all your earlier projects.
I like leaving my roles and responsibilities for one more project, but that is not required.
You can just tell them the project.

-> You have worked for the location and quickly move on to the other projects as well so that it tells
them that you have worked across domains and the overall experience will be portrayed.

So feel free to use this template, tweak it as you need it.

CORE JAVA

1) What are the important components of java

-> What are the various important components that are involved in compiling and running a Java program?

-> When we start working on java applications, the very first software that we download and install is a JDK, which stands for Java Development Kit.

-> The JDK is comprised of JRE and it also comes with several comments or programs such as the Java compiler, the Java command that we use to run a Java program and more.

-> We download the JDK and install it for a particular operating system.

-> There are different versions for different operating systems for the JDK and the JRE, we can install JRE separately as well, but usually we install the JDK and the JRE.

-> When we compile the Java program, the Java compiler will convert the Java class into bytecode, not the machine code, which the underlying operating system understands, but the byte code, which is platform independent, the bytecode is only understood by the JVM.

-> The Java Virtual machine, which we launch using the Java Command.

-> When we run our Java class file that gets generated during compilation with Java Command, it will launch a JVM that can interpret the bytecode to the underlying operating system that makes Java platform independent.

-> A JVM is an instance of the JRE.
It is a representation of the Java runtime environment.
So the platform independency comes because we can compile a Java program on a Windows operating system and then we can take the classes and run them on Linux using the JVM on Linux, because the bytecode is understood by the interpreter or the JVM on Linux, and it can convert it into the machine code, which Linux understands.

-> JIT, which stands for Just in Time compiler, is another important component, which is a part of the JVM that will interpret the code to the underlying operating system in a optimized fashion.

-> It knows much more details about the operating system and it converts the byte code little bit at a time by default, if we don't use the JIT interpreter or compiler, the JVM will translate the entire bytecode into machine code and it will execute against the operating system.

-> Whereas if we use the JIT, it converts a little bit of bytecode at a time

and then it can do certain
    optimizations by inlining functions, etc., which makes our application perform better.

-> By default, the JIT is enabled when we use a Java command if you want to disable it for debugging reasons
    or whatever, you can use the hyphen D option javadot compiler is equal to none.

-> This will disable the JIT.

----

2) What are constructors

-> What are constructor's how do they differ from other methods of the class, can we invoke one constructor
    from another constructor and can we invoke the super class constructor from the child class constructor?
    Let's take a look.

-> A constructor is a method that is used to initialize the properties of an object when it is created.
    It has the same name as the class name, unlike the other methods in this case, Class Accord , the constructor
    is also Accord.

-> But other methods start and stop, have different names of their own, but the constructor name is the
    same name as the class name.

-> The constructor is only invoked when instance or the object of that class is created, whereas the other
    methods can be invoked as many number of times as we want.

-> To invoke other constructor's in the same class, we used the this method and we can pass in the parameters
    just like any other method.

-> And if the class accord extends Class Honda to invoke the super classes constructor the hondas constructor
    from within the class constructor, we use the super method and pass in any parameters.

----

3) abstract class vs interface

-> What is the difference between an abstract class and a interface, if a class has at least one abstract
    method, then that class needs to be marked as abstract class, if not the class wont compile any class
    that extends abstract class should then provide the implementation for these abstract methods in a
    abstract class.

-> If not, that class should also be marked as it abstract class.
    Abstract class can have any number of methods with implementation, but as soon as it has one abstract

method, it becomes abstract class.

-> Whereas in interfaces all the methods are abstract.

-> The class that implements the interface should provide the implementation for those abstract methods.
A class can implement any number of interfaces, whereas it can only extend one abstract class.

----

4) Why is multiple inheritance no supported

-> Is multiple inheritance not supported in Java, multiple inheritance is not supported because if we
inherit from two classes, which have the same method, for example, here we have the rich confused kid class,
which extends father and mother, which have the same method money.

-> Now, when we invoke the money method using the rich, confused child's instance or object, the compiler
will not know which money method it should bind to or which money meant that it should invoke.
That is one reason why multiple inheritance is not supported will get a compile time issue.

-> If we try to do this to make things even more complex, we have the classic diamond problem that we
see in languages like C++.

-> If we have a grandparent class which also has a money method, then the compiler will find it even more
difficult to decide which money method should be invoked when the rich confused states object is used to
invoke the money method.

----

5) Can a class implement two interfaces with the same method

-> Can a class implement two interfaces with the same exact method signature?

-> Yes, here I have an interface called Car, which has the go method, which is void and also a avoid
stop method.

-> And I also have a driverless interface with sit down and relax method and a void go method, the same
go method we have in the car interface as well.

-> Now the Honda class implements both the car and the driverless.
It has no issues because it has to provide a implementation for the go method.

-> So it overrides the go method and it provides the implementation.
So we don't have the typical Diemen problem, which we see in case of extending classes.

----

6) What are the Object class methods

-> What are the different methods that every class inherits from the object class, the different methods
that every class inherits from the object class are equals hash code finalize clone and two string.

-> We also have wait, notify, etc. Those cannot be overridden, but we can use those methods in multi
threading.

-> But these methods equals hash code finalize.
Clone and two string can be overridden.

----

7) What is the Default hashCode implementation

-> If we do not override the hash code method and provide implementation for it in our class, what is
the value that will be returned when we create objects of that class and invoke the hash code method?

-> The default hashcode method in the object class returns a value a number that is nothing but the memory
location of that particular object.

-> So if we do not provide our own Hash code method, the value that will come back is the memory location
of that object.

----

8) What is the default toString implementation

-> What is the default implementation of the two string method do? If we do not override the two string method
in our classes and if we try to print an object of that class, then we will see a value which will
have the class name, followed by the @ symbol and the hexadecimal representation of the memory location
of that particular object.

-> So that is the default implementation in the object class.

-> The two string method is implemented in such a way that it will always return the class name followed by
@ symbol and then the hexadecimal representation of the objects address.
And if we override it, then we will see whatever implementation we provide, whatever we return back
from the two string method that will be used instead.

----

9) equals method vs == operator

-> Another important interview question is the differences between the equal two operator the double equals operator
and the equals method when we compare objects.

-> Let's take a look at the double equal to operator by creating two objects of user.
Here we are creating two instances, U1 and U2 of a class called user.

-> And setting the IDs to one and John, both of them have the same I.D. and name, but they have their
own memory locations on the heap.

-> Now, if we use the double equal to operator to compare these two objects, it will return a boolean false because
the double equals compares the object references or the memory locations and not the contents within the object.

-> So you can call it a shallow comparison and not a deep comparison
Now, if you use the equals method U1 dot equals U2 the equals method is available for every
Java class from the object class.

-> Every class implicitly extends the object class and the equals method is derived or inherited from the
object class.

-> The default implementation of the equals method uses the double equals operator.
So it is a shallow comparison by default.

-> So even this one, this statement U1 dot equals U2 will return a false.

-> We have to override the equals method in our classes, the user class, and provide the implementation
to compare the id and name.

-> There is an exception for string's primitives and enums, for the wrapper types strings and enums the
equals method is already overridden to do a deep comparison.

-> For example, if we create two strings with the same contents ABC and ABC. S1 double equals S2 will
return false because we are using a new operator to create a string.

-> In both cases, they both have two different memory locations.

-> This will return a false, but if we use the S1 dot equals S2, it will do a deep comparison because the
equals method is overridden on a string class and it will return a True.

-> Similarly for the wrapper types int I1, equal to one, two, three, integer two is equal to one, two,
three.

-> If we do a I1 double equals I2 it will return a false.
But if we do I1 dot equals I2 It will return a true.

-> To summarize, the double equals operator compares or looks for the references or compares the references of
two objects by default, the equals method also does the same.

-> We should override the equals method and do a deep comparison and return boolean value on our own for
the string primitive types and enums.

-> The equals method will do a deep comparison of the contents.
----

10) final finally and finalize

-> What are the usages or differences between final finally and finalize?

-> Final is a key word that when we mark variables, for example, primitive types, they become constants.
We cannot do this once we define int i is equal to one, two, three as final.

-> The value of I cannot be changed if we use it against an object declaration.
The object reference cannot change later on.

-> We cannot point the object to a different memory location.
When used against a class, the class cannot be inherited and when used against a method, we cannot
override that method in a child class.

-> Finally is a block when we do exception handling, we use it along with try catch, we can also use
finally without a catch directly, with try.

-> Finally gets executed even when there is an exception and also and there is no exception, it gets executed
always.

-> So it's a good place to put all the clean up code where we can clean up all the DB connections.
The socket connections are input output files, streams.

-> Starting Java 7, we have a try with resource block, which, when used, automatically closes
all the resources that are declared inside the resource block.

-> So starting Java 7, if we use the try with resource block, then we don't need the finally.

-> Finally, we have the finalized method, which the JVM calls when the garbage collector is about to be
called, but we should not rely on this to clean up all our resources in the program or application
because we never know when the JVM is going to call the garbage collector.

----
11) What are generics

-> What are generics and what is type erasure?

-> Prior to Java one point five version, this is how we defined a collection here, have a list into
which I want to store all the employee I.D's., which are integer values.

-> when I do this, it is fine, I'm adding an integer even when I do this, where I'm trying to add
a string accidentally, the name of the employee that is allowed as well, because there is no way we
are mentioning the type.

-> That is where Generics came in starting Java version
One point five.

-> Using generics, we can specify or abstract out that type of data that can go into the collection.

Now, if we try to add an integer, no compile time issues, if I try to add a string, the compiler will catch it right at compile time and throw an error.

-> Type Erasure is the process of the compiler erasing this type for runtime, once it does, it checks the compiler will erase this type because of compatibility reasons.

-> Because we want the versions, the older version code probably the code written on Java one four or Java one three to run on the new JVM's

-> So that is the reason for backward compatibility reasons runtime Generics has been compromised or runtime generics is not present in Java.

-> It is all compile time Generics. So type erasure implements compile time Generics by erasing the type for runtime so that the older versions of Java code, which is compiled on JDK one four or one three, can still run on the newer JVMS.

COLLECTIONS

1) What are the different collection Types

-> What are some of the important interfaces and classes in the collections API?

-> We have list set queue blocking queue and map. list allows duplicates and the implementation classes of a list interface are array list and linked list. set does not allow duplicates.

-> We have hashset linked hashset.
We also have a sorted set that is implemented by tree set, which sorts all the objects or data that is stored into that data structure.

-> Then we have first in, first out data structure, which is the queue, which is also implemented by a priority queue.

-> We also have a blocking queue that helps us implement the producer consumer pattern very easily.
We then have the map, which allows us to store data as key value pairs, the most used one is the hash map.

-> We also have a linked hash map and a tree map, which is a sorted map.

---

2) ArrayList vs LinkList

-> What is the difference between an array list and a linked list?

-> An array list uses an array for its internal implementation, whereas a linked list has its own custom, note type with a previous and next pointers, it is like a doubly linked list.

-> Since an array list uses an array
for adding an element randomly at any location, it can get very expensive all the way to big O of n,
because we'll have to shift the current element in the array to insert an element at that location,
whereas for Linked list it is very easy.

-> We simply create a new node and point the previous and next pointers appropriately.
But when it comes to random access, arraylist is super quick because it is backed by an array, we
can access any element by using the index, whereas accessing elements can get expensive in case of
linked list will have to traverse through the doubly linked list.

-> So usually in applications that are very read intensive, array lists are the way to go.
If we have a lot of read operations with a few right operations, then arraylists are the way to go.
If not, you can use Linked lists.

---

3) Vector vs ArrayList

-> What is the difference between vector and classes like arraylist and linkedlist and hash table versus
hash map?

-> These hash table and vector classes are from previous versions of Java, where all the methods on them are
synchronized, meaning if one thread accesses one of the methods on these objects, no other thread will
be able to access them until the first thread finishes that will degrade the performance of our application.

-> They are thread safe, but they can be very slow. That's the key difference. These classes are not synchronized.
Their methods on these are not automatically synchronized.

-> We have to take care of thread safety as per our application needs.

---

4) HashMap vs LinkedHashmap

-> What is the difference between a regular hash map and the linked hash map, the linked hash map maintains
the order of elements in which we add them to the hash map, whereas the hash map doesn't do it.

---

5) How to create a Generic Class

-> How can we define our own generic class to create our own generic class
We create a class just like any other class right next to the class name.
We specify a placeholder for the generic type within less than and greater than symbol.

-> This can be any alphabet, but by convention we use T for type once
We have that placeholder.

-> We can use that placeholder to define fields, parameters to constructors,

parameters to methods or
    even return types of a method

-> And to use this generic class, you create a instance of that generic class.
    And when you do that, you specify the type you want to use.
    This is very specific.

-> The generic type which will be replaced here, that T will be replaced with whatever type you want to use
    and you will pass in the appropriate data and use that generic type and you can use the same generic type
    with different data types or different types of data.
----

6) Failfast vs Failsafe Iterators

-> What is the difference between fail fast and fail safe iterators, traditional collection classes like
    array lists, array set provide us the implementation of failed fast iterators.

-> That is when we invoke the iterator method on these collections.

-> The iterator that is returned back is a fail fast iterator.

-> If we use that iterator and loop through the elements, try to access those elements and at the same time,
    if we try to modify that collection, add, delete, etc on that collection, whether it is in a single
    threaded environment or a multithreaded environment, will get a concurrent modification exception fail
    fast.

-> Iterators will not allow parallel modification to happen, will get a concurrent modification exception.
    But when we use concurrent collection classes like copy on write Array list or copy on write array set,
    these classes provide us with the implementation of failsafe iterators.

-> They Return a Failsafe iterator, which will allow parallel modification, whether it is a single threaded environment
    or a multithreaded environment, will not have any issues iterating through and modifying
    the collection.

-> At the same time, we see no exceptions.Those are called failsafe iterators.
----

7) Producer Consumer Pattern

-> Which collection class would you recommend to implement the producer consumer pattern?

-> Blocking Queue is the collection class that should be used to implement the producer consumer pattern,
    the blocking queue has a put method that can allow the producer to add work to the queue, and this method
    will block if there is too much work on the queue and if the consumer has not already consumed a lot
    of work.

-> On the other hand, the consumer will use the take method that will block on the queue.

    If there is no more work to do, it will wait for the work to come in from the producer.

-> So blocking queue is the collection class that will make it super easy to implement the producer and
    consumer pattern.
----

8) Comparable vs Comparator

-> What is the difference between comparable and comparator interfaces?

-> The comparable interface provides a class the ability to define natural or default, ordering of its
    objects, comparable interface is from Java dot lang package.

-> When a class implements this interface, it needs to provide the implementation for the compare to
    method that will be used to compare tow objects of that class.
    And this methods Logic should be in such a way that it returns a negative value.

-> If object one has to come before object two a positive value object one has to come after object two and zero
    if both these objects are same.

-> So we define our own logic and return these values depending on how we want the comparison to happen.
    So Comparable provides the natural or the default ordering.

-> And if we want to create any number of custom ordering for how the object should be compare, then we
    use the comparator interface, create our own comparator.
    This comparator interfaces from Java.util package.

-> And when we create a comparator class, we need to provide the implementation for the compare method
    from this interface, which takes two objects, compares them.

-> And even here we return negative value if object One has to come before object two positive object

-> Two has to come before object one and zero if both are same.
    When we add our objects to collections, that is where these two are very powerful.

-> If you use collections like treeset tree map, etc. and when you add your objects to them.
    If you do not provide a comparator, then the default ordering or the natural ordering provided by the
    comparable interface.

-> If your class implements the comparable interface and gives a comparitive method, that ordering will be
    used.

-> And if you pass in a custom comparator you create, then that comparator will be used and to compare
    then compare method inside it will be invoked to figure out how your objects should be sorted.
----

9) What are concurrent collections

-> What are concurrent collections when we use collection classes like array list. hash set, hash map in a multithreaded environment, when the first thread gets a lock on one of these objects, no other thread will be able to access the entire object unless the lock is released.

-> Also, these collection classes implement fail fast iterators.

-> That is when we use one of the iterators written by these classes to iterate through the elements and also parallel try to modify that collection will get a concurrent modification exception.

-> That is where the concurrent collection classes like Copy on write array list, copy on write array set come in. And these classes allow multiple threads to access the same collection and modify it at the same time by creating a copy of that collection. And they will sync up the copy with the original collection later on.

-> And these collection classes, the concurrent collection classes will implement failsafe iterators, that is if we get that iterator from one of these collections.

-> Iterate through the elements, we can also modify the collection at the same time, all those problems with the traditional collection glasses are gone in case off a map we have the concurrent hash map, which allows fine grained locking instead of locking the entire map.

-> when there are multiple threads the locking happens at a very fine grained or a bucket level as required.

Multi Threading

1) How to create threads

-> What are the different ways in which a thread can be created?

-> There are two different ways in which we can create a thread.

-> The first is where our class will extend the thread class and override the run method. The threading logic will go inside in this run method.

-> We then create a instance of our thread class and invoke the start method on it. The second way is to implement the Runnable interface instead of extending the thread class.

-> This gives us the capability of extending any other Java class if required in case of extending a thread class.

-> We cannot extend any other class if required in our application because Java doesn't support multi class inheritance or multiple inheritance.

-> So in case of implementing a Runnable interface, we will override or provide the implementation for the run method from this Runnable interface. That is where the logic leaves.

-> Then you create an instance of this class and pass that instance to the

thread instance or object.
    You create instance of the thread.Pass that class instance to the thread instance invoke thread dot start.
----
2) What is Synchronization

-> What is thread synchronization when we have multiple threads accessing the same object or data?

-> There is a chance that these threads might corrupt each other's data or objects.
    That is where synchronization comes in.

-> And once we mark our methods using the synchronized keyword, the first thread that enters the
    method will attain a lock on that method, as well as all the synchronized methods in that thread class so
    that no other thread will be able to access those methods until the first thread finishes its job.

-> We can also have synchronized key word on static methods and at a block level as well
----
3) What are class level locks

-> What is a class level lock, every class in Java has a unique lock associated with it.

-> When a thread executes a synchronized static method, it will attain that class level lock.

-> All the other threads will have to wait to access this static, synchronized method and any other synchronized
    static methods on that class.
----
4) What are synchronized blocks

-> What are synchronized blocks?

-> When we use synchronized key word on a method, the entire method will be locked and can be used only
    by one thread at a time.

-> If we know exactly which lines of code within that method needs locking, instead of locking the entire method
    we can use a Synchronized Block instead.

-> And when we use the synchronized block, only those few lines of code within the block will be locked.

-> And when we do that, we can pass in an object or a class on which we want to attain the lock as well.
----
5) How do threads Communicate

-> How do threads communicate with each other?

-> Threads communicate with each other using wait, notify and notify all methods when a thread invokes a wait method, it gives up the lock on a particular object and an another thread can take over that lock and do its work.

-> Once the other thread finishes its work, it will use the notify method to notify the other thread which gave up the lock that it has finished its work and it can continue doing its job.

-> It can also use notify all method if there are multiple threads waiting for that lock and they gave up the lock for this thread, it can use the notify all method which will notify all those threads to use this wait, notify and notify all methods the threads should be in a synchronized context ,either in a synchronized block or a synchronized method.

-> Otherwise, a illegal monitor state exception will be thrown.

Java 8

1) Features

-> What are some of the java 8 features you have used?

-> I have used lambda expressions, functional interfaces default methods, predicates functions and streaming API.

---

2) What is a Lambda

-> What are lambda expressions and why use them?

-> Lambda expressions bring the functional programming syntax to Java, they are like anonymous functions or closure's where we don't have to use a function name, a return type or even access modifiers like public, private, etc. What typically needs a lot of lines of code can be easily achieved through a lambda expression.

-> Lambda expression starts with parentheses, followed by a arrow symbol hyphen and greater than symbol, followed by the body of the lambda code.We want to execute within the lambda expression.

-> We can also pass parameters to the lambda function, just like how we pass parameters to a function.

-> The advantages of using lambda functions are very few with very few lines of code. We can achieve a lot of things very easy to implement.

-> Anonymous inner classes wherever we want, anonymous inner classes. We can create them on the fly using lambda expressions and they can be even passed as parameters to other functions.

---

3) What are Functional Interface

-> What are functional interfaces if an interface has only one abstract method, then that interface is
called functional interface.

-> Examples of inbuilt functional interfaces are Runnable, which has only the run method comparator, which
has only one abstract method, which is compare to.

-> A functional interface can have any number of default methods that were introduced in Java eight, but
it can have only one abstract method.

-> We can mark a functional interface using @ functional interface annotation once we use this annotation
on an interface, if we try to add more than one abstract method to the interface, we'll see compile
time errors.

-> Functional interfaces can be expressed as lambda expressions.
That is a rule that the interface should follow to be expressed as a lambda expression.
It has to be a functional interface.

-> If we want to express that interface as a lambda expression, it should have only one abstract method.

---

4) What is the Use Lambda

-> Can you explain us with an example on how you have used lambdas to simplify coding applications?

-> One of the examples is when we spun off multiple threads within our application, we typically create
a class that implements the runnable interface and provide a implementation for the run method within
which the multithreaded logic will live, will then create an instance of this class when pass it to
the threads constructor are we.And we invoke thread dot start.

-> It involves so much coding just to get a thread working.
Instead, once you start using lambda expressions, it will become so simple.

-> You will say Runnable are is equal to will start the lambda syntax the run method within the Runnable
interface does not take any parameters.So we start with empty brackets.

-> Then the arrow hyphen greater than symbol than the body of the
Function expression, lambda expression where the logic, formality threading will live, we can then simply
pass this runnable instance to a thread and use thread dot start just like before.
So all this creation of class will go away with lambda expression.

---

5) What is a Predicate

-> What is a predicate, a predicate is a function with a single argument that returns a boolean, true
or false back. To create a predicate, we use the functional interface predicate that takes a generic
type, and it has a single method called test that takes the generic type and returns a boolean value

back.

-> Since it is a functional interface, we can express it using lambda expressions.
Here is one example, we use the predicate passing the generic type integer, and on the right side
we use a lambda expression to express this functional interface.

-> The implementation is very simple.
I is the parameter to the lambda expression or the function, and the logic here is I is greater
than 20.

-> So that will return
True, if I is greater than 20, but is less than 20 or equal to 20, it will return false.

-> And here we invoke p dot test on that predicate.
We invoke the test method.

-> This here will be used as the logic, as the implementation for the test method.
We get the appropriate result back predicate can work with different data types.

-> Here is an example of string predicate, which checks if the string length is greater than five, always a
predicate returns a boolean value within the predicate.

-> We can do whatever logic we want with the input, but it has to return a boolean value back.
----

6) What are Predicate Joins

-> What are Predicate Joins?

-> Predicate joining allows us to use more than one predicate together, and we can also negate the result of
a predicate here, have an example where a function takes a predicate as a parameter and a integer
array, and it applies this predicate on each of the array elements and it will print the array value
only if the predicate returns True.

-> And up top here in the main method, we have an array defined and two predicates, one that checks if
a number is greater than 10 and another one that checks if a number is even if we simply apply the first
predicate which checks, if the number is greater than 10, then we will get all the numbers in this
array which are greater than 10 on the console.No surprises.

-> And when we apply the second predicate, we'll get all the even numbers.
No surprises there.

-> But if we want to reverse this first predicate, if we want to get all the numbers that are less then 10
you can simply use P1 dot negate and we invoke that method.
It will reverse the predicates logic or it will use false instead of true however you look at it.

-> So to the method here below, you will pass P one dot negate which will send

you back the opposite
    results.

-> Instead of greater than 10, you will see less than 10 on the console.
    And if you want to apply both these predicates on the array, then you can use P1 and P2.

-> That is, it will return all those numbers or it will display all those numbers on the console, which
    are both greater than ten and which are even if you use the end method on the predicate to accomplish
    that.

-> And finally, if you want or you simply use P1 or instead of P1 and so greater than 10 or even either
    of those.
----

7) What is a Function

-> What is the function? Function is just like a predicate, except for it can return any type of value,
    not just a boolean value we can pass in what type of data it will accept and the return type as well.

-> And we can express it as a lambda expression because it has only one method called apply it is a functional
    interface with a method called apply.

-> Whatever logic we implement using the lambda expression will be used and that value will be returned
    back instead of a boolean value.
----

8) What are Default methods on interfaces

-> What are default methods in interfaces and why do we need them?

-> When we create an interface and that interface is implemented by several classes in the application,
    all those classes will have to provide the implementations for the abstract method in that interface.

-> Which is fine, but in the future, if we change this interface and add more abstract methods, all
    those classes will have to provide a default implementation for those new abstract methods that are
    added to the interface.Otherwise they will not compile.

-> All of those classes should be marked as abstract.
    That is where the default method that was introduced in Java eight comes in inside an interface.

-> When we define the new method, instead of making it to abstract method, we will define it as a default
    method.

-> Default methods can have implementations.

-> It is where we'll put the default implementation so that the other the classes that implement this
    interface, if they want to, they can override this method.Otherwise, they will not.

-> But the code will not be broken, all those classes will still compile because we have the default
    implementation in the interface itself.So default interfaces were introduced.

-> Do not break the classes in the application when the interfaces change or a new abstract methods get
    added to the interface.

-> And if the other classes do not want to use those new methods.
    They can go with the default behavior.

-> The classes, which aren't all right, can always provide their own implementation for these default
    methods.
----
9) Can a class implement two interfaces with the same default method

-> A class implement two interfaces which have the same exact default method signature.

-> It is a conditional yes here I have two interfaces interface a an interface x which have the same exact default
    method default void

-> m1 now if Class B implements those two interfaces.
    Let's see what happens.

-> Class B implements A comma X, we see a compilation problem immediately.
    And the error is duplicate default methods named M1 with no parameters are inherited from type X and A.

-> We go towards the classical diamond problem to avoid this will have to provide a over ridden version of this
    default method in this class B. Click here say over ride, the default method and do provide your own
    implementation here and the error is gone.

-> So you can't just implement two interfaces and then stay calm.
    If you have the same default method in both of them, you'll have to provide a over ridden version of that
    default method.
----
10) How to use Stream Filter

->   How can you filter out the even numbers in a given list using streams?

->   It is super simple to do that we take the given list, invoke the stream method on it, the first step
     to use streams is configuration.

->   In this case, since we want to filter out, we will use the filter method in the configuration step
     pass it a predicate in this case.

->   The predicate is to check for even if it is even, it will return true.
     If not, it will return false.

->   So the filter will filter out all the even numbers from the list and in the processing step we will
     collect all of them using that dot collect method and to the collect method we pass in a collectors

dot to list that will give us back a new list that will have all the filtered elements which are even.
----
11) Other Methods on Stream

-> Have you used any other methods on the stream other than just the filter method?

-> Yes, I have used filter dot count that will give us the count of the filter objects instead of collecting
them to another list.

-> We can simply get the count.

-> We can sort the given stream using the Sorted method.

-> We can pass in a comparator to that sorted method which it will use.

-> We can easily create a competitor using a lambda expression and we pass that comparator to the sorted
method in the configuration step and in the processing step we will collect the sorted list into a new
list.

-> I have also used the max which will return the maximum element and we give it the comparator.

-> It will compare and it will return back the maximum element in a given list.
Similarly, the minimum. Will give me the minimum, the smallest element in a given list.
----
12) Map vs Filter

-> What is the difference between the filter and map method while working with streams?

-> Well, the filter method simply filters out based on a predicate we pass in the map method takes the
given list and it will convert it into a list with different content all together.

-> It will act on the given content and it will transform them based on the logic we pass into the map
method using a lambda expression.

Java 9

1) What are private methods in interfaces

-> Can we define private methods in interfaces?

-> Yes, starting Java 9 we can have private methods on interfaces, the reason or the advantage of
it is to reuse the code across the default methods and the static methods we can have an interfaces.

-> If we want to reuse some code across the default methods that we define, we

define a regular private
    method, not a static one, and then invoke it inside the default methods.

-> But if you want to reuse the code across static methods, default methods, etc, then this private
    method also needs to be static.

-> Only then they will be able to use it in other static methods on the interface.

----

2) What are Immutable Collections

-> How can we create un modifiable or immutable collections in Java. Prior to Java nine
    We have to use the collections utility class and the methods available on that class.

-> But starting Java nine, we have the off method on every collection like list and set.

-> Have the off method which will give us back a immutable or un modifiable collection
    when this method is invoked.

----

3) Stream API Updates

-> Three new methods are added to the streaming API on the stream, we can now use the take while method
    which will take the elements from the stream as long as the predicate that is passed into the take while
    method returns.

-> True, it will stop as soon as the predicate fails or returns false.

-> It will not take the rest of the elements from the stream. Drop
    while is the opposite drop while will keep dropping the elements as long as the predicate returns
    True, and when it returns false, it will take all those elements.

-> And one more useful method is of nullable on the stream which we can use to ignore the null values while
    using methods like flat map.

-> So take while drop while and of nullable are the three enhancements that were done to the streaming API in Java nine.

----

4) Enhancements to try with resource

-> Private the Resource Block was a feature that was introduced back in Java six, if we define any resource within
    that try block, automatically that resource will be closed, provided that resource should implement
    the auto closeable interface and override that close method.

-> That resources close method will be automatically invoked for us when we use the try with the resource block
    in Java nine, it has been enhanced so that we don't have to define the resource inside the try block
    earlier we had to define it like this in here, but we no longer have to do that.

-> We can define or declare the resource itself outside and simply specify the variable name of that resource
    in the try block.

Java 10

-> What are some of the changes that happened when it comes to release starting Java 10. Starting Java
    10 Oracle promises a six month release before Java 10 up to Java 9

-> We had to wait for a long time in between each release and each release
    Used to have a lot of features in them.

-> But starting Java 10 will master only a few features at a time, which will be very easy for the developer.

-> Java 10 introduces us two such cool features using var to define variables and also some API updates
    in the Collectors' API.
----
2) var

-> What is the use of var that was introduced in Java 10. Var can be used to define inferred types.

-> Once we define a variable using a var the type of that variable will be inferred from the type of data
    or value we assign to that variable.

-> No, it is not like JavaScript var in Java.

-> Once we assign a type of value to a variable defined by var we cannot assign a different type to
    it later on in the code that is possible in JavaScript but not in Java.

-> And also VAR is not a keyword.

-> It was left out from the keyword list on purpose because we don't want to break someone else's code.

-> If someone had something like this in their code, for example, int var in earlier versions of Java,
    we don't want that to break their code.

-> So for backward compatibility reasons var is not a keyword.

-> Hopefully in the future they might add it to the list, but for now it is not a keyword var is of great
    use when we define complex connections.

-> So on the left hand side, we don't have to specify the entire collection type
    again, making it much more readable and we can also use it in loops, etc. When we are using complex
    collection classes, we cannot use VAR to define a lambda expression.

-> When we express lambda expressions on functional interfaces, we cannot use lambda expressions.
    It needs to have an explicit type and also we cannot define fields at a class level using the var keyword.

-> We can use var's inside the lambda if we want to, but not on the lambda expression itself.
----
3) Collectors API updates

-> While using the streaming API when we use the filter methods, etc., and create new collections from
existing collections, how to create a unmodifiable list or a set as a result of the collect
method to do that in Java 10

-> we have the collectors dot to unmodifible
list to a modifiable set to modifiable map, etc. These are the new api methods that are
introduced in Java 10 we can use them and whatever comes back, we cannot change them.

-> It is unmodifiable if we try to change it, we will get a exception as follows.
Will immediately see a unsupported operations, exception on immutable collections, so we get our immutable
collection back when we use these to unmodifiable list etc.

Java 11

1) Introduction

-> What are some of the updates that happened in Java 11, Java 11 adds new methods to the string API.

-> It also adds new api s to the file IO packages. it is added is empty on the optional class, which is very helpful.
And with every release, we'll have some depreciations and removals.
----

2) String API Updates

-> What are the new API methods that are added to the string class in Java 11, the first method is the
is blank method, which we can use to check for blanks within a given string, it will return.
True, the complete string is blank.

-> We can have any number of blanks.It will return True

-> In that case, if the string has some characters, it will return false.

-> Next is the streaming update that happened, string dot lines method will take a string which has
new lines, and it will return as a stream by splitting the string based on the new lines.

-> We can then collect, use the collect method on that stream and create a list out of it.

-> Next is the update when it comes to spaces and unicode spaces, not just the regular spaces, when we

use a string dot trim method, which is in earlier versions of Java, it will only trim the normal
characters, not the unicode spaces.

-> So the Unicode spaces represented as follows backward slash
u 2000 it is one way of representing a Unicode space.

-> We can trim and clean up the unicode spaces
Using the strip method that was introduced in Java 11, Strip method will remove both the spaces at
the start of the string and at the end of the string that is leading and trailing.

-> But if you want to be very specific, you have strip leading and strip trailing.
And the last of the methods is the repeat method that will repeat a given string as many number of times
as you want.
----

3) File API Updates

-> Java 11 also makes it easy to write text or string data to a file, we can use the update that
happened on the files class in Java dot neo dot File package, file start.
write.

-> String is a method that takes the path to the file and the content that we want to write to that file
the string content and we'll write it to the file.

-> Similarly, files dot read string will take the path and return as the string data from the files so write
string and read string make it super easy to work with strings while dealing with files.
----

4) isEmpty method

-> Java 11 also adds a is empty method on the optional class that is very useful when we work with reactive programming
in prior versions of Java, we used to have optional dot is present where we can check if this
optional has some value inside it.

-> But starting Java 11, we can also check if it is empty.
Earlier we had to negate that is present to do this type of check.

-> But starting Java 11 is empty Method will let us know if the optional does not have a value for it doesn't it will return.
True, if not, it will return false.

Java 12

1) String API Updates

-> Java 12 introduces indent and transform methods on the string class, the Indent method can
be used for indenting the string that is adding spaces right at the beginning of the string.

-> It can provide positive value, which will add as many number of spaces as you provide.
And if you do a negative value, if the string has spaces right at the beginning, they will be taken out.

-> The transform method takes a function and it will apply that function on the given string, the return type of that function need not be a string.

-> It can return any other data type.So whatever that function returns, that will be the result of the transform.
----

2) Compact Number Format

What is compact number format that is introduced in Java 12? starting Java 12 the number format class has a method called Get Compact Number instance, which, when used for formatting numbers, will express them in a shortcut format.

thousand, for example, will be represented as one k

So depending on the locale, it will use the appropriate shortcuts.

And we can also pass in style like short, long etc while doing it instead of one m and one k.
If you pass in long, it will express it as one million, one thousand and so on.
----

3) More Unicode Chars

-> Java 12 add support for several new Unicode characters to represent the characters in the Chinese language and also the support for representing all those characters on a chess board?

-> Here is an example.If I run this Java code, so these are all the Unicode characters for representing some of the characters on the chess board like that.

-> You can represent everything on a chess board now using Unicode characters in Java.
----

4) Collectors API updates

-> Java 12 adds a teeing method to the collectors class, which takes two down streams and merges their results using the merger we provide.

-> So as we stream through a collection, we can use two down streams.
We invoke the teeing method on the collectors and then you can use two down streams.

-> One downstream here is counting the total number of elements in the collection.
The second down stream is filtering the results and storing them into a collection.

-> Both these results, the counting and this new collection will be stored using this merger class we provide.

-> So the count will go into this variable and the filter new collection will go into the collection inside this
merger class, that is how that teeing method works on the collectors.

Java 15 Features

1) What are Sealed Classes

-> What are sealed classes and interfaces, sealed classes and interfaces is a preview feature that was
introduced in Java 15 using which we can control which classes can implement a particular interface
and which classes can extend a particular class.

-> We do that using the sealed keyword.

-> First, we seal the interface or the class and then we use the permits keyword to specify those classes
which can implement that interface or if it is a class that can extend that class.
----

2) Record Enhancements

Java 15, also enhances the record preview feature, we can use the sealed interfaces and classes
with the record feature as well.

When we define a record, we can implement a sealed interface or extend a sealed class.
Also, we can use custom annotations on the fields that are defined within a record.

When we define a record and define the fields for that record, we can use our own custom annotations
on those fields.

That is also a additional feature that was introduced in Java 15 record is still a preview feature in
Java 15.

Coding Problems

1) Introduction

-> In this section, you will work on some coding problems, when a coding problem is given, it is to
see how Hands-On you are and also to check your logical thinking abilities.So keep it interactive.

-> Instead of directly jumping and trying to write some code, keep it interactive and explain how exactly
you are approaching the problem and how you are coming up with the solution that will tell them how
you can communicate and also how you think.

-> Follow all the standard conventions like naming your classes, methods and variables correctly, make
sure your code is readable and also you can validate the input data for a

particular method throw
    the appropriate exceptions.

-> Make sure your code is secured and all that non-functional things should also be followed when you write
    the code.

-> There can be multiple ways in solving a problem so you can take one approach and tell them how you have
    solved it.

-> And if you can think of any other approach, you can show them that approach as well.

----

2) Find the Factorial

-> Write a program that will calculate the factorial of a given number, we can do it two ways the non recursive
    and the recursive way, let's try both create a new class call.

-> It factorial with that main method finish.
    Hit control D to delete that line, start with a static method, public static This guy returns a int
    call it factorial and it receives a number of which we need to calculate the factorial.

-> The logic for calculating factorial is starting from one go all the way to the number and multiply each
    number with the next number that will give us the factorial to do that.

-> A result start with the result is equal to one starter for loop for a while loop int i is equal to one i is
    less than or equal to that given number I plus plus.

-> Within this loop, we will store the result into the result while multiplying the result star, I so
    a result will initially be one, and then as the i increments will be multiplying with the result and storing
    it in the result, when the next loop comes in or when we go to the next iteration in the loop, the
    result will have the next number that will be multiplied with the previous number and that will be stored
    in the result all the way to the number and finally will return the result back.

-> So by the time this for loop finishes, will have the factorial inside the reult now invoked, this use sysout
    within the sysout use class name factorial dot the factorial method pass in a number five.

-> Run that program. We should get 120 in the result
    There we go.

-> So that's the non recursive way to do the recursive way.
    Public static int recursive factorial int number recursive function always starts with the exit condition.

-> In this case, the exit condition is when the number.
    Becomes zero, num is equal to zero.

-> We will stop and return one semicolon right there.
    Else we will recursively invoke the same function return.

-> Within brackets, inStar recursive factorial n minus one, or num minus one.
    This should be num not n we are calling it num num state recursive factorial num minus one.

-> So initially when it is five, it will come here.
    It will check.

-> If it is equal to zero, it is not.So it will go here

-> Return five star recursive factorial of five minus one four next it will become four star recursive
    factorial three and so on.

-> And at the end when it comes to zero, one will be return.

-> One will be multiplied with whatever the next number is, two, three and so on.
    That's how the recursion works.

-> Instead of regular factorial invoke the recursive factorial function Pass in five run the program, we
    should still get the same result.

-> Let me see what the problem is recursive factorial, I have saved it right here, I have an error, num, I forgot a semicolon added.

-> Run we should still see 120, so that's the recursive way of doing it.
    Remember, any time you use recursion, there will be a exit condition then the recursive invocation
    of the function.
----

3) Generate Fibonacci Series

-> Write a program to display the given number of Fibonacci series numbers.
    So if I give you fifteen, it should display 15 Fibonacci series numbers.

-> To do that, go to your id, create a new class call it fibonacci series with a main method,
    finish double click to maximize control D to the delete that you can create a static method and write all this
    logic there or simply put it in the main method.

-> Int Num one is equal to zero comma.Number two is equal to one comma the fibonacci series
    Next number. A fibbonacci series starts with zero, then one.

-> And the following number will be the sum of the previous two numbers that is zero and one, when added,
    will become the third number, which is one.

-> The following number will be one plus one, which will be two the following number will be one plus two which
    is three and so on.

-> So first will start by displaying the first two numbers sysout.

-> We will use the print statement, we don't need println here, num one, plus add a space in between the number
    plus num two, this will display the first two numbers in the Fibonacci series, which is zero and
    one.

-> Then we'll take the count, the range which we want to go to int count, say, 15.
We want to display 15 Fibonacci series numbers sys out not sysout.

-> We need a loop for loop for int i is equal to zero, i is less than count I plus plus.
And the logic is super simple.

-> First, Will Calculate, the next number in the Fibonacci series using the fib variable is equal to num
one.Plus num two.

-> use a sysout print will not use println and all the fibbonacci series numbers will come on the same line will
add space before each number.

-> So Space plus fib will print the next number once we print it.
Next, we need to switch these numbers.

-> Now, num one is equal to num two.
The next number will be coming into num two.

-> Num two is equal to what FIB holds right now so that the next time the loop comes in, it will add
the previous two numbers.

-> So this should be in the Fibonacci series for a score run, the Java application and see the output
zero one one one plus one two two plus one three three plus two five five plus three eight and so on.

-> That is how we generate Fibonacci series.
----

4) Reverse a String

-> Right, a program to reverse a given string.
Create a new class. Call it string reverser with a main methosd, finish. Double click to maximize.

-> You can start by telling the interviewer that there are classes like string buffer and string builder
which can take a string.

-> Say, ABC, and they can right away reverse them, there is a method on these classes which we can use
out of the box.

-> That is not what the interviewer is expecting, but what you prove your knowledge.

-> He wants you to just use the string class and do the reverse.
So get rid of this and create a new method.

-> Public static it returns back a string reverse is the method name.
It takes a string input.

-> Start by defining a new string result, which will initially be blank, then we are going to loop through
use a for Loop int I will start at the end of the string is equal to input dot length minus one because
the index starts from zero all the way to the length minus one.

-> You will see that in a second.Why we are doing lenth minus one i is greater than or equal to zero i minus minus.

-> I'm starting at the end of the given input string and coming all the way to the first element.
    And within this, I will say the result is equal to result plus input dot character @ character @ method
    and given index will give the element at that particular index in the string character@ i will give
    me the last element

-> Because we are starting with the last element length minus one.
    And as this loop goes through, I will get one element at a time from the back of the string, I keep
    adding that to the result.
----
5) Check if a String is Palindrome

-> Write a program to check if a given string is a palindrome or not, a palindrome is a string, when
    reversed will exactly be the same as the original string.

-> For example, step on No pets when we reverse it.
    It will still be a step on no pets.

-> Similarly, Malayalam as South Indian language spoken in the state of Kerala when you reverse it will
    still be Malayalam.

-> So let's write a program for it.Go to your id.

-> Create a new class call it Palindrome Checker palindrome Checker with a main method.
    Let me zoom that in.

-> Create a new method, public static so that it's a utility method.
    We are going to check and return a false so boolean
    False or true, is palindrome is the method string.It receives the input string.

-> First thing is to validate if the inputs that are coming in are valid, if input double equals null, and also
    check for blank.

-> If we want to throw new illegal argument exception here, you can ask the interviewer if you want to throw
    an exception or do you want me to return a false if the input is null with the double quotes, you
    can say input cannot be null.

-> Hit control one add a written statement.

-> For now, it will return false.
    That's OK.

-> We'll change it at the end.
    Next, we need to reverse the string.

-> The easiest way to reverse a string is to use the string builder class string builder hit control one
    assign statement to a new local variable.

-> You can pass the input to the string builder constructor.

We create a string builder using the string.

-> Next you can invoke string builder dot reverse method that will reverse the string for us to hit control
one as instatement to our local variable.
This is the reverse string.

-> Call it reversed, and we need to compare the original input, with a reversed string, we can do all
that in a single statement, return reversed dot to string.

-> We need to invoke to string method to convert the string builder into a string dot equals pass in the original
input.

-> So that comparison will happen. If it is equals, it will return true.

-> If not, it will return false to invoke it right here, use a sys out
Sysout.

-> Invoke the method, it is a static method so you can use the class name Palindrome Checker Dot is
palindrome.

-> Let me pass in Malayalam all lowercase.
Right, click run as.

-> Java application, and it says true in the output here, the have missed a couple of checks, you can
add two more additional checks in your code.

-> One, you can convert whatever input comes in to.
All lowercase are all uppercase.

-> Let's see what happens.
We are passing M capital for malayalam.

-> And run the program, it says false because here we are doing a equals comparison instead of equals.
Ignore case.

-> either we do equals ignore case or you convert the entire input into lowercase or uppercase.

-> That method is available on the string classes so we can either use that or here when we do the comparison,
ignore the case.Now it returns True.
----
6) Find the numbers

-> Given an array of integers, find out if any two elements are numbers within this array, add up to
the given target number.

-> In this case, you can see that 14 and 18 add up to the given target number, which is 32.

-> These elements need not be sequential one after the other they can be anywhere in the array if found,
return a boolean.

-> True, if not found, then return a boolean false.

-> Create a new class, call it target finder.
With a main method.

-> Control D to delete that public static boolean find numbers is the method that takes a integer array
arr , int target number.

-> Follow the coding along with me and then I'll explain the logic to you, I'm going to use a hashset
here as we process the array elements.

-> I'm going to store some of the elements into this hashset he will realize the reason by the end of
this program, assign the statement to a new local variable, call it processed numbers.

-> And this is of type integer use a for loop.
hit controls space, iterate over a array and we will start from index zero, go all the way to that array dot length
and hit control one on the method add a written statement.

-> By default it will return false within this for loop.

-> I'm going to follow a logic where I will subtract the current array element from this target number.
I'll explain you int required number is equal to the target number minus array
Of I, it might look confusing, but once we finish, you will see that it's very simple and easy.

-> So here I'm subtracting the current array element from the target we are looking for.
So in this case here, 32 minus 30, that will yield two now

-> My responsibility is to check if two exists anywhere in this array to save us some time instead of looping
through continuously.

-> I'm going to do the check right now using this hash set.
So here the logic comes in.

-> If the processed numbers dot contains the difference, which is the required number, then we have found
our number return true else put the current array number into the hashset.

-> So here I will say processed numbers dot add arr of I.
So this way, we need not keep looping through as we put the processing numbers here.

-> We will check if the difference is already there in this processed numbers and if it exists, that means
at some point that there was a number in the aray which, when added to the current number, will result
in the target number.

-> Thats it, that's a simple logic.
If not, by the time this for loop finishes, if this return is not executed, we return a boolean.
False.

-> Let's test it.

-> Go to the main method, use a sysout target finder, dot, find numbers, pass in integer array, new int array within flower brackets, say 20 30 came out of it.

-> So let me pass the target number for, say, 40 is my target number, and then I will go back to the
array here and initialize it 20.

-> For some reason now I go back 20, 30, 10, 20, 50.

-> So here I have five elements and the target number is 40, this 20 and this 20 should add up to 40, so
we should definitely get a result true Run as Java application.
There we go.

-> We get a result true.So when this array is passed here, we loop through the array.
We have a hashset.

-> The target number is 40. So we go here 40 minus the first array element, which is 20.
So the difference is 20.

-> We check if the process numbers has a 20 right now, it will have nothing because we have just started.
So it will go here.

-> Put the 20 this 20 inside that processed numbers hashset.
The same will go on with 30 and 10.

-> When it finds this guy here, this 20, it will again do this check process numbers dot contains target minus
this 20 will be 20 when this check happens.

-> We already have a 20 in the processed numbers.
At that point it will return a true change this to something else instead of 40.

-> If you say 80, we have numbers that add up to 80, which is 50 and 30.
So it should still return a boolean.

-> True yes it does, let's take it to a very high number 150 that should fail it should return false back.
You might be asked to return the exact numbers that have added up to the target number.

-> You simply need to take the current number here.

-> When this is true, you have those numbers, you can put it in a list, you can put it in an array and
you can return them back.

-> You have the required number and you have the current array of i those two other numbers that will add
up to the given target number whenever this condition here is true.

-> So you can return those two back or simply display them here.
Let's do a sysout for now before the return statement.

-> Let me cut it paste it before that and display the required number.
And do a sysout of the array of I.

-> Run the program again, not 150, let me take a positive case say 50.
Run it and it shows me 20 and 30 will add up to 50.

Spring Boot

1) What is Dependency Injection and IOC

-> What is dependency injection and what is a IOC inversion of control?

-> When we develop software applications, we organize our code across components and in case of Java, these components are classes.

-> For example, here we have a product Dao class on the right side, which is responsible for performing all the database operations on the left side.

-> We have a product controller, which is a Web layer class which uses the product Dao instead of we creating the object of this product Dao inside the product controller?

-> We delegate this responsibility to external frameworks or components like spring.

-> Spring at runtime will dynamically create an object of this product DAO and inject it into the product controller.

-> The product controller can then use this product Dao inside its methods this process of External components are modules, creating the dependencies and injecting them into the required classes is known as dependency injection.

-> We as developers need not worry about how to create objects and all that, and we can focus on the business logic instead.

-> And the process of moving this control of object creation from the class to an external component or module like spring is called inversion of control.

-> It is a design pattern where we are giving away or inverting the object creation control from our application code to the external component like spring.

----

2) What are the Spring Bean Scopes

-> What are the two different spring bean scopes, they are Singleton and Prototype. Singleton is where only one instance of a bean will be created for the entire IOC container and the same instance will be injected wherever required.

-> Singleton is the default scope.

-> If we configure the prototype scope for are beans, then multiple instances of the beans will be created and injected wherever required.

-> Which one would you use and why?

-> If our application needs statelessness or if our application classes are beans are stateless, we can go with the Singleton scope like controllers, DAO s, etc., which do not have any state within them.

-> And if they have a state, then we'll have to go with the prototype scope, because if we have state
with Singleton, the data can be corrupted across multiple threads

-> But we don't have to worry about multiple threads in prototype because each thread will get its own
instance of a bean.

---

3) Prototype in Singleton

-> Can a prototype bean be injected into a singleton bean?

-> Yes, once the prototype bean is injected into the single ton bean at runtime, the same instance of the
prototype bean will be used by the singleton bean.

---

4) What are HTTP Scopes

-> What are the different spring bean HTTP context, scope's, they are request session and global. Request
scope is where a new bean instance will be created for every HTTP request coming in.

-> Session scope is where there will be only one bean instance used across the session that is will have
multiple HTTP requests and responses happening within a given session.

-> For all those requests and responses, the same bean instance will be used.

-> The global or global session makes sense only if we are using Portlets within our application and
this scope applies across portlets It's like a global session which we can have across portlets and the same bean will be used
across this global session or Portlets.

---

5) What are the Problems with traditional spring

-> Can you talk about the problems you used to face when you have used traditional spring framework before
Spring boot was introduced?

-> When we use the various modules while working with spring framework like Spring core to do dependency injection,
MVC to do web layered development and DAO, ORM to develop the data access layer, using hibernate, etc.,
we have to use a lot of XML based configuration or Java based configuration to get the job done.

-> This can get cumbersome to maintain the application over time.
And then we also had to make sure that we include all the modules or dependencies that are required
for a project in the MAVEN pom dot xml or a gradle field file and ensure that these modules like the
Core, the MVC, the Dao, ORM are compatible with each other as we move from

one version to another.

-> And finally, once we do all this and develop the application, we had to manually deploy our application
to a external web container like Tomcat or application servers like Web Logic, WebSphere, etc..
So all these were problems before spring boot came in.

----

6) Why use Spring Boot

-> Why did you use spring boot in your project or what are the advantages of using Spring boot?

-> The first advantage or feature of Spring boot is auto configuration, that is we no longer need to write
a lot of Xml or Java based configuration to configure Spring MVC to create a web layer or ORM using
Hibernate Spring data jpa etc.

-> Once we add the required dependencies automatically, the dispatcher Servlet will be configured for
our Web player.

-> A data source will be created for orm layer if we use the spring data jpa and also a transaction
manager is created an injected as well without us writing any code

-> So auto configuration is the number one super cool feature of spring
boot. Secondly, we don't have to worry about the module availability and version compatibility across
these libraries or modules, which was a big headache with the traditional spring applications.

-> Thanks to spring boot starters provided by the Spring Development Team, these starters are ready to
be used in our projects.

-> So if you are working on a web layer or a restful application, you simply add a Spring boot starter
web.

-> Similarly, if you want to use the jpa spring data jpa if you want to use orm modules like Hibernate,
you simply add that starter dependency to your spring boot application.

-> All the other libraries are dependencies that are required for the web layer for using jpa and hibernate
will be transitively pulled thanks to these starters which hide all those complexities from us.

-> And also you no longer have to worry about the version compatibilities across these dependencies.

-> There is a parent project that every spring boot project extends and this parent project will have all that
version information and the compatibility details for these libraries that are required.

-> So the module availability and the version compatibility problems are gone thanks to the starters and
this parent project provided by Spring team
Next, is the deployment, deploying our application once it is ready to production to any other container,

were used to be a headache in case of traditional applications.

-> But Spring boot comes with embedded servlet container when we launch our spring boot application.
If it is a Web application is simply right click to run as spring boot application, that will launch it on a embedded tomcat by default.

-> You can also use embedded Jetti or undertow as your Web containers.
So it's super easy to create and launch your applications using spring boot
Last and very important spring boot also offer actuators which allow us to do health checks.

-> We can see the entire configuration, the auto configuration that is happening behind the scenes for our application using these actuators.

-> Check all the mappings that have happened internally for our application, the information, the health metrics, all that can be retrieved by simply adding one dependancy to our Spring boot Pom or gradle build file.

-> We can do this even in production.
So the health monitoring of our application can be easily done.

-> So using Spring boot not only increases the speed at which we develop application, but the deployment as well as monitoring the application.
----

7) What is @SpringBootApplication

-> Can you talk about the Spring boot application annotation, the spring boot application annotation is the starting point of every spring boot application where all the magic happens.

-> It is a top level annotation, which contains three other important annotations.
Inside it, they are the spring boot configuration enable auto configuration and component scan.

-> The Spring boot configuration annotation tells Spring boot that this class here can have bean definition's, we can define various methods that will expose out beans, which will be later on used by Spring boot for dependency injection wherever required within our application.

-> That is what this Spring boot configuration annotation tells spring boot next enable.
Auto configuration is a very important annotation, which tellss Spring boot to enable and use auto configuration based on the dependencies we have on the class path.

-> For example, here in the Pom Dot XML, if we have a spring boot starter WEB Spring boot will automatically create a dispatcher Servlet bean and configure it for us to handle the incoming requests.

-> Similarly, if we have a springboot starter JPA MySQL Connector, it will create a data source for us automatically by looking at the application dot properties and using the properties inside the application dot properties.

-> That is what this annotation tells spring boot to automatically create and configure those beans that
are required for the application based on what we have on the class path

-> Last and very important, the components scan annotation tells Spring boot that it should scan through
all the sub packages of the current package.

-> Wherever this class lives, which is marked with the components scan annotation indirectly through
the Spring boot application, it should scan that package and all the sub packages for the spring beans
that are marked with @ component at service, at repository, etc. and use them for dependency injection
at runtime.

-> So Spring boot application does all that magically behind the scenes, thanks to the combination of these
three annotations which are inside the Spring boot application annotation.

----

8) What is @SpringBootTest

-> What do you know about the spring boot test annotation. The spring boot test annotation uses a spring
extension class to run our tests instead of using the typical Junit run.

-> It uses a extension class to run our tests.
And this extension class knows how to search for a class within our application that is marked with
@ spring boot application annotation, use that class to create the entire spring context.
for our application that is
Why are all the beans load all the dependencies, do all that magic, and then that extension will start
running our test marked with @ test J unit annotation.

-> So by the time our tests are run, all the spring context is created and the beans are available for
testing.

-> That is the magic of @ spring boot test annotation.

Spring Data JPA and Hibernate

1) What is Spring Data JPA

-> Spring data JPA And why did you use it in your project.

-> Spring data JPA that makes it super easy to create the data access layer
for our application before spring data JPA to create the data access layer while working with ORM
tools like Hibernate.

-> We had to create the DAO interfaces the DAO implementations use entity managers or the Hibernate template,
which in turn used datasource.

-> We had to configure all that to perform crud operations against the database from our application Spring
Data JPA says you no longer have to do all that boilerplate coding you

simply create a JPA entity class and a interface
    that extends one of the repository interfaces from spring data JPA API and they will be able to perform
    all the crud operations and much more on the database.

-> We no longer have to deal with entity manager factory and
    entity manager.

-> They will be taken care of by spring data JPA internally spring data JPA will use them.

-> It is one more layer and it makes our life as a developer super easy to create that data access layer.
    It also has support for finder methods where we don't have to write any queries.

-> And Internally, the queries will be generated against the database table by simply defining abstract
    methods in the interfaces.

-> It has inbuilt support for paging and sorting the results that come back.
    JPQL gives us the object oriented syntax to perform queries and we can also execute native queries where
    required.

-> That is the reason we use spring data JPA. within our projects.

----
2) How to use Spring Data JPA

-> How to use spring data JPA?

-> To use spring data JPA We had the spring data JPA starter dependency to the spring boot application,
    then we had the driver dependency.

-> If it is MYSQL you will add the MYSQL connector ,if it is Oracle you will add the Oracle driver dependency
    to your pom dot XML, you then have to create one class and interface depending on the application.

-> If you have one database table, you will simply define one model, class or entity that will represent that
    database table, you mark that entity with the JPA annotations like entity, ID, etc. and you then create a repository
    interface, not a DAO class, but the interface that extends one of the repository interfaces from spring data
    JPA.

-> And you tell it which entity it has to deal with and what is the type of primary key within that entity
    that set.

-> Once you do that, you will be able to use this repository interface in your controllers or any other
    layer where ever you want to perform the crud operations

-> And if you want to define finder matters, this is how you can define the finder methods without writing
    any sequel and without writing any Java code.

-> You simply define abstract method like this, and it will automatically generate the sequel required to fetch the data for you. And our application is run.

-> Spring boot will look at the dependencies that are there in your project.
    It will automatically create the data source by looking at the database dependency.

-> And the Spring data JPA dependency uses the properties in the application dot properties and it will create
    all the beans that are required and it will dynamically create a implementation for this interface
    using which you can perform various crud operations against your database.
----
3) Create Coupon Service Data Access Layer

-> In this lecture, we'll create a coupon micro service spring boot project and the data access layer for it
    using spring data JPA to do that, go to spring tool suite file menu, new spring starter project.
    Give it a name.Call it coupon service.

-> The group name is Com Dot Bharath Dot Spring boot, the artefact name is the same as the name we typed up top,
    the description is Coupon Micro Service and the package.

-> You can copy the group name from up top paste it right there.
    Click on next.

-> The first dependancy you want to choose is the spring data JPA
    So if you type in data JPA pick that spring data JPA from under SQL, that is the first dependancy next
    type in MySQL and select the MySQL driver dependancy so that we can connect to the MySQL database from
    within our application finish that will create the project.

-> Go to source main java, the first step is to create the model classes are the entities so you can
    right click on the package.

-> New class, the only entity we need is coupon entity that will go into a package called DOT model or
    in some projects, they name this folder as dot entities instead of model finish.

-> The coupon class will represent the database table to become an entity we have id code discount
    and expiry date.

-> So let's define all those fields, private long ID, private string code , a coupon code private
    the discount.

-> I'm going to use big decimal to represent the discount from Java math, big decimal discount.
    And finally, private string exp underscore date

-> To keep things simple, we are using the varchar in the database exp date
    So ID code, discount and expiry date, we have all the fields defined getters and setters for these,
    you can go to source go to generate getters and setters do select all.

-> And I want it after the last field, which is after expiry date generate control shift f to format.
    So far, this is just a plain Java class to make it a JPA entity.

-> You mark this class with @ entity annotation.This is mandatory.
And another annotation that is mandatory on an entity class is at ID.

-> In this case, the ID Field is the primary key.
So you mark it with a ID annotation.

-> And this ID field is a auto increment field if you see here in the database.
So you use another annotation to tell that this is a auto increment field using at generated generated
value, annotation from java x dot persistence.

-> And the strategy that should be used will go in the brackets.
Strategy is equal to generate then type generation type dot identity that will market as auto increment
type.

-> So now this guy is a entity next you simply create a repository interface so right click on the package,
new interface.

-> Give it a name, call it coupon repo and the package instead of Model use repos.
This interface should extend one of the spring data JPA interfaces, this can be the crud repo that guy crud
repository are the JPA repository, which makes it even more easier for us.

-> So extend JPA repository.OK, finish.
And here on this interface, let me Double-Click, to maximize it, we need to specify what type of
entity these repositories dealing with it is the coupon entity.

-> And the primary key field in this coupon entity is of type long, we need to tell it, what type is the
primary key so long is the type thats it so we have that data access layer ready and we can perform
all the crud operations to test it real quick.

-> We have to configure that data source.
So go to source main resources application properties and define the datasource properties, starting
with the data source url so spring Dot, let me zoom it in for you.

-> Spring dot data source dot url is equal to the JDBC url which start with JDBC Colen MYSQL
colon two forward slashes.

-> Local host is where MYSQL server is running three three 06 is the port number slash coupon DB is the
database name next spring dot data source start usernames.

-> spring dot data source.
Dot user name.

-> That guy, not that it's the user name.
Username in my case is the root user, and then the last one is Spring Dot datasource, dot password
is equal to my password is test one, two, three, four.

-> Whatever your root password is, enter that right there.
So that will configure their data source for us.
And the last step is to write a simple test to test it.
Go to source test Java.

-> Open up the test class, that would got generated, so the test class is marked with @ spring boot test annotation
and this @ test annotation is a J unit annotation test.

-> Save coupon test, Save coupon is the method to test it.

-> We need to inject the repository, which is the coupon repo call it Repo Mark this with it @ auto wired
annotation so that that will be injected when the application is run or when the test is run repo dot save
is the method pass it a new coupon entity.

-> Select this new coupon hit control one extract to a local variable call it a Coupon set the field's on
that coupon coupon dot set
We don't have to set the id because it's the auto increment field, set the code within the double quotes

-> I'm going to call it supersale.
Is the coupon code next coupon dot set.

-> The discount is a big decimal.
So you can use new big decimal pass in a integer value, say, a discount of twenty dollars.

-> Finally, coupon dot set the expiry date within the double quotes 12 12 2025.
So that's our test, select the test, right click run as J unit test, as soon as we do this the entire spring
boot application will be launched.

-> All the wiring happens and we should see a record getting inserted into the database.
There we go.

-> So we should have a record in the database.
Go to MySQL Workbench do a select star from coupon.

-> Make sure you are using the coupon db to start with and then select star from coupon and you have
the record in the database.

-> So it's that simple to create the entire data access layer in two simple steps.

-> First, you create the model objects, then you create the repository and you are ready to go.

-> And if you want finder methods, you don't have to write any queries.
In this case, I want a finder method to return back the coupon by code.

-> For that, you go to this coupon repo interface and simply implement a abstract method which returns
a coupon object back find by B is capital y is small, which ever field you want to find by you, will enter that
field name C capital.

-> I want to find a coupon by code and within brackets I pass in the string code that's it.
You don't have to write any sequel's statements or use any annotations.

-> Now you can go to the test and write a new test test.
find by code is the test that may add some lines at the end so that you can see it clearly at test
annotation, Repo dot find by code and pass in the code

-> In this case, the code is super sale
That is what we have in the database.

-> And you can do a assert on this, let's hit control one assign statement to a new local variable.
This is the coupon we are getting back take that coupon and do a assert.
assert equals.

-> Expected is the let's assert on the discount, the discount that comes back should be 20 and here the
coupon dot get discount dot int value, that comparison will do it.

-> Now, select this test, just run only the test, find by a coupon test run as J unit test and see
if the assertion is successful.

-> See that the test has passed, it is green, meaning we have got the coupon back by using the finder
methods thanks to spring data JPAs finder methods, you don't have to write any query
You simply use, find, by and then the field name as it is on the entity.

-> In this case, it is the code.You follow the camel case.
That is why the C is capital here and you get the data back.

---

4) ORM

-> What is an orm. orm stands for object relational mapping, it is the process of mapping a Java class
to a database table and its fields are members to the database table columns.

-> Once we do that mapping, we can synchronized our class objects into database rows
We as object oriented developers will deal with objects instead of writing SQL and invoke methods
like save update, delete on the objects and automatically the sequel statements will be generated
for us.

-> And using JDBC internally, these ORM tools will do the insertion updation deletion, etc. So
we no longer need to deal with SQL directly.

-> And also we no longer have to deal with low level apis like JDBC. we simply do the mapping and
then we start using methods like save and pass in our objects to it.

-> That is the power of ORM.

---

5) Create Product Service Data Access Layer

-> This lecture will create the product micro service and the spring data JPA layer for that data access layer
to do that go to STS file menu new spring starter project call it product service, the group name is com dot bharath dot
Spring boot artifact is product service and the description is product micro.
Service and the package is com dot bharath dot springboot , click on next pick the MySQL driver from up
top and Spring Data JPA as well.

-> Hit finish that will create the product for us or the project for us go to source main java
    Right click start by creating the entity call it product that will go into spring
    Boot dot model package finish.

-> Product has four fields on it, the id the name of the product description and the price, let's add
    the fields for them, private long ID private string, the name of the product.

-> Private string, the description of the product. Private big decimal.
    The price of the product.

-> Go to source generate getters and setters.
    Select all I want it wanted after the last field which is price generate format it save it mark this with @
    entity annotation and mark the id with at id annotation also mark it with generated value, annotation
    generated value.

-> And the strategy we want to use is generation type dot identity. That's the id.
    That's the entity for us.

-> Next, create the repository right. Click new interface.

-> Interface name is Product Repo that will go in to repos package click on ADD Select JPA repository
    that does spring data JPA repositories extent Double click to maximize the generic type.

-> The entity type we are dealing with here is the product which we have just created and the id is of
    type long.We have the repository as well.

-> One last step is to configure the datasource in the application dot properties so you can copy those
    properties from the application dot properties of the coupon service, control a control C, come
    back here, paste them and change at the end of it instead of coupon DB, change the database name
    to product DB.

-> product DB everything else remains the same, the username and the password for the testing, I'm going
    to leave it to you.

-> Go to Source Test java just like how we have tested the coupon service, you need to write a test here
    and save a product into the database by injecting the product repo which you have created injected into this
    test and write a test which will invoke the save method on the repo pass the product details of the
    product intended to do it and see if it gets there in the database.
----

6) JPA

-> What is JPA?

-> JPA stands for Java Persistence API, and it is a standard from Oracle to perform object relational mapping in Java ee applications.

-> Like any other standard from Oracle, JPA comes with a specification and an API, the specification
is for the JPA vendors or the providers, the API is for us.

-> The developers are the programmers to master it.

-> There are several popular JPA providers like Hibernate, Open JPA, Eclipse Linc, etc, which
implement the JPA API by following the specification, the specification is nothing but a set of
rules written in plain English so that these vendors can implement the API easily and consistently.

-> We as developers will learn one single API, so before JPA came into existence, we as developers had to learn Hibernate
API, Open JPA or Eclipse Link API, depending on which of these ORM tools we are using for our application.

-> But now we learn one single API and all these vendors or providers implement this API and we can switch
from one vendor to another without making any changes to our application.

-> That is the power of JPA.

-> Hibernate is the most popular JPA provider and used in most of the Java ee applications out there. So
the JPA API is a set of classes that we master will master entity manage factory entity manager are two
or a couple of important classes in the JPA API and a lot of annotations like Entity @ table, which we
use to map our Java classes to the database tables and annotations like @ id @ column to map our fields
to the database table columns.

-> There are many other annotations which you will be mastering in the next few sections.
This entity manager factory, an entity manager, are specific to JPA.

-> Spring data will even hide this from us.We need not deal with this entity.
Manager, factory and entity manager.

-> Spring data will make our life a lot more easier as we will see in the next few lectures.
So always remember that JPA is a standard for performing object.

-> Relational mapping in the Java ee world and hibernate open JPA eclipse link are providers or implementers of
this particular JPA, API or specification.
----
7) What are the different Entity Object States

-> This lecture is for your own understanding, although you will not be directly dealing with these objects states,
the entities are the domain class objects that we create, go into three different states internally
when Hibernate manages them at a low level, transient state, persistent state and detached state.

-> In a transient state, the object is not yet associated with the underlying hibernate session.
It was just created and it is not yet persisted to persist.

-> It will be invoking either the save method on that object, on our repository, when we invoke save,
it will go into a persistent state, or if that record is already there in the database, we invoke
a find method and that record will be fetched and converted into an object which will be in a persistent
state as well.

-> And when we invoke the delete operation, the object will go from persistent state to transient state
again.

-> So it's not associated with the Hibernate session anymore.
And that record is gone from the database.

-> But the object exists in our application and it is in a transient state.

-> Last but not the least the dethatched state, this is where the object was previously in a persistent
state, but if we invoke the session or entity manager, the low level api s, which we don't use in
a spring data application or rarely used in a spring data application, and we invoke close and clear
on them, the object will go into a detached state.

-> It still exists in the database, but now it's not associated with the underlying hibernators session
or entity manager in JPA.

-> And again, if we invoke a save method on the detached object using another session or entity manager,
it will return back to the persistent state.

-> The key here is the object being in persistent state.

-> If it is in a detached state or a transient state, that object can be garbage collected and removed
at some point because it's not being used internally in the application.

-> We can also pass the detached object across the application layers.
So always remember that internally there is a transient state that is a persistent state.

-> When an object is associated with the underlying session or entity manager and a detached state to go
into a detached state will have to invoke the underlying low level API like close clear on entity
manager.
----

8) Wha are various JPA Associations

-> In this lecture, we will learn what JPA associations are? or what JPA association mapping is? when
we develop huge enterprise applications, let's say online shopping application, we don't just put all
the data into one single database table like order.

-> We will usually normalize our database tables in to separate tables to hold the appropriate data like
order, customer product address.

-> And each of these tables will have relationships with other tables, usually

through primary keys and foreign
    keys.

-> That is where association mapping comes in to map this database tables to our domain classes or entities
    the JPA entities along with their associations.

-> We need JPA association mapping using that.

-> Once the JPA entities are marked with these associations, we can navigate from one entity to another.
    Behind the scenes will have all the sequel queries Joins etc generated by the Hibernate or any other
    Jpa implementation.

-> JPA supports four types of relationships or associations, starting with one to one, the relationship
    between a person and a license is one to one.

-> A person has only one license and a license belongs to a particular person that is a one to one relationship.
    The second type of relationship JPA supports is many to many.

-> An order can have multiple products and a product can belong to multiple orders as well.
    That is in many to many.

-> The last two are one to many and many to one, a customer can have multiple phone numbers, but a phone
    number always belongs to a particular customer.

-> So from the customer's perspective, it is a one to many relationship.
    But from the phone number perspective, it is it.

-> Many to one relationship, it's the reverse of it.
    And also, these relationships can occur in two modes, we can configure them in two modes,
    uni directional or bidirectional.

-> uni directional meaning only we'll be able to access that relationship in one order.

-> For example, in this customer case, if we configure one to many, only from customer to phone number
    will not be able to access the customer using the phone number entity.
    But we can access the phone number from the customer.
    That is unidirectional.

-> We can only configure it on either side of the relationship.
    But if it is bidirectional, that means we can access either of the objects or entities from either
    of them, we can navigate from customer to the phone number and phone number to the customer that is
    bidirectional.

-> So we can do all that association unidirectional bidirectional configuration using the four annotations
    provided by Jpa and several of its properties, starting with at one to one.

-> At one to many, at many to one and at many
    To many will be using all these annotations and also various properties on them in the next few lectures.

----

9) What is Cascading

-> Cascading is the process of propagating the non select operations, like insert update, delete from the main object in the association to the child object, we can control how the propagation happens and at what level or what operations using the cascade element on the many to one one to many annotations.

-> It takes different values, Cascade is equal to persist means an insert operation on the main object should be propagated to the child.

-> Object cascade is equal to merge means an insert or an update operation on the main object should be propagated to the child object remove

-> As the property itself says, if you delete the main object automatically, the child objects should also be deleted.

-> And the next two will lead us to use the underlying entity manager, if you manually refresh.

-> A main object using the underlying entity, manager JPAs, entity manager automatically the child objects will be refreshed.

-> And detach will automatically detach the child objects if you detach the parent object manually using the entity manager, the child objects will be detached.

-> Last but not least, cascade is equal to all will work for all of these when a main object is saved, updated, Removed refreshed or detached the child objects will also be affected.

-> So all this can be applied to an association and depending on what you apply, it will control the cascading effect.

----

10) What is Lazy Loading

-> When objects are associated or related with other objects and when the parent object is Loaded, the child object can be loaded immediately, which is known as eager loading, or they can be fetched on an on demand basis, which is known as lazy loading.

-> For example, here we have a person and person has a association with phone numbers that is a person. Person can have multiple phone numbers and the association is one to many.

-> If we define these phone numbers to be loaded lazily, if we configure it such a way, then when we fetch a person object from the database, only the person object will be loaded.

-> The phone numbers, the list of phone numbers will not be fetched from the database table. But if you invoke the get phone numbers method in the application at a later point in time, that is when all the phone numbers, records will be fetched from the phone number table.

that is lazy loading or on demand fetching.

-> This improves the performance of the application because we need not load everything unless it is
required sometimes in the application, we might only need the person data.

-> We might not even require the phone numbers.
So lazy loading is where the data in the association is fetched on demand when it is first used and
when the parent object is loaded, it will not be loaded right away.

-> So to enable lazy loading in our application on the annotations that we use, like @ one to many,
many, to many, we have an attribute called Fetch and it takes two values.
fetch type is an enumeration which has two values eager
Which will fetch the data right away as soon as the parent is loaded.

-> The child will also be loaded and lazy, as you have already seen, lazy.
Once you choose this, fetch time it will fetch the data only when the child data is accessed for the
first time in the application.

---

11) What are two levels of caching

-> Hibernates supports caching at two levels, the level one cache is at the hibernate session level and
level two cache is the session factory level, so session, factory and session are the low level hibernate
objects which are internally used by hibernate.

-> And when we didn't have JPA, we didn't have spring data.
We used to directly use these in our applications, but now we no longer use them.

-> But internally, if we use level one cache, which is free, which is always there by default enabled,
it happens at the session level.

-> But if we configure level two caching, which needs additional steps, which we will do, that will be
at the session factory level.

-> A session factory, as the name itself says, is used to create multiple hibernate sessions.
And so if we use level two caching, objects will be cached at the session factory level that is they are shared across
hibernate sessions.

-> So if multiple users are accessing our application and we are using multiple hibernate sessions, the
cached objects will be shared across those user sessions as well.

-> Level one cache comes for free, for example, when client one accesses our application internally, the
hibernates session is used by JPA by spring data.

-> The very first time the data is loaded from the database and it's put into the cache
The next time the client accesses the same data, the session will fetch it from the cache instead of
going against the database.

-> And if another client access your application and if a different hibernates

session is used, this cache
will not be referred to, it will have its own cache, and the next time the client accesses it, this session
will check against that particular cache and not the earlier cache.

-> So that's the difference between level one cache and level two cache, level two cache
That will be a common cache across these sessions because it happens at the session factory level.
So here is how Session factory level cache will look like.

-> A session factory is responsible for creating different hibernates sessions and it will have a common
cache So all the sessions will share this cache so when this client access or application, the session loads excutes

-> select query loads the object and it will store it in the cache and then it will send it back to the client
and then another client access our application, a different session is used.

-> But that session will first check to see if the data is there in the level two cache
That is how we configure it.

-> Once we configure the level two cache, that check will be done.
And if the data is there, then there will be no database queries.
That data will be sent back to the client.

-> So level two cache is very powerful because data here is cached across sessions.
But level two cache needs some additional work and hibernate does not have inbuilt support for it will use caching
providers such as eh cache, which is very popular.

-> Server Cache, Jboss tree cache
Os cache.Tangosal cache, etc., but eh cache is the most popular one and very easy to configure and very
powerful, which will be configuring as level two cache for hibernate in the next few lecture's.

----

12) How to configure Second Level Cache

-> How did you configure a second level caching for your project?

-> The first step is to pick a second level caching library like Hibernate eh Cache or hazal cache
Once we pick that, we add the dependency to the Pom dot xml

-> create a configuration file, for example, in case of eh cache will create a eh cache dot xml
where we mention the temporary location where the cache objects should be stored, the maximum
elements that should be stored in memory time to leave for those objects.

-> And all that information will be specified in this XML file.

-> We then go to the Springboot project's application dot properties and turn on the flag, which is
use second level cache to true.

-> And then we configure the region factory class from Hibernate, which is the Eh cache region factory for eh cache Similarly, hazel cache will have its own class.

-> Then you point to the configuration file.the eh cache dot xml, which is on the class path

-> And after that configuration, you go to the entity class and you, Mark, your entity classes which
you want to cache using the @ cache annotation, and you can specify a caching strategy that you want
to use.

AOP

1) What is AOP

-> In this lecture, we will learn what AOP is , AOP stands for aspect oriented programming.

-> This can be best understood by looking at an enterprise application which is typically divided into
layers UI layer, business logic layer, Data access layer, and many more.

-> All these layers typically have some non-functional requirements like security, profiling, logging,
transaction management, when we are especially using a database or a mq system.

-> Logging to write our errors or information to the log files, profiling to see how our application is
performing and security, you know what it is for.

-> These are called crosscutting concerns because these are required across these layers as well as across
the applications in our enterprise or even another enterprise.

-> Any application we develop will need all this and more.
So that is where aspects come in and aspect oriented programming comes in.

-> In object oriented world, we define a class and object is the key unit that represents the class in
the aspect oriented world and aspect is the key unit.

-> You can think of it as a specialized class that addresses one of these crosscutting concerns.
It could be security, profiling, logging, transaction management, etc. So these aspects can be applied
to our classes and objects at runtime.

-> Doing that will have several advantages.Number one, crosscutting concerns.

-> As I already said, we can address all the non-functional requirements which are common across the enterprise
or even enterprises allowing us to reuse.

-> Once we develop an aspect, we can apply it across classes in our application and across applications.
Quick development, we can focus on our business logic without worrying about the non-functional requirements,
because we already have the aspects and we can apply them or we can create an aspect later on and we
can apply it.

-> Focus on one aspect, this using this, we can specialize a particular developer interested in working
on security aspect, he can develop aspect for that.

-> Another developer can work on logging, transaction management, etc., which will allow specialization.
Finally, a powerful feature is enabling and disabling aspects at runtime, we can enable them and
we can disable them using configuration if we don't need them anymore.

-> There are several popular frameworks in open source Java world to implement aspect oriented programming,
spring and aspectJ are two popular ones.

-> Spring also works together with aspectJ, and it has its own implementation of aspects as well.
----

2) Wha is the AOP Terminology

-> In this lecture, you will learn the four important concepts of aspect oriented programming, they are
number one and aspect two an advise three point cut and finally join points.

-> An aspect is a plain Java class that can contain a number of advisers and point cuts.
This class is where we address a particular crosscutting concern for our application.
This could be security, transaction, management, logging, etc..

-> An aspect is made up of a number of advisers and advisers is a method that addresses a part of the concern.
A join point is a point in the Java program where this advice needs to be applied.
This could be a method a field or a constructor.

-> So once we develop an aspect and a lot of methods in it, we need to apply those methods to the other
classes and their methods in our Java application.

-> That is where join points come in. So these are the points where these advisers should be applied.
This could be a method a field or a constructor.

-> A point cut provides an expression language, it's like a regular expression language.
To match a particular join point that is a field method or a constructor, it uses a syntax to express
Join Points.

-> Let's take a look at all this while using spring and aspect J
When we create an aspect, we mark with @ aspect annotation from aspect J.

-> Within this class, we create a number of advisers to address a particular concern, which in this case
is security for our application.

-> This apply security is a method in which will be writing all the logic to supply security for other
classes and their methods in our application.So this is the advice.

-> We use the point cut annotation to express a join point using this expression, we are telling that

this advice should be applied to all the methods that have a string.
My method in them, in their name.

-> While using spring framework, the Join points could be before method after method after a method, returns
after a method throws an exception and finally around around works like before and after.
It is a combination of before and after.

-> So if we use around, that advice will be applied both before a method gets called and after a method
gets called spring supports method level join points.

-> There are no field or constructed level joined points in spring, so to summarize an aspect is a class.
That has a number of advisers, each aspect addresses a particular concern, like security transaction
management for our application.

-> A join point is where a particular.
Advice should be applied to the other classes and methods in our application and the point cut that provides
expression language to express the join points.

Transaction Management

1) What is a Transaction

-> What is a transaction, a transaction is where everything happens or nothing happens.

-> For example, if we are working on a money transfer application, when the money is deducted from the
first account and before it is credited to the second account, what if the application crashes, the
money is lost?

-> It is neither in the second account, nor in the first account.
That is where we will wrap such code in between a transaction or inside a transaction.

-> And once we do that, unless we commit, no permanent changes will happen.
And if something goes wrong in between, we can always roll back the transaction to the previous state.

-> So that that money will still stay in the first account if something goes wrong.
Transactions also allow us to have Save Points and roll back to that particular save point.

-> If we had multiple things happening in a transaction, multiple steps, then we can always roll back
to the certain step instead of rolling back everything.

----

2) What are transaction ACID Properties

-> What are transaction acid properties acid properties stand for Atomicity consistency, integrity
and durability, and Atomicity is where it is atomic meaning everything should happen or nothing should
happen in case of a bank transfer, the money should be deducted and it

should be credited.

-> Otherwise the whole thing should be rolled back.
That is, atomic or Atomicity consistency is where the data should always be in a consistent state.
That is in the case of this bank.

-> Transfer the sum total of the.
Money in these accounts should be same before the transaction and after the transaction, it shouldn't
be in a corrupted state.

-> Integrity is where one transaction should not affect the transaction when one transaction is happening,
the data within the transaction is invisible to the other transaction until the commit happens
the other transaction should not be able to see what is going on here.

-> Finally, durability.
Once the transaction is committed, that should be permanent, durable.
We cannot roll it back once the commit happens.

---

3) What are Distributed Transactions

-> From this presentation, you will learn what distributed transactions are.

-> A distributor transaction also known as Global transaction or a xa transaction typically spans across different
databases or even resources like messaging brokers.

-> A good example is money transfer between two different banks.
We have to make sure that when money gets deducted from your one bank account, it also gets safely
transferred to the other bank account or database.

-> So each bank has its own database and the transaction spans across databases.
That is a distributed transaction.It could also be within your organization.

-> The distributor transaction need not always be across different organizations.
Within the same company.

-> We might have applications which will be using multiple databases and we have to make sure that all
of that work gets done within one single transaction.

-> A distribution transaction involves a transaction manager and one or more resource managers, the transaction
manager is responsible for communication between our application and all the resource managers and resource
managers know how to get the work done by the underlying resources, like the databases or messaging
brokers.

-> They use a mechanism called Two-Faced Commit, so in case of distributed transactions, the entire transaction
gets done in two phases in phase one, each participating resource manager.

-> Will will be writing all the records to the local record, so the transaction manager makes sure that
all the resource managers have written the data to a temporary location.

-> If there is any error, they will communicate that error or issue to the transaction manager.
    And if it is OK, they will communicate that status to the transaction manager in phase two.

-> The transaction manager now knows if all the resource managers are OK or not.
    So if they're all OK, then it will tell each of them to commit everything they have written to the
    temporary location.

-> If there is an issue, then it will ask them all to roll back to summarize.

-> A distributor transaction make sure that applications using multiple data sources can do it all or nothing.
    So everything should happen within the transaction boundary and they do it using transaction managers
    and resource managers in two phase commit.
----
4) What are the Transaction Isolation Levels

-> What a transaction isolation levels? As the name itself says, and we have multiple transactions
    within an application which ran against the database, we need to isolate the underlying data for each
    transaction.

-> That is what we control using the different transaction isolation levels.
    So they give us transactional concurrency to run multiple transactions in parallel without affecting
    each other.

-> To understand the transaction isolation levels, you need to understand three database anomalies or
    transaction anomalies that exist, Number One Dirty Reads

-> Let's say transaction A update is updating the price information of a particular product, one two
    three.

-> In the meantime, even before transaction A commits, transaction B comes in and reads some data from
    the database with a query that qualifies what transaction is trying to do in this case, product number
    one, two, three.

-> Now Transaction B will get all the stale data because transaction A hasn't committed the data yet.
    This is called a Dirty Read at some point.

-> If transaction A rolls back the transaction transaction B would have already used the data which will
    leave the customer unhappy.

-> The price was less than thousand before.
    Second anomaly is non repeatable reads, this is where transection A read some data, in this case a
    product with ID one, two, three, transaction B comes in and updates the price two thousand now
    transaction A reads the data again, but it will see a different set of information as transaction B as committed
    the data.

-> So there is a difference between when the transaction A read the data for the first time and when it
reads it again in the same business logic.

-> Finally, Phantom reads, This is where transection A this is very similar to repeatable reads, except
for the data qualifies in the where class

-> In this case, transaction A is reading all the products that are priced more than 2000.
Then Transaction B comes in and inserts a new product with price three thousand, which qualifies for
this where class now transaction A
reads the data again within the same logic for application or a piece of code, and now it will get
a different set and additional product will come back in the results.

-> This is called Phantom Reads
Now that you understand these three isolation, these three anomalies at the database level or transactions,
these anomalies can be avoided by using transaction isolation levels in the Java EE world
there are four transaction isolation levels, transaction transaction read uncommitted transaction
committed repeatable read and serializable.

-> The transaction, read committed, if we use this transaction isolation level, the transaction can read.
Uncommitted data.

-> This is where dirty reads will happen if we use this read, uncommitted level
Dirty reads non-repeatable reads and phantom reads can occur all of the anomalies will occur if we use
the first level.

-> The second one transaction read Committed ensures that our transactions read only the committed data.
So if we use this level, then we will avoid the dirty reads.

-> Still will have that non-repeatable reads problem dirty reads are prevented.
Non-repatable reads and phantom reads can still occur.

-> Then comes the transaction repeatable read.
This is the popular one most used one by default.

-> Many databases have this as a default transaction repeatable.
Read.

-> This is where we can prevent the non-repeatable read problem as well.
But the phantom reads can still occur.

-> Finally, we have serializable, which is the most strict transaction isolation level, and also the
less performing as the transaction isolation level increases.

-> As we switch from Read committed, un committed to read committed then to repeatable Read and serializable.
The performance of our application will go down because as we move closer to serializable, if we use
serializable, that means it's almost like no two transactions can access the same set of data.

-> At the same time, it could be a table level lock.

-> So in this lecture, you have learned the different anomalies, basically dirty reads, non-repatable
reads and phantom reads, and you also learn that these three can be avoided using their different isolation
levels.

-> And serializable is the most strict and less performing isolation level that the three are OK to use
based on the type of application you are developing.

-> Transactional, repeatable read is the most popular and most used transaction isolation level.

----

5) What is Optimistic vs Pessimistic Locking

-> What are optimistic and pessimistic locking mechanisms, both optimistic and pessimistic locking mechanisms,
determine how the transactions in our application access the underlying data in the database.

-> Optimistic locking means a specific record in the database table is open for all the user sessions or
transactions.

-> While using optimistic locking will not be locking any of the database records, instead, every time
the transaction read some data, it will also read the version, number or timestamp.

-> These are two columns that we add to each and every table in our database and every transaction, and
it reads, the data will get these two.

-> And if a other transaction updates the same record, it will update the version number and timestamp
as well.

-> So the first transaction, when it is ready for an update, it will check if that version number and
timestamp that it has initially read is the same in the database.

-> If another transaction has updated it, then it knows that the data is dirty and it has to discard that
transaction or redo the transaction by reading the data again.

-> So an optimistic locking.
We are very optimistic and hopeful that another transaction will not update the data.
So we are not going to lock the database records.

-> This is very helpful when our application is dealing with a lot of Reads and a few updates in pessimistic
locking.

-> This is where our application will explicitly lock the records or record or even the entire table for
read, right.Only for the current transaction.

-> The other transactions have to wait until the transaction finishes its work because we have exclusive
lock on it, pessimistic locking provides better integrity than optimistic locking, but we have to
design our applications carefully to avoid deadlock situations.

-> In pessimistic locking, appropriate transaction isolation levels need to be set so that the records
can be locked at different levels.

-> The general isolation levels are read un committed, read, committed, repeatable read and finally
serializable isolation.

-> It is not a great idea to read uncommitted data as the uses it uses data from one transaction that's
not completely committed from a different transaction.

-> On the other hand, serializable isolation is used to protect phantom reads.
Phantom reads are not usually problematic, and this isolation level tends to perform.
Very poor, uncommitted are committed.

-> Read committed and repeatable reads are what we use frequently when we do pessimistic locking.

-> I will do a lecture or video on this isolation level, specifically for now, optimistic locking
pessimistic locking or two different ways in which we can control how transactions in our application
access the data in the database and optimistic locking.

-> We have a version number or time stamp or both on each and every database record and every time a transaction
reads the data, it gets them and before it updates the data, it will make sure that they both are
the same.

-> If not, it will discard the transaction.
Pessimistic locking is where we explicitly lock the DB records or the table itself.

-> We have different isolation levels and read committed, and repeatable read are frequently
used isolation levels.

Micro Services

1) What is a Monolithic Application

-> A monolithic application is where we put different modules in a software
into a single application or code base, for example, if we are developing a hospital management software,
which has a patient registration module that captures the patient, details the patient clinical module
that handles the X-rays, blood tests and other diagnostic information bad management model to handle
the inpatient beds and claim management to handle the insurance claims.

-> If we put all these modules into a single application or code base, it becomes a monolithic application.
Or time as this code base grows, it will be very difficult to fix defects or to add new features.

-> If we fix a bug in one of these modules, we'll have to ensure that none of the other modules are impacted.
    At the same time, if we want to add a new feature or fix that bug, do a build and do a deployment
    we will have to bring down the entire application affecting all the modules as well.
----
2) What are Microservices

-> What are micro services, micro services are small and focused applications that bring together pieces
    that change for the same reason.

-> For example, we can split the monolithic hospital management software into four different micro services.

-> The patient registration micro service that gathers the patient details the clinical micro service that
    handles their diagnostic information like X-rays, the bed management, micro service that takes care
    of patient beds, and the claim management micro service that does the insurance claims.

-> The code boundaries here are defined by the business, boundaries are the business problem these
    micro services solve, that is, instead of having a huge code base for one single application,
    these micro services will have tiny little code bases for the problem they are solving and they are
    autonomous, meaning they can be built and deployed on their own without impacting another micro service.

-> And if they have to communicate with each other to get the job done, they will do it using network
    calls through the APIs that the other micro services expose.

-> So if you have to define a micro service or if you have to check a particular application is micro service,
    you simply see if that application can be changed and deployed without changing or impacting another
    service.
----
3) Why Microservices

-> Why use micro services architecture? Micro service have several cool characteristics, starting with
    heterogencity that is we can build different Micro service applications in different programming
    languages, depending on the business functionality and the requirement.

-> And we can deploy them to different operating systems as well.
    This is not possible in case of monolithic applications where we typically right the entire application in
    one language.

-> These micro services then can communicate with each other using restful apis messaging, etc., micro
    services are robust.

-> That is, even if one of the micro services goes down, the other micro services will continue delivering what they should deliver.

-> They are stateless, which enables easy scalability on Micro service applications, will not maintain
state so they can be easily scaled when the load on the micro services increase, we can
have different instances of the same Micro service deployed, which can handle the load.

-> Micro services are very easy to deploy, unlike monolithic applications where we have to deploy the
entire application and all the modules in case are micro services, a change or a new feature can be
easily deployed to production because we only have to deal with that particular micro service and all
the other micro services will not be affected.

-> This increases that time to production, which is very key in applications today.
Last reusable and replaceable.

-> We can use the same micro service wherever required.
And since these are loosely Coupled, we can easily replace one micro service with another micro service
if required some day.

---

4) REST vs Messaging

-> Should we use rest or messaging for communication between micro services, we can use both of them,
but rest is usually good for Synchronous request response scenarios because it uses HTTP protocol,
which is easy to implement, request response scenarios, and also to develop external facing API,
which can be exposed outside of the organization.

-> On the other hand, messaging can be used for asynchronous communication, for non blocking type applications.
It is much more reliable because the messages can be persisted or stored and they will not be lost.
And usually messaging is used for applications within the organization or for the micro services that
communicate within the organization.

REST API

1) What is REST

-> What do you know about Rest?

-> Rest, which stands for representational state transfer, is a bunch of principles are architectural
guidelines, http is the easy way to implement these guidelines.

-> Every application out there performs crud operations, create, read, update and delete.
Which brings us to the first principle of rest, which is uniform interface and easy access using the
http methods post, get put, delete, etc..

-> We can perform all the crud operations through a uniform interface.
    Once we know these four methods, we can perform all the crud operations against any application out
    there.

-> We call these verbs because they do the work for us.
    Similarly, for the easy access principle of rest, http provides us uri s which can easily
    identify any resource.

-> That is the reason they are called Noun's.

-> For example, to create a resource, we leaves the HTTP post Method and the url to
    create employees, we leave employees slash employees url, and then the employee will be created and will get
    a http 201 response back along with the employee data.

-> Similarly, if you want to read a single employee back, you will use a http get method with the
    url which has the employee, ID , and they will get the employee details back with a http
    Response 200 for update you will use the http put method passing the details you want to update and you will get the update
    details back, along with the HTTP response.

-> For a partial update, we use http patch method and passing only that data we want to update and
    we get the entire object that got updated back.

-> Last but not least for delete, we have http delete operation.
    Specify that id of the employee you want to delete and employee is gone.

-> So as a developer, once we master the http methods and the urls , we can do anything with
    any application.

-> That's the beauty of rest and https implementation of rest.

-> Rest should also support multiple formats, that's another key principle it should support xml to
    support Json.

-> It can support text or any other format our applications
    Once they say that they are restful, Web services are restful apis, they should be able to support multiple
    formats.

-> To summarize, the rest is a combination of principles are architectural guidelines mainly having a
    single interface?

-> HTTP is a great way of implementing rest because it provides us the http methods.
    Are verbs next easy to access.

-> We can use the urls in HTTP to easily access the apis that are exposed out
    And finally, it should support multiple representations that is different data types should be
    supported.

----

2) HTTP PUT vs POST and PATCH

-> While creating restful APIs, when do you use the HTTP post and when do you use HTTP put?

-> Although there is nothing that stops us from using http put to create a resource in most cases,

-> we'll be using HTTP post for creating a resource and HTTP
put for updation of resource.OK, that's great.

-> What about put versus patch?

-> We use put method to update a complete object, whereas Patch is used to do partial updates.
If you want to only update one or two fields on the object, then we use the HTTP patch method.
----

3) How did you create REST API

-> Can you briefly walk us through the steps that you follow to create a restful api using spring boot
the first step is to create a spring boot project.

-> While doing that, we add the dependency, starting with the spring boot starter web dependency, which
has all the restful api s and annotations that are required.

-> And then depending on what we are using for the data access layer, we add the appropriate spring data
JPA or spring JDBC and driver jar dependencies.

-> You then create the data access layer using spring data, JPA etc. depending on what you are using,
your code will vary for the data access layer.

-> But the key for the restful layer is to create a controller class mark it with @ rest controller annotation.
Map it to a particular uri, using the request mapping annotation at a class level, then we have
annotations that we use as we create methods within this restful controller.

-> So bind these Java methods to the http methods using the appropriate annotation.
If we want to bind it to post http methods, we use the post mapping if we want to bind it,
to get use, to get mapping.

-> Similarly, we have annotations for HTTP, delete, put and so on.

-> And here you are all you also provide a uri to which they should be binded to. when the client
uses a HTTP post method.

-> With this url This method will be invoked by spring web if you want to pass any path variables.
This is how we do it.

-> We will have a placeholder in the URI mapping when we do it within angular brackets and we use @ path variable
annotation provided a placeholder name which we have used up top, and that variable will be taken out
of the url.

-> That value will be injected into this parameter and we can use that inside

the method.
    So it's that simple.

-> Create application with the right dependencies, mark the controller class with the rest controller, then
    request mapping to you that controller a path url path and create as many number of API methods you want.

-> Bind them to the HTTP methods using the appropriate annotations and path
----

4) Create Coupon Service REST API

-> In this lecture will expose the restful API from the coupon service that will allow other applications
    or clients to create coupons and also retrieve the coupon, given a coupon code,to do that, go to the
    Pom Dot xml of the coupon service.

-> Start by adding a new spring starter dependency, you can copy the spring data JPA. dependency paste it
    above it and change it from data hyphen JPA to Web.

-> So it is spring boot starter to web this guy has everything we need to write our restful controllers go to one
    of the packages.

-> Right click create a new class call it Coupon Rest Controller and this will go into package called controllers.
    Finish.

-> The first step to make this a rest full controller is to market with at rest controller annotation,
    @ rest controller annotation, and we can map it to a particular URI.

-> So request mapping is the annotation that we will use.

-> And within the double quotes forward slash API is the URI. I want to be map to this particular controller.
    This is going to have two methods.

-> The first method will save the coupon into the database and it will return the coupon details back.
    to public coupon.

-> This will save the coupon, I will call it create coupon, create coupon, it receives a coupon object
    and it will return back coupon object hit control one import coupon because it is in the model package,
    hit control one add a written statement that will return the same coupon for now.
    Will change that in a second.

-> Next, mark this with @ Post mapping annotation at post mapping because http post is the method that
    will be used to create coupon in the database.

-> We are going to add a coupon to the existing collection of coupons within the double quotes slash coupons.
    Is the url

-> Within this method, we need access to the coupon repository, so auto wire that coupon repo is the interface call it repo market with at auto wired annotation.

-> Inside the Method you use Repo dot save passing the entity that came in the request coupon.
And we can simply return back what the save method returns, the save method will always return the
coupon that is saved to the database, which will have the ID on it, the auto incremented field.
So we will return back the coupon that just got created in the database.

-> And this coupon, when we receive it in the request, we need to mark it with @ request body annotation
at request body annotation so that it will be serialized or
de serialized that the Json that comes in will be de serialized into this coupon object and the next method is
to find the coupon by code public it returns coupon as well.

-> Get coupon by code is the method name.
This guy receives a string coupon code and he should return a coupon back.
So Repo dot find by code we have that method, which we have written earlier.
That is what should be returned back.

-> Whatever that matter returns that will go back mark this method with @ get mapping.
So http get is what we use to retrieve coupons so we use http to get mapping annotation within
the double quotes that url

-> All that you want to use is slash coupons, slash.
This is the placeholder.

-> So it goes in angular brackets code whatever code comes in, in the url that will be automatically
put into this.

-> And you use an annotation call at path variable at path variable within the double quotes
You specify the name you have used here.

-> That is how we bind it.
So when we pass in coupon code like Super sale Spring.
Will take it, Spring boot will take it, and it will automatically inject it into this parameter, we'll
use it on our method switch the coupon and return it back.
It's that simple.

-> Let's add a context and a port Number to do that, go to the application dot properties.
And here, if you type in context, hyphen path, use that server dot servlet dot context path
That is how we define a context path for our application.
Coupon service is the context path I want.

-> And then the Server Dot port number is equal to I want to use nine zero nine zero for the coupon service
and for the product service.

-> We lose nine zero nine one later on. Run this application.
Right Click Run as Spring boot app. Save everything.

-> That will launch the application on Port nine zero nine zero.
Yeah, I have one problem, which is in the application dot properties, which says the context should
always start with a forward slash go to the application dot properties right

here where I define the context
    path there should be a forward slash. That's it.

-> Run the application again, go to the play button, start the coupon service.
    And that should launch the application on port nine zero nine zero, and we should be able to retrieve
    the coupons that are there already in the database or we should be able to create new coupons.Go to Posman.
    That will be our testing tool.

-> Open up a new tab, localhost colan 9090 is the port number slash coupon service slash API because we have
    mapped our coupon rest controller to slash API, slash coupons, slash code slash coupons, slash the
    code I have in the database is super sale.That is the coupon I have in the database.

-> Let's try to retrieve it, localhost 9090 coupon service ,I have to assess there it should be coupon service.
    That is why I get a 404 for send it again. This time I get the coupon back.

-> Let's try to create a coupon, copy this data, open up a new tab, change the method to post.
    Go to Body select raw.

-> Change this type from text to Json, that is the input i want to send paste the data I have copied.
    Let me scroll this down.

-> I want this coupon code to be mega sale, take out the ID field, and I want to give a fifty dollar
    discount and the expiry date.

-> I'm going to leave it as it is.Go grab that URL from the previous window, come back, paste it the post url these slash coupons.
    We are adding a coupon to existing collection of coupons.
    That is how this url should be used.Hit send.

-> That creates a new coupon in the database with ID two and we get the details back.
    So both our rest ful end Points are working as expected in lectures later on.

-> This restful end point will be hit by the product Micro service to fetch the discount.
----
5) Create Product Service REST API

-> In this lecture, you will create the restful API for the product service that will allow the client
    applications to create a product to do that, first go to the Pom dot xml, go to the spring
    data JPA dependency, grab it, copy it, paste it right above change it from starter data JPA
    to web.

-> Do a control shift f to format. Save it right
    Click on one of the packages, create a new controller class, call it product rest controller that
    will go into a package. Spring boot dot Controllers finish.

-> Mark this with @, rest, controller annotation, next, also mark it with @

request mapping annotation
    to map it to a URI slash API.Is the url what I want to use?

-> This guy will have only one method that will return back a product once it is created so create product,
    is the method name

-> That takes a product model, object call it product and it will also return back a product hit control one
    add a written statement.

-> Mark this with @ request body so that that will be de serialized.
    When the request comes in Mark this method with @ post mapping, because we are creating a product
    or adding a product to existing collection of products, or if there are no products, we'll start from
    the very first product.

-> So we create a product within this method, so we have to inject the repository at this point, product
    repo is the interface that we have created earlier call it repo mark it with at auto wired annotation,
    return, repo, dot, save and pass in the product that came in the request.

-> No surprises so far.It's very simple.
    But before the save happens, we want this restful controller to make a call into the coupon service
    and apply a discount on this product product's price before it is save it to

-> The database that is where the rest template from Spring Web will come in and we will use that to make
    restful calls to the coupon service in the next lecture.
----

6) Use RestTemplate

-> This lecture will make a restful call using rest template from the product service to the coupon service.

-> To do that, inject a rest template.
    Call it a rest template mark it with at auto wired annotation.

-> We need to expose a bean of this type automatically it will not be created for us.
    So you go to the application class that would have got generated.

-> When you created the project and here you will define a new method, Java based configuration, public
    rest template is what we want to return back.
    Call it anything. Call it rest template.

-> And within this, you return a new rest template, object out and mark this with it @ bean annotations
    so that spring will create a bean and use it to auto inject it.

-> Our auto wired into this right here into this product rest controller.
    Now you use this rest template, make a restful call to the coupon service.

-> So use rest template dot.It has several methods get for.
    Object get for entity post put and so on.

-> So depending on the http method that you want to use, there is appropriate method in here.
    Right now we want to use a http get to make a http get call.

-> So get for object, pass in the url in the double quotes and then the response type.
So we are expecting a coupon to come back.

-> But this product service does not have a coupon class to handle the response that comes back.
So go to the coupon service, go to the model, copy this dto right now.

-> This is a model object coupon Dot java, come back.
Create a package here.

-> Right Click on the source main Java End product service.
Create a new package.

-> Call it com Dot bharath dot Springboot DOT DTO, a data transfer object.
This will be a simple Pojoaque paste the coupon on it.

-> Open the coupon dot Java double click to maximize get rid of all this JPA annotations, it is not a model
object, it is simply a pojo that we are going to use to handle the response so you can get rid of
all these jpa imports. Save it now.

-> Go back to the product rest controller here.
The response type is coupon dot class.

-> Hit control one on this line assign a statement to a new local variable and name it coupon.
So when we hit the restful url, which we are yet to configure, we'll get a response of type coupon.

-> This call is just like what you have done earlier from Posman, this guy with the coupon in the url now.
We need access to this url so copy this url go to your method.

-> We don't want to hard code this url right here.
So we will configure it in the application dot properties.

-> Go to source main resources application DOT properties.

-> Let me zoom that for you will call this coupon service url
So coupon service dot url or coupon service.url however you want to name it.
Or let's call it coupon service, dot url
is equal to paste that url the coupon code itself will not be hardcoded.

-> That will come in in the request when the request comes in to create a product, that coupon code will
be sent in so add HTTP at the beginning of it, HTTP two forward slashes localhost 9090 coupon service
Api coupons.

-> Now go back to the product rest controller.
Inject that value the constant that we want to use.

-> Call it string coupon service url mark it with @ value annotation from spring within brackets
within the double quotes use the angular brackets outside of angular brackets a dollar symbol so that it
will pick it from.

-> The application dot properties and injected go to application dot properties, grab the name of the
property you how configured come back paste it within those braces and that

will be injected for us you
    use that coupon service, url right here.

-> Plus, we need the code.So the code information we apply that discount for this product will come in when this product request
    comes in.

-> If you go to this entity right now, this product entity does not have a coupon code, but we'll add
    it private string coupon code.

-> This field doesn't exist in the database and we don't want to save the coupon code in the database.
    We only need it in the restful controller when the request comes in to apply that discount.
    So mark this with @ transient annotation.

-> Any time you don't want a particular entities field to be mapped to the database, you mark it with @
    transient annotation from javax dot persistence, hit control one, create getters and setters for it.
    OK, save it.

-> Now go back to the controller.
    Here we will append plus to this url product, dot get coupon code once we have the coupon code it just
    like this, url see here, 9090 at the end of it.

-> It Will have a coupon code that needs to be sent to the coupon service.
    We get the entire coupon object back before it is saved.
    The product is saved to the database.

-> You say product dot set the price of the product to product dot get the current price dot subtract.
    The big decimal has a method called subtract. Subtract what, subtract coupon dot get discount.
    So we are subtracting the discount.

-> That this coupon applies to a particular product from the current price of the product and then saving
    that product to the database.
----
7) Test End To End

-> This lecture will test the product rest API before we go ahead and do that, go to the application dot
    properties.

-> I have a small Spello here.It should be coupon service.
    Dot url once i change that I come back to the controller here as well in the value and we do the injection.

-> It should be coupon service dot url next you configure a context path for the product web application
    type in context.

-> Hyphen path is equal to forward slash product service is
    The context path. Next server Dot port is equal to nine zero nine one is the port
    Number I want to use for the product service.

-> Let's try to bring our application up. Right click Run as Spring boot application.

Looks like it's happy now go to Posman, go to the post request, which you have sent earlier, to create
a coupon copy that come to a new tab method is post, go to the body of it, select RAW, change the
type from text to Json paste what you have copied.

-> Go grab the post url from earlier.
Come back paste it here, change the port to 9091.

-> Change Coupon Service to product service.
API is fine.

-> We are adding a product to existing collection of product slash products and the information we need
to send in is the product name say MacBook Pro.

-> The price of the product and the description of the product, all that needs to go in, let's go check
the model object quickly to pass in the right fields in go to model, or you can go check the database
as well.

-> We have four fields out of which we need to pass in name, description, price and the coupon code name
description.

-> Within double quotes, it's cool price is a numeric values say two thousand dollars.
comma, the last one is the coupon code, colan coupon code I want to use is super sale that I have
in that database when we pass this the product.

-> Service should use this coupon code, make a restful call to the coupon service that is already up
and running on port 9090

-> Apply that discount to these two thousand and then save it.
This product information in the database.

-> Lets see if all that happens by hitting send.
Let me make sure the url is right localhost 9091 they have a spelling mistake, Product it
should be product service, slash api slash products send there we go, see that discount of twenty dollars
applied on it.

-> So it has saved the MacBook Pro to the database.
You can go check your database, go to the product database, use product DB, execute that to a select
star from product, execute that and you see a new product, MacBook Pro its cool one nine eight zero
is the price all the have and in the price as 2000, the discount has been applied because the super
sale has a discount of twenty dollars.

-> So you can play around with other coupons as well.
Try to create a product, apply a different coupon and that price should be or the discount should be
discounted from the actual price.

----

8) What are Spring Boot Profiles

-> How did you manage the application configuration information like the JDBC

urls the username password,
    the restful urls and so on for your application?

-> For that, we have used spring boot profiles that will define the configuration information for each environment
    as it changes from dev to testing to stage to prod, will create a separate application DOT properties
    file and we'll name them as follows.

-> Application Hyphen Dev for Development Application hyphen Test for testing environment application. Application
    hyphen prod for properties.

-> These test prod Dev that we use in the file names are nothing but profiles.
    To activate a particular profile.

-> We use the following property in the main application dot property Springboot Profile dot Active.
    Once you specify whether it is Dev Test or Prod, automatically that configuration will be picked
    up and used in that environment.

-> Another recommended way is to use the JVM argument as follows.

-> When the Spring boot JAR file is launched, during the Java command we pass an hyphen d spring dot
    profile dot active and the profile that should be activated and used and automatically Spring boot
    will use the appropriate file for configuration.

-> So Spring boot profiles are the answer for our configuration across environments as the urls
    user names etc change.

SOAP Webservices

1) What is SOAP

-> What is soap? soap is a Web services standard, which stands for Simple Object Access Protocol.

-> It allows two applications or multiple applications which are built in different programming languages,
    running on different operating systems to communicate with each other using the http protocol and xml.

-> It comes with some predefined XML elements that we will use to exchange messages across these applications.

---

2) What are the Java EE Web Service Standards

-> What are the Web service standards in Java?

-> We have two service standards, one for the soap based Web services and one for the rest based Web services
    for the soap web services.

-> The Java ee standard is Jax W.S., which stands for Java API, for xml based Web services for
    the rest Web services.It is the JAX RS, which stands for Java API, for xml, based rest services.

-> Although it says XML, it supports various data formats in rest, like any other Java EE specification
or standard. It comes with that specification and that API.

-> The specification is for the Web service providers or frameworks like Apache, cxf , Apache axis in
case of soap and for the RESTful World it is Apache cxf again.

-> And then we have frameworks like Jersey.

-> These frameworks implement these web service standards and this API is for us, the developers.
Once we learn the API, we can use any of these frameworks. Our code and our application will not change.
The same API will work across these frameworks that implement these standards.

----

3) What are the Two Types of SOAP Design

-> What are the two different ways in which we can implement soap based Web services, we can use the Wisden
first or top down approach where we create the wisden file first and then generate the Stubs and implement
the web services endpoint.

-> Or we can go the Java first route, which is also known as bottom up Web services, where we create the
Java classes first, mark them with the appropriate Jax Ws
annotations, and then we can generate the wisdel file from that class.

-> We create our interfaces, we create that is called the Java Plus.
So we have two approaches of wisdel first and Java first.

----

4) What is WSDL

-> What is a wsdl file? wsdl stands for Web services, description, language, it is an xml based
file which acts as a contract between the social services provider and the Web services client.

-> Once we create a wsdl file, we can use it to generate the Java or any language tubs and implement
the soap web service on the provider side.

-> And also the client can use this wsdl file to generate the stubs and to create a client code to invoke
those Web services as well.

----

4) What is the WSDL Structure

-> Can you briefly explain the structure of a wsdl file, the root element of a wsdl file is the
wsdl definition's element within which we have the abstract section and the physical section.

-> The first element of the abstract section is the wsdl tyes.

-> This is where we define all the schema types that we will use later on to define the messages, the input and output messages to our Web services, operations or methods.

-> Once we have those messages, which we create using the wsdl message element, we use those messages
to create the operations. These are like the methods and these methods receive inputs and send out outputs.

-> So we'll be using messages in here, all the Web services operations for a particular Web service end
point will go inside the port type wsdl element.

-> Next is the wsdl binding.

-> This is where we specify what type of binding we want to use on the request and response message types.
Then we move on to the physical side of things using the wsdl service and wsdl port we defined on
which url this website which can be accessed.

-> The connection between the abstract portion and the physical portion of the wsdl is this binding element
on the service when we define the port.

-> We use this binding which binds it to this binding up top here to the abstract section.

---

5) What is the Top Down approach

-> What are the steps to implement a top down or contract first or wsdl first Web service?
We start by creating a wsdl file based on the business requirements.

-> Once we have a wsdl file with all the operations, the bindings and all that, we will generate the
stubs, the Java stubs from this wsdl file using the code gen plugin in case of a framework like Apache
cxf, it has a cxf code gen plug in.

-> When we point this plug in to the wsdl file, it will generate the Java stubs for us once we have
the generated Java stubs will create a class that implements one of these Stubs.

-> And will Over ride the method inside it that receives the soap request and returns a response back and we mark
these classes with the appropriate annotations, like @ Web series, annotation from Jacques's W.S.
java ee Standard.

---

6) What is the Bottom Up Design

-> How do you implement a Java first or Bottom-Up Web service?

-> To implement a Java first Web service, we start by creating an interface where we will define all the
methods we want on that particular Web service will mark this interface with the @ Web service annotation
from Jacks W.S. and then we mark these methods with @ web method annotation.

-> The parameters of this method will be marked with @ Perram and the result will be marked with @ web result from Jack's W.S..

-> Once we have this interface, we implement this interface by creating a implementation class and we
implement all the methods in that interface at runtime frameworks like Apache, cxf will scan through
these classes, look at these annotations that we have used and the information we have provided, and
they will generate the wsdl file on the fly, which the clients can use to create client applications.

----

7) What is a SOAP Client

-> How do you create a soap web service client application to create a soap service client application, we
need to have the Wsdl file that the Web service provider uses.

-> Once we have that wsdl file, we copy it to our project, generate the Java stubs from this wsdl
file using plugins like apache cxf code gen plugin.

-> Once we have the generator stubs, it is super simple to invoke the Web service.

-> We create a client class and within the client class we'll use the generated service and port type classes
pointed to the web service url passing the appropriate request and we get the response back.
All the serialization and deserialization will be taken care of by frameworks like Apache CxF.

----

8) What is MTOM

-> How to send attachments or upload and download files by working with social web services, MTOME, which
stands for message transmission and optimization protocol, is the best way to send attachments to enable

-> MTOME while working with Apache cxf we can enable it at the bus level or we can enable it at a port
binding level for a particular web service once we turn on MTOME
It is super simple to use attachments in our implementation classes.

-> For upload, we will use this data handler from java X activation as the input parameter.
This will have the file, the uploaded file data will come into this automatically at runtime.

-> We can read it and we can do whatever we want with that file.
Similarly, if we want to send a file back, you simply return back a data handler and file source.

----

9) SOAP vs REST

-> What are the differences between rest and soap web services and which one should be preferred?

-> Rest Web services are very simple to implement and use thanks to their architectural guidelines,
they perform well when it comes to the messages themselves.

-> The messages are very lightweight.They are stateless.
So the applications can be easily scaled as the load increases.

-> And it supports multiple data formats, not just XML, which is the case.
soap based Web services soap used to dominate the Web services world before rest.

-> But it is slowly becoming legacy because it is a little hard to implement.
Soap Web services compared to rest.

-> It has the message overload where we have to use some predefined soap headers, the soap envelope and
all that when it comes to messaging and it only supports the XML data format.

-> Soap is still being used where we need a very strict wsdl contract where the non-functional requirements
like encryption, decryption at the message level and also reliability of the messages is required right
out of the box, several soap implementations like Apache's, cxf etc. have all these features in built and
that is one use case I see where soap is still being used.

-> So soap is still being used for internal applications within the organization where these requirements
are of high value.

-> Otherwise rest is the preferred way to go.
Most of the public facing APIs today are restful Web services.

Security

1) What are the Components of Spring Security

-> What is the spring security flow and the important components that are involved when the request comes
in from the Web browser or from a restful client to a secure spring boot application?

-> The first component that intercepts this request is the authentication filter.
It is a servlet filter that checks if the request is authenticated already, if the user or the client
is already authenticated.

-> If not, it forwards it to the authentication manager, the authentication manager component, Intern uses
a authentication provider.

-> This is where the authentication logic lose authentication provider.

-> In turn, uses a user detailed service to fetch the user details from a database or LDAP or any other
source.

-> It also uses the password encoder to encode the incoming password because we don't deal with.
plainText passwords in spring boot.

-> If the authentication is successful, the appropriate response will be sent back.
If the authentication fails, authentication provider will hand over the

response back to the authentication
    manager.

-> The authentication manager hands it back to the authentication filter, the authentication filter based
    on the response.

-> If it is successful, it will use the authentication success handler and stores all the user details and
    authentication details in a security context object.

-> If the authentication fails, then the authentication filter will use the authentication failure handler
    to send the appropriate error response back to the client.

----

2) How did you secure your REST APIs

-> What did you secure, your restful micro service applications?

-> We have used Oauth along with JWT to secure our micro service restful applications.

----

3) What is OAuth

-> What is Oauth? Oauth stands for open authentication and authorization standard, it allows one application
    to access the user's data within other applications without the user directly sharing the user id and password
    of these applications with the application that needs access to this data.

----

4) What are the Key Components in OAuth

-> What are the important Oauth architectural components?

-> There are four components in Oauth workflow.
    The first one is the authorization server.
    This is the guy that will authenticate and provide or generate that token.

-> Second is the resource server.
    This is where our secure resources will be stored.
    Our applications live

-> Third is the resource owner, the guy who has the ownership on the resource and who grants permissions
    to those resources.

-> And finally, the client who needs access to those resources on the resource server.

----

5) What is the OAuth Workflow

-> Can you explain the Oauth workflow?

-> There are different workflows possible in Oauth depending on the use case, but a typical workflow
    will start when the user tries to access a client application.

-> And this client application needs access to other resources that this user owns.

-> At this point, the client application will ask the user to share the user id and password with the client
application itself.
Or it can ask the user, the resource owner, to share the user id password with the authorization server.

-> So the authorization server, once it gets the user data and password validates, it sends a token back,
the token is generated and the token is shared with the client application.

-> The client application now can use this token to communicate with the resource server and the resource
server will validate that this token is a valid token by communicating with the authorization server at
that point if the token is valid.

-> The resource will send the data this client application requires.
----

6) What are the OAuth Grant Types

-> What are Oauth grant types, depending on the use case, we are working on the workflow using which the username
and password will be exchanged between the end user or the resource owner and the authorization server
and the client application can differ.

-> Grant type allows us to configure these different workflows.
----

7) What are the Different Grant Types

-> Can you explain the different Oauth grant types, there are four different grant types, starting with
the authorization code.

-> When we configure this grand type, the client application will redirect the end user or the resource
owner to the authorization server he will have to authenticate with authorization server
once that authentication is successful, the authorization server will tell the client app that the authentication
is successful and then the client app will request for a token, gets the token, uses it to communicate
with the resource server to get the required resources.

-> Next is the password grand type when we use this
The workflow is that the client app will directly take the user name and password from the resource
owner or the end user uses them to communicate with the auth server to get a token and uses that token
to communicate with the resource.server third one is the client credentials.
Here there is no end user or the resource owner.

-> This workflow is typically used for communication between micro service applications.
The client app will have a user id and password given by the authorization server.
It will store those details locally and it will use them to get a token when required to communicate
with the resource.

-> Server this is very useful in case of single sign on applications.
Next is the refresh token

-> The Oauth token will expire at some point.
    The Oauth token that is issued when we use any of these grand types, it will expire after some time
    if it is not used or after that time finishes.

-> So we can use if we configure the refresh token on grand type, that client app can refresh and
    get a new token Instead of going through the entire workflow of these grand tapes, it can simply use the old token
    and get that new token from the authorization.
----

8) What is JWT

-> What is JWT in a typical Oauth workflow when the client application sends the Oauth token that was given to
    it by authorisations over to the resource server. Resource sever has to make sure that the token is a valid
    token for that.

-> It will communicate again with the authorization server to check if the token is valid.
    Secondly, the token that the client application sends will not have the user and user roll information
    which are required for the resource server to check the access level for that.

-> Again, the resource server will get all those details from the authorization.
    Server that is where the JWT comes in.

-> JWT stands for Json Web token and this JWT will have all the user and user role details when the token
    comes to the resource server it need not communicate with the authorization server to get those details
    that token will already have them.

-> And secondly, JWT tokens will be signed by the authorization server.
    So the resource server when it receives the token from the client, it can verify that signature using the
    public and private keys and it may not make another call for the validation.

-> If this signature is valid, then the token is valid.
----

9) Hot to configure JWT

-> What are the steps to configure and use the JWT while using Oauth in spring security?

-> The first step is to go to the authorization server project and in the authorization config will start
    using the JWT access token converter to create JWT tokens instead of the regular tokens.

-> The Oauth tokens will use the JWT access token converter and generate the JWT tokens and will store
    them using JWT token store while creating these tokens.

-> There are two ways in which these tokens can be signed.

-> One is the symmetric way where we use a private key or a symmetric key that will be shared with the resource server project as well.

-> So both the authorization server and the resource server will use the same key to check, to sign and to check.
The authorization server will sign the token using this key.

-> The resource server we'll use this key to symmetrically validate that the token is valid and it is signed
by the authorization
Server we can also use asymmetric key where we use the Java key tool to generate a key pair, a private
key and a public key.

-> The private key will be used by the authorization server, will store it in a key file, will put that
under our source and resources of the authorization server.

-> This jks file, which is generated using the Java key tool that comes with JDK installation.
Once we have the key file, it will have both the private and public keys.
We'll share that with the resource server as well.

-> The resource server will use the public key, which is inside this jks file to verify that token when
it receives the token and the authorization server signs the token using the private key, which is
in that dot jks file.

---

10) How to rotate tokens

-> How often did you rotate tokens in your Oauth application depends on the use case.

-> If it is a read right application, like a financial application or a banking application, we will
rotate them every five and 10 minutes.

-> If it is more a read only operations like a LinkedIn feed, we can leave the tokens there even for
an year that we can rotate the tokens once in every six months or once in every year.

---

11) How to use Tokens with Frontends

-> While working with JavaScript front ends that were using restful APIs through Oauth
Where did you store the Oauth token on the frontend

-> We stored them on the server that is hosting the Front end
And in some cases, if we had to send this token to the Web browser, for some reason we have used tls
that is the transport layer security to ensure that there is no unwanted access.

---

12) What is CSRF

-> What is your CSRF, CSRF stands for cross site request forgery.

-> When we are browsing through a website, that website might set some cookies onto the Web browser to
maintain that user session.

-> At the same time, if the user is also browsing through another website, this website can steal the
cookies that were sent by the other website and use those cookies to forge excession.

-> It can send in a request by forging a session and it can steal our data.
The user data that is there on this website on the server
that process is called cross site request forgery.
----

13) How to prevent CSRF

-> How did you prevent cross site request forgery while using spring security for your Web applications?

-> When we use spring security for our Web applications automatically, spring security will prevent CSRF
from happening.

-> There is a CSRF filter that will generate a token and it will include the token in every Web page that
goes back to the client.

-> So when the client tries to submit a form or a request, that CSRF token should come back to the server.
If it doesn't happen, the CSRF filter will reject that request.

-> So even if another hacking website, a forging website tries to steal the browser session and it tries
to submit a form on behalf, just like acting as if it is the original website, that guy will not have
the generated CSRF token, only the Web application, real web pages will have the CSRF token.

-> So that request will be rejected.

-> That is how this CSRF filter in spring security works.
----

14) What is CORS

-> What these cors and how did you use it in your application cors stands for Cross Origin request sharing.
That is when we have our back ends running on one domain, on one server with its own IP address and domain
name and front ends on a different domain, port number, etc. will have issues when this Front end, whether
it is building Angular, react, etc., tries to communicate with this restful API on the back end,
which is running on a different domain. That is where cors comes in.

-> If you do not configure cors for application, you will see cross origin request errors once you configure
cors using the simple annotations in spring like @ cross origin.

-> These applications will be able to communicate with each other seamlessly the front end will be able
to make restful calls and get the responses back.

-> Cors allows us to configure the security at origin level, the HTTP methods that are allowed and the HTTP headers
that are allowed all that can be configured while using cors.

->   Using this annotation, you can specify which application, front end running on which domain can
     communicate with the back end.

->   That is the origin http methods like get put posts which are allowed from this cors origin
     or cors domain application that is also possible.

->   And then at the http headers also, you can specify what headers can be included when that cors
     origin request comes in.

Java Messaging Service

1) What is messaging

-> What is messaging, messaging is the process of exchanging business data across applications or across
    components within a application.

-> We have a sender of the message, we have the receiver of the message, and we have a messaging server
    or Broker, like active MQ, Sonic, MQ WebSphere, MQ, Tibco MQ and many more.

-> It is the responsibility of the messaging server to receive the message and ensure that it is delivered
    to a receiver.
----

2) Why Messaging

-> Why use messaging, messaging offers various advantages, starting with heterogeneity, that is, these
    applications that use messaging can be written in different programming languages and can run on different
    operating systems, and they can communicate with each other through their messaging brokers and the
    virtual channels, like queues and topics they are loosely coupled because they don't have to know about each
    other.

-> They communicate through the messaging server and they are completely loosely coupled, which offers
    us flexibility and agility to replace one micro service application that is using messaging with another
    micro service application without affecting any other micro service applications using it since they
    are loosely coupled, last and very important to reduce bottlenecks through asynchronous communication.

-> That is, the sending micro service or an application need not wait for the response to come back.
    Once it puts a message, it can continue doing its work and whenever there is a response, it can take
    over and use that response.

-> Also, as the message load increases, we can easily scale by increasing the number of receivers or
    consumers.

-> All these advantages make messaging a key part of any architecture, especially a micro services architecture.

----
3) What is JMS

-> What Is JMS? JMS stands for Java Messaging Service.

-> It is a Java EE standard to use messaging in Java applications just like any other Java EE standard.
It comes with a specification and API.

-> The specification is a set of rules written in plain English, and it is for the messaging brokers or
messaging servers like Active MQ, Web Logic, WebSphere, etc..The API is for the developers.

-> Once we learn the JMS API, we don't have to learn each of these messaging brokers apis before JMS
If we want to use any of these message brokers and communicate to them from our applications, we had
to learn. These brokers like ACTIVE MQ WebSphere, their own apis.

-> Now we have one standard we Master the JMS API, a few interfaces, classes and their methods and
will be able to work with any of these messaging servers or brokers from within our application to do
messaging.

----
4) What is the KEY JMS API

-> What are the key interfaces or classes in the JMS 2.0 API and what are the steps to send and receive
messages?

-> The key classes and interfaces are the connection factory the JMS context
And then using the JMS context, we can create a consumer or a producer and start producing and consuming
messages.

-> And the typical steps are to look up for a connection factory, the messaging broker or the messaging
service like WebSphere,Active MQ, etc. They will create the connection factory and they will put it in the
JNTI.

-> Once we have access to the JNTI, we can get that connection.
factory we create a session or A JMS context depending on the version of JMS we are using.
Look up for a destination which is a qr a topic, and then send and receive messages.

-> So we will get the initial context, which is the point to the Jnti, we then look up for the Q or
A topic.We also get the JMS context by getting the connection factory.

-> Once we have the JMS context, you can create producers, consumers and start sending and receiving
messages.

----
5) Two Types of messaging

-> What are the two messaging models or types of messaging, they are point to point and publish subscribe

Point-to-point, also known as P2P, is where the messages are exchanged using a virtual channel called
Q The center of the message is called producer.

-> The receiver is called a receiver or a consumer.

-> Once a consumer consumes the message from the Q, it is gone from the Q and no other consumer will be
able to read it.

-> P2P supports asynchronous fire and forget as well as Synchronoss request and reply messaging as well.
P2P is like sending email.

-> Once the email is received, it's gone.
No, it will be received only by one person or one consumer publish and
subscribe is where the messages are exchanging that using a virtual channel called Topic.
And here the sender of the message is called the producer.
And there are multiple consumers known as subscribers.

-> The messages will be broadcasted to all the subscribers, like a newspaper subscription.

---

6) JMS Transactions

-> How can we use transactions in JMS producers and consumers when we create a JMS context object,
we can enable transactions using JMS context, dot session, underscore transacted.

-> If we pass this parameter, then a transaction will be started as we produce the messages.
All those messages will not be committed to the queue until we issue a JMS context dot commit and they
will not be visible for the consumers to consume them.

-> Only when you issue a JMS context dot commit
then the consumers will be able to consume those messages and they will be visible to them.

-> If we want to roll back, we can use JMS as context.
Dot rolled back.

-> Similarly, on the consumer side as well, when we create the JMS context, we enable the transaction
and we can use JMS commit or JMS roll back to read all the messages.

-> Only once we do a JMS commit all the messages will be gone from the queue.

-> If we do roll back, those messages will still be available on the queue for other consumers or for
the same consumer to consume them later on.

---

7) What is Message Grouping

-> What is message grouping as the messages are received on a queue, they will be consumed by different
consumers, but if we have a scenario where we want multiple messages that are being received to be
consumed by the same consumer, we use the concept of message grouping will group all these messages

by setting a header on these messages and which are messages have the same header value will be consumed
by that same consumer, that JMS provider is to ensure that they are consumed by the same consumer.

-> So when you create the messages, you will set a string property on that message.
Call JMSX group id and provide a value which all messages will have the same JMSX group id
value will be consumed by the same consumer.

----

8) What is a MDB

-> What is a MDB? MDB stands for Message Driven Been, it is a part of the EJB standard and any application
servers like Web logic WebSphere JBoss, which has support for EJB, has support for MDBs MDBs are java
classes

-> They are just like message listeners in JMS.
We create a MDB by implementing a interface and configure it using xml or annotations.

-> Once we do that, whenever there is a message on a particular Queue, automatically these applications
servers like JBoss will trigger the MDB and they will hand over that message to that MDB and it can
handle those messages asynchronously.

Design Patterns

1) What are the Design Patterns you have used

-> What are the different design patterns you have used in your projects, at a minimum, we would have use
the following patterns, the data access object pattern to implement our data access layer.

-> It's simple, Class, where we rate all the data access code, A Singleton pattern to implement utility
classes where only one instance of a particular class should be created, the prototype pattern indirectly
or directly.

-> When use frameworks like spring, we have the option to configure singleton beans or prototype beans
for our application, the dependency injection and IOC patterns when we work with spring and
the dependency injection across the layers.

-> The MBC pattern for our weblayer, especially if we are using spring web, factory pattern
If you are using utility classes and using factories to create the instances of those classes.

-> In addition to this, you can specify as many number of design patterns as you know, and you have to
use it like facade builder, command, etc. but be ready to answer any questions on these patterns you
mentioned.

----
2) What are Singleton Best Practices

-> How did you implement the Singleton pattern, can you talk about some of the best practices you have
followed?

-> There are different ways in which a singleton pattern can be implemented, but at a minimum, we will
make the constructor of that class private so that nobody will be able to create the object directly.

-> Then we define a field of the same class type globally will then implement a static method which will
create an instance, assign it to that field which we have defined globally, and we will return that
back.

-> And when we create that instance, we will define it in a static block and acquire a class level lock
so that no other thread will be able to access this particular block and that particular instance field,
which we have defined globally as a static field.

-> We will then also make sure that nobody will be able to clone this object by implementing the Clonable
interface and overriding the clone method.

-> We will throw an exception,thrown not found exception if somebody tries to clone this object will also prevent
serialisation and deserealisation so that nobody will be able to be deep clown it by implementing the read resolved
method.

-> And when somebody tries to serialize and deserealize they will get back the same instance which we have globally.

Testing and Code Quality

1) What is Unit Testing

-> What is unit testing? when we develop java or Spring boot applications, we organize our code across layers,
the controllers layer the services, layer, integration layer, data access layer and each of these
layer will have classes or components of their own.

-> Testing these classes in complete isolation is nothing but unit testing.
We're going to test all the public methods in each of these classes and we do that.
We will be mocking out the dependencies, these classes
have if the controller class depends on a services class or a repository interface, we are going to
mock it out and test only this method scored in complete isolation.
That is unit testing.
----
2) What is Mocking

-> Can you explain how you how used mocking in your projects, the first step to use mocking is to add
a mocking library like Mockito to our projects class
path once we have it, when we write our unit tests will mark all those

objects that we want to mock
    with @ mock annotation.

-> We then mark the class under test with at inject Mock's annotations so that Mockito will inject all
    these mocked objects into this class under test.

-> We then write that test methods within which we use the when method.

-> To mock the method calls on this mocked object when a particular method on this mock object is invoked,
    then we are going to return the mocked output instead of invoking the actual methods, will be stubbing
    them out or will be mocking those method calls.

-> We then invoke the method under test on the test class and we can verify using the verify method from
    Mockito, we can verify if the expected methods on the mocked object are being called.
    And after that we can have several Junit asserts, as usual, for the expected result.
----
3) What are the various Testing Tools you have used

-> What are the various testing tools you are aware of? here, you can list out the tools you used in your
    projects, typically we use Junit

-> Mockito for mocking power mock if we require to mock private and static methods and also DB unit,
    if we want to mock the database creations database calls and all that.
----
4) What are the important JUnit 5 and Mockito annotations

-> What are some of the important Junit and Mockito annotations, the first important annotation is the @
    test annotation, which we use on every test method, and then we can use the @ before each and before
    all annotations on any methods that we want to use to set up the test data before each any method that
    is marketed before each will be executed, before every test and before all will be executed once for
    the entire test class.

-> Those are some of the important J Unit five annotations.

-> Similarly, we can use after and after all, to execute something after a particular test or after all
    the tests.

-> When it comes to Mockito @ mock annotation is the annotation from Mockito we use to mock a particular
    class or interface.

-> Then wherever we want to inject this mocked objects, we use that @ inject mocks on that class.
    This is the class usually which is under test and will be mocking all the dependencies on it.
    So we'll mark it, @ inject mockito annotations or at inject mocks.
----
5) Code Review Checklist

-> What are some of the code review guidelines you have followed to answer this question, you can download
this code review checklist and add to it as Per your experience.

-> The very first thing we look in the code is to ensure that the code is delivering the functionality
that is expected as per the acceptance criteria or as per the requirements.

-> Then we should also ensure that developers are not reinventing the wheel.
They are using all the components that exist in the project or a open source library instead of writing
a lot of code.

-> If there is an open source library that is approved by our architect, we can use that right away in
the code.Then we ensure that the basics are covered.

-> The object oriented principles like encapsulation, inheritance, where we get to reuse a lot of code
should be followed as required.

-> According to interfaces is another good code review checklist ensuring that all the null pointer exception
checks are done.

-> The exceptions are not being eaten up and that being handled and logged correctly.

-> Then we move on to the clean coding practices where we ensure meaningful and short names for our classes,
methods and variables, short functions which do only one thing.

-> And a function should not take too many parameters if there are too many parameters wrap it inside a object.
We can use code formatting templates across our project so that all the developers use that template
and the code will be formatted according to the standards moving onto the nonfunctional side of things.

-> We ensure that the proper log levels are used and enough information is being returned to the logs are
not too much.

-> Information is going into the logs then security, which is a very important and aspect, ensure that
no sensitive information is being logged to the logs use immutable objects wherever we can.

-> And also ensure authentication, authorization, encryption, digital signatures are used wherever required
to secure our application code, then comes the performance where we have to use the right type of collection
classes and also ensure the thread safety is being done correctly.

-> Using concurrent classes are concurrent collections wherever required instead of synchronizer collections
and all that cool things memory leaks can happen when we have static fields or static variables that are holding on
to object references.

-> We have to ensure that they do not have a long life and they are short lived to avoid memory leaks.
If there are any hard coded values, those should be moved to properties

files and they should be injected.

-> Last but not the least, there should be enough test coverage and tests should be meaningful.

-> They should not be just written for the sakes of test coverage, but they should really test the functionality
and all the educes should also be tested.

GIT

1) Why use GIT

-> What are the advantages of using Git as a version control system?

-> Git is a distributed version control system where the entire project is downloaded to the local machine
and we can work with it offline.

-> It is free and open source, fast and lightweight, because we can work with the local repository on
our machine. It is very secure.

-> It uses secured hashing mechanisms to communicate between the local and remote repositories to push
the changes or to pull the changes from the remote repository and to create branches.

-> Git makes it super simple.

----
2) What is the typical Git Workflow

-> What is a typical GIT workflow that you follow?

-> We typically start either by creating a new remote repository for a project or by cloning an existing
repository of the project if it is already there.

-> When we clone a project, we'll get a local copy of that project on our machine.
We can then make changes add those changes to the staging area commit those changes.

-> And finally, when we are done with all the changes, we push those commits to the remote repository,
which can then be seen by other developers.

-> To pull the changes, we will use a pull command, or a fetch command and fetch all the latest changes that were
done to the remote repository by other developers on the team to our local repository, when we commit
a change in new commit id will be created and the head always points to the latest commit.
----
3) Commands you have used

-> What are some of the Git commands that you use on a regular basis, I use git status to check the status
of my local repository and the files in it, git stage or git add the files

that I have changed
to the staging area, git commit with that commit message, will commit all those changes in the staging
area.

-> And when I'm ready, I use the git push command to push all those commits to the remote repository, git
log will show me all the commits that I have done git

-> If I pass in the committee, it is it will show the differences between two different commits
I also use git reset and revert once in a while to revert the changes in my staging area.
or the changes I have committed.

-> I also use Git Branch to know the branch that I am on and also to switch branches or to use a particular
branch I use git check out and that particular branch like Dev etc..
----

4) Tools you have used

-> Have you used any GUI based online tools to work with GIT, I have mostly use the command line, the GIT command
line to work with Git, but I have also tried the support the id's like spring tool suit intelli J idea
has for git where they provide UI based tools.

-> I also worked with source tree, which is a git client tool, and also the GitHub desktop, which is
super simple to use.

-> So based on what you have used on your projects, you can use them to answer this question.
----

5) Fetch vs Pull

-> What is the difference between a git fetch and a git pull, the git fetch command will only retrieve the
changes, the list of changes that happen in the remote repository, and it will also show the differences
between the local repository and the remote repository.

-> It will not fetch the actual files that were changed from the remote repository.

-> It brings back only the list and the differences between the local and remote, whereas git pull will
pull the actual changes or the files that have changed from the remote repository to the local repository
on our machine.
----

6) reset and revert

-> What is the difference between a git reset and revert? a git reset allows us to reset the head to a particular
commit and all the recent commits after the commit will be gone, they will no longer be available,
where as a git revert will create a new commit by undoing the changes in those commits which we have done,
which we want to revert.

-> So if you have commit four and five we want to revert to three will not lose four and five, it will
create a new commit by undoing the changes in four and five, it goes back to the state of three.

-> But that will be created as it new commit and the head will point to that commit.
So that's the key difference between a reset and a revert.

-> when revert is preferred over reset because we will not lose the changes that were made.
It is like undoing the changes with that new commit.

Build Tools

1) What the different Maven Scopes

-> What are all the maven dependancy scopes that you are aware of?

-> I have used compile scope, meaning the dependancy is required during the compilation of the source
code and test code, and that dependency will also be Bundled if you are creating a var file or er
file for that application.

-> Test scope, which means that the dependancy is required only for the compilation and execution of tests
provided, meaning the dependencies required for compilation and testing, but it shouldn't be included
in the bundled var file or er file, it is going to be provided by application server,
Web container or the environment where we are deploying our application import.

-> Scope makes sense only in the pom dot xml of the parent projects where it is of type Pom and it is
usually used in combination with the dependency management section.

---

2) snapshots vs release

-> What is the difference between Maven snapshot and release versions?

-> When we created a maven project, it will have hyphen snapshot in it, meaning that this project is still under
active development.

-> If another project depends on this Project Snapshot version while it is still in development, then
that project should expect that there will be New Code changes just coming in every day or every couple of
days.

-> Every time a commit happens on this project, even though it uses the same version since it has a
snapshot.There will be new changes that will be added.

-> But at some point this project decides that devops engineer or the architect decides that this project
is ready for release.

-> They will give it a release version.
There will be no Hypen snapshot in the version.

At that point, there will be no more Code changes for this particular version.

-> Any project that depends on a release version will get the same copy, irrespective of how many times they pull that library.

---

3) How to Control Dependencies

-> When we create multiple maven projects within a team or organization, how do we ensure that all these projects are using the same version of dependencies and plugins?

-> One easy way to do it is to create a multi module project with that parent pom dot xml.
Within this pom, the packaging type will be pom are not jar or war will define a dependency management section in the parent pom where will put all the dependencies that are required for our projects and their version information as well.

-> Any project that needs a particular dependency will use the dependency block, but it will not specify the version.

-> The version will be automatically derived from the parent project because in the child project will define what is the parent project for it and the version information for any dependency will be taken from this parent so that we can ensure all the projects are using the versions consistent.
The same for the plugins as well.

-> We can define a plugin using the plugin management section in the parent pom dot XML and the version of it.

-> And in the child projects we simply use the plugin and we don't have to specify the version.

---

4) How to Override a Transitive Dependency Version

-> What is a transdue dependancy in Maven? when we define dependencies for a project, if the dependancy requires another library to do its work, then that dependency will automatically pull for us.
That is nothing but Transidue dependency.

-> Can you override the version of a Transdue dependency?
Yes.

-> To Over ride a version will have a define the Transdue dependency in our Pom dot xml and then specify the exact version of the dependency we want by default transduely
It will pull whatever version is required by that library.

-> But if we want a particular version, we have to explicitly define that trancedu dependency and define the version which our version we want to use as well.

```
1)  Continuous Integration vs Delivery vs Deployment

->  What is the difference between continuous integration, continuous delivery
and continuous deployment,
    continuous integration is where we write several tests which will be run
every time a new commit happens
    to ensure that our application still works as expected.

->  And it stops right there
    continuous delivery extends continuous integration and runs production like
test in a stage environment
    getting us close to production but in continuous delivery will have some
manual steps.

->  This could be performing some manual tests, which could not be automated
approval of a manager or the
    devops architect.Continuous deployment means that are that there are no more
manual steps.

->  As soon as the code is committed, the application gets built.
    Several tests are run across environments, testing environments, staging
environment, etc. and our
    product will go to production.
--------------------------------------------------------------------------------
---------------------------------------------------------
2)  What is Jenkins

->  What is Jenkins and how did you use it in your project, Jenkins is a ci/cd
tool or framework that allows
    us to create ci/cd pipelines that is right from Code Commit.

->  Our application will be deployed all the way to production.
    Jenkins does this by integrating and working with various tools.
    It has plug ins to work with various tools.

->  When our code is committed to repositories like Git Jenkins' will get
notified.
    It will pull the latest code from these repositories, do a build by working
with tools like Maven,
    Gradle, etc. run all the tests, whether that unit tests or integration test
by working with tools
    like Junit, Selenium, so on, and then do production level deployments while
working with deployment
    tools like Excel, Deploy and many more.

->  These deployments can be to the servers running within our organization or
even to the cloud.
--------------------------------------------------------------------------------
--------------------------------------------------------
3)  How to create a Jenkinsfile

->  What do you know about the structure of a Jenkins file?

->  A Jenkins file is a groovy script that starts with the pipeline at the top.

->  And the first element inside the pipeline is the agent on which this build
can be run, and it includes
    multiple build stages like build, test, deploy, etc., each of these stages
will have multiple steps
    in turn to build and deploy our projects.
--------------------------------------------------------------------------------
------------------------------------------------------
4)  What are the Steps to automate a deployment
```

-> Can you briefly explain how you have automated the deployment process using Jenkins for your project?

-> We created Jenkins file, which will have all the build stages for our project.
We then pushed this Jenkins file under the project directly in GitHub, create a Web hook on GitHub
that will push events to the Jenkins job.

-> We create whenever a commit happens on this project, automatically the Jenkins job will be triggered and
the build will be run, which will run through all these stages.

-> It will build, test and deploy our application to wherever we want automatically.

---

5) How to passParams and Inputs to Jenkins Build

-> How can you use the jenkins' environment variables that are available when a build is happening inside
your building script?

-> We can use the ENV object that is available at runtime by using the dollar expression dollar with in flower
brackets env, object, dot, the appropriate environment variable name will give us the environment
variable, for example, the build number.

-> How can you pass parameters to your build scripts dynamically at runtime?
When we run a build, we have the configure option on the Jenkins' dashboard, we have an option where
we can configure params for our build scripts.

-> Once we do that, we can read those parameters using the dollar flower bracket syntax.

-> How can you take inputs dynamically when the build is happening, if you want to request for some inputs?
How would you do that?

-> We can use the input function and provide a string which will ask the deployment manager or a devops engineer
for the inputs, for example, approve deployment.

-> Then it will show a pop up when the build is running that in that particular stage, it will show a pop
up call, proceed or abort.

-> so you will have the options of proceed or abort if you clicks on abort.
Then the build will stop. If we click proceed, it will move on.

Docker

1) What is Containerization

-> What do you know about containerisation, to launch a Java software application
We need to have a physical or virtual machine on top of it.

-> We need an operating system, a JDK application server like Jboss, or Web

logic on top of it.
We deploy our application and run it.This process can be time taking and expensive.
That is where containerisation comes in.

-> Containerisation allows us to create a abstract representation of all these softwares, including our application.

-> We create something called an image, and once we have an image, we can launch any number of containers we want for our application and all the software that it requires in any environment very easily and quickly.

---

2) What is Docker

-> What is Docker? Docker is a containerisation tool or framework that allows us to package ones and run anywhere, it implements the open container initiative standard and once we dockerize our application, we no longer will have our applications as war jars dll in case of Dotnet or django code in case of Python django applications, but instead will have a docker image.

-> This image will have the entire infrastructure information along with the application itself.
Once we create this docker image, we can launch any number of containers on any operating system that has Docker installed on it and our applications will be up and running.

-> We can have one image and launch container which will have all the components that are required for an application to run.

-> Or we can put these components into different images.
We can have db server on a different image web server on a different image or application itself on a different image

-> And we can launch separate containers out of these images, which will then communicate with each other.

---

3) What are the Docker Components and Workflow

-> Can you talk a little about the important docker components and a typical docker workflow?

-> The first component is the Docker repository or a registry where all the docker images are stored centrally.
The second component is a Docker host.
The machine where we install, the docker engine, the third component is the Docker client that gives us the ability to run commands against their docker engine.

-> When we execute a docker pull command, for example, against the docker engine from our command line, the docker engine will first check if that image is available locally.

-> If not, it will pull it from the central docker repository and it will then store it locally.

And when we docker run command, the Docker engine will check.

-> If it exists locally, it will use that image and it will launch a container for us.
We can also create our own docker images for the applications we create and build those images, we can then
push these images to the central docker repository, which can be used by other teams or other developers and testers.

----

4) Why Docker

-> What is the advantage of using Docker for containerisation of your applications, Docker uses the concept
of image layers and overlay.

-> Docker image is made up of multiple layers.
It uses the concept of union filesystem to integrate these layers together.
And we pull the image.

-> It pulls several layers, starting with the base kernel and then it will merge.
All those layers are overlays one layer on top of another.
from the next time we pull the same image, instead of pulling all the layers, it will only pull the
players that have to change it.

-> For example, if our application has changed and if we have a new Jar or OR that was created as a part
of the build, only that layer will be pulled and it will be overlaid on top of the other layers, improving
the performance and reducing the network bandwidth as well.

----

5) What are some of the Docker Commands you have used

-> Can you talk about a few docker commands you have used, Docker images will list out all the images that
we have on our machine, Docker pull or docker Run can be used to launch a container by pulling the container
from the central docker repository.

-> If it's not there on the local. Docker ps will list out all the containers that are running.
Docker start can be used to start a container
using an image Docker stop can be used to stop a container to build our own docker image from the
docker file we use the docker build command to push that image once we created to the central docker hub

-> we can use the docker push command if we want to execute a command inside the container that is running
on our docker engine, then you can use their docker, execute and passing the command you want to execute.

-> Docker Search will search for a particular image on the central docker repository.

-> Docker attach can be used to attach to a detached container that is running. Docker
commit will create an image out of an existing container that is up and running.

-> Docker rm can be used to remove a container docker RMI.

Can we use it to delete a image
    And finally, if you want to prune or delete the images that are not being used at all, you can use
    Docker Image Prune and to delete all the containers that are not being used, you can use Docker Container
    Prune.
----

6) What are Docker Volumes

-> What are docker volumes and bind mounts for? when we delete a docker container, the application data on that container
    will also be deleted or lost.

-> If we want to persist this data onto the host machine on which Docker is running, we can use Bind
    mounts or volumes.

-> Bind mounts, or volumes are just folders on the host machine that will be binded or mounted to a folder
    on the container.

-> So our data will be stored on the host machine instead of the container itself.
    Even if the container is deleted, the data will still live on the host machine.

-> And the next time we launch a new container using the same docker image, that container and the application
    inside it will be able to continue using the data on the host machine through this volume or Bind mount.
    And it can update that folder with data as well.
----

7) Volumes vs Bind Mounts

-> What is the difference between bind mounts and volumes, bind mounts can be any folder on the host machine and they
    are not managed by Docker.

-> Any process on the host machine can have access to these bind mount folders.

-> On the other hand, volumes are managed objects which are managed by Docker they live under a certained folder
    within Docker.

-> And no other process on the host machine will have direct access to these volumes.
----

8) How did you dockerize your application

-> Can you briefly explain how you have Dockerized your spring boot project?

-> We have first created a file called Docker File under our project without any extension, the very first instruction
    within the docker file specifies the base image based on which the rest of the image will be built.

-> This will be Java or open JDK.
    Next, we have added the jar file, the spring boot Fajar to this image.
    This jar file typically will live under the target directory or the building directly, depending on
    whether it is MAVEN Project or Gradle project.

-> We copy the jar file on to the image and the last instruction will be the entry point, which specifies
a command that should be run.

-> As soon as the container is up and running, we use the Java hyphen jar and the jar file in which we
have just copied to the image so that this jar will be run and our application will be up and running
as soon as the container is up.

---

9) What is docker compose

-> What is Docker compose when we have multiple Docker containers that use each other, we can bring all of
them up or take them all down in one shot using Docker compose, for example, if we have two micro services,
product, micro service and the coupon micro service, where the product Micro service uses the coupon
micro service to get its job done, will define those micro service containers as services inside the

-> Docker compose file with all the container information, the image based on which the container should
be launched restart policies, the port that should be exposed, the environment variables and all that.
Instead of defining them from the command line, we define them right in the docker compose file and
these services can use each other.

-> So the product service depends on the coupon service as well as both.
These micro services need another container, which is their Docker MySQL container.
We fill in all those details here.

-> The images is Mysql restart policies, environment for the MySQL and all that.
And then the other two micro services will be able to depend on this particular container.
So when we run a docker compose up using this file, first the Docker MySQL container will be launched,
then the Coupon micro service container will be launched, and finally the product micro service container will
be launched, it uses the other two containers.

-> So Docker compose allows us to launch multiple containers that use each other in one shot.
We can also take them down with a single command.

Kubernetes

1) What is Container Orchestration

-> What is container orchestration, container orchestration is a process of launching a cluster of containers
while providing various non-functional requirements or services like fault tolerance, that is, if
one container goes down, there should be one more or several other containers to back it up on demand
scaling.

-> That is, if the load on one of the micro service containers goes up.

If there are too many requests coming in automatically, the scaling should happen.
More containers should be created and when the load goes down, the scale down also should happen.
The containers should be taken out so that they will not consume too many resources on the cluster.

-> Next is auto discovery or micro service applications running on different containers should be able to automatically
discover each other and use one another public access.

-> The containers within the cluster need to be accessed by other containers within the cluster as well
as outside the cluster auto update and rollback is a super important, non-functional requirement wherein
we can deploy applications without affecting the clients.

-> While the deployment happens, the old container, some of the old containers will be still kept up
and as the new containers come up, the old containers will die out.

-> Roll back, which is the last non-functional requirement in container orchestration is where we can
go back to the previous cluster state if required.

-> If we have any issues with the latest cluster state as we deploy a new feature or new product, we can
roll back to a earlier cluster state where it was stable.
----

2) What is Kubernetes

-> What is Kubernetes? Kubernetes is a container orchestration tool that will take our containerized
applications and quickly create and maintain a cluster out of them.

-> Once we have these clusters, we can use them for our Dev testing and even production environments.
As a part of our CI/CD pipeline, we can set up a kubernetes cluster on our organization servers
or on the cloud.

-> Kubernetes is maintained by cloud native computing, foundation CNCF and also all the popular cloud
environments like Google Cloud, AWS AZURE and many more
have in built support for kubernetes, we can also set up our own kubernetes cluster on these cloud
environments as well.
----

3) What is a Pod

-> What is a Kubernetes pod? Pod is the lowest level object in kubernetes cluster, it logically groups
related containers together to define a pod

-> We use the spec section within the template section in the deployment, Yaml file, and we have a spec
section that is where we specify which all containers are a part of a particular pod.
----

4) What is a ReplicaSet

-> What are replica sets, replica sets manage the pods which specify the number of pods we want through a
replica set.

-> We define a replica set in the deployment dot yaml using the spec section under the spec section.
We have the number of replicas.

-> This specifies how many pods we want in the cluster at any given point in time.

---

5) What is a Deployment

-> What is a deployment? A kubernetes deployment is how we manage pods and replica sets as a single unit,
we define the pods in the template specs section for the pod with all the volumes and everything that
a pod requires.

-> And then we specify the replicas, the number of replicas we want off that pod.
All that is managed in a deployment dot yaml file

---

6) What is a Service

-> What is a kubernetes service object, a service object logically groups a set of pods that need to communicate
with each other and also with the world outside the cluster itself, we create a service object using
the service yaml file where we specify the type of service and the ports that need to be exposed out and all
that.

-> And we bind this service to a particular deployment which will have the pods inside it through the
labels and selectors.

-> Using the labels and selecters, we map a service to a group of pods within a deployment.

---

7) What are different Service Types

-> What are the different service types in kubernetes?

-> The first of the service type is cluster IP. This allows the pods within a cluster to communicate
with each other.

-> It does it by creating virtual IP addresses for these pods, using IP tables or IP before, and they
will map those virtual IP addresses to the actual IP address of the pods as they come up and go down.

-> It also does the load balancing across pods if they host, the same containers.

-> Next, we have Node port, which is built on top of cluster IP, Node port exposes a higher order port
so that applications outside the cluster can communicate with the applications running within the cluster.

-> Then comes the load balancer service type, which we use when we are configuring a
External load balancer like a aws load balancer, azure load balancer, etc. and these load balances can
balance the load across nodes within a kubernetes cluster.

-> Another one is external name, if we want to configure and use an external application, which is outside
the cluster, as if it is in the cluster, then we use this external beam service type next ingress.

-> This is not a service type, but when we want to expose out a lot of parts within the cluster in one
shot, ingress is the way to do.

-> And it allows us to use lower port numbers instead of using the higher port numbers in case of Node
Port.

-> When we go the ingress route, we can use the Lower port numbers or the actual port numbers of the services.

----

8) What are Namespaces

-> What are kubernetes name spaces when we have multiple applications deployed to a kubernetes cluster
We don't want these applications to step on each other's tows and we don't want just one or two applications
using up all the cluster space c.p.u and other resources.
That is where needed spaces come in.

-> And they allow us to logically divide the cluster once we create an inner space assigned to a particular
project or team.

-> All the applications for that project or team should confine to that namespace and can only use the
CPU processes, etc. allocated to that namespace.

----

9) Explain Kubenetes Architecture

-> Can you talk about the components in a kubernetes cluster, the machines in a kubernetes cluster
are referred to as nodes

-> We have Master nodes and we have worker nodes in the Master node
The key component is the API server.

-> This is the component that allows the master node to manage the entire cluster.

-> We communicate with this api server through the api it exposes out by using tools like kube ctl

-> The next important component is the scheduler component.
This is the guy that schedules the required pods in the cluster on the worker nodes.

-> And once the cluster is in place with all the desired pods and containers, the control manager component
will run in the background and it ensures that the cluster always remains in their desired state.

-> That is, if one pod goes down automatically, the control manager will ensure that another pod comes
up.

-> The ETCT component or the Etcd component is a distributed storage service.

-> It stores the current state of the cluster so that if one master node goes down, another master node
can take a copy of this etcd look at the state and start working with the cluster.

-> Moving on to the worker node, the main component in the worker node is the Kubelet, this is the
guy which does bulk of the work.
It is responsible for creating the pods, the containers inside the pods and all that.

-> The api server communicates with the Kubelet on the worker node to get the work done.
Every worker node will have it
Kubelet component last and very important, the proxy component.

-> This is the network proxy and load balancer that every worker node will have any application that is
outside the cluster that if it wants to communicate with the containers inside the pods within the
cluster, they will have to go through this proxy, which can also do load balancing for us.
----

10) Volumes vs PV

-> What is the difference between volumes and persistent volumes in kubernetes?

-> Persistent volumes, Life-cycle will last beyond the scope of a pod or Node, that is, even if the
Node dies, the persistent volume will still live.That is not the case of volumes.
If a node dies, then the volume is gone.
----

11) What are PV and PVC

-> What is a persistent volume and what is a persistent volume claim, persistent volume is simply the
space on the cluster that can live as long as the cluster is alive a persistent volume claim is a request
for some space from the persistent volume for an application or a group of applications.
----

12) How to use a PVC

-> How do you create and use persistent volumes and persistent volume claims? we will create a yaml file to
create a persistent volume

-> We request for the capacity that we need on the cluster and the host path that should be used as the
storage space and also access modes like read, write, etc..

-> Once we have, that will create a persistent volume claim per application

wherein we request how much
space we need for the application from the persistent volume and what is the mode.

-> We then use this persistence volume claim in the volume section of our deployment yaml file and mounted
just like any other volume.

----

13) What are Config Maps and Secrets

-> How will you pass configuration information or sensitive details like usernames and passwords in a kubernetes
cluster?

-> We can use the config map object for this.
Once you use a config map, it will define data as keys and value pairs.

-> And we can bind that data and use it within our deployment, just like we bind any volume.
If it is sensitive data, we'll use secrets for it.

-> The difference between a config map and a secret is that secrets will be stored on a temporary location
and only when the secret information is required, they will be brought on to the pod.

-> And as soon as the data is used, the passwords, etc. are used and the pod is done with it.
They will be deleted from the pod location, but they will exist on the temporary location.

AWS

1) What are Regions Zones and Edge Locations

-> What are AWS regions zones and edge locations? AWS services are spread across the world and these
locations are called regions, these regions are divided into zones where the data centers are located
and these zones are interconnected to each other.

-> Every region will have at least two zones.

-> Even if one zone goes down, the other zone will back it up.

-> Edge locations exist between the end users or the clients and the zones, and they will catch the requests
and responses so that the response to the end user will be really fast.

-> Instead of every time coming to the zone, the request can go to the edge location and the response
will be cached and sent back really quick.

----

2) What is EC2

-> What is AWS EC2, EC2 stands for elastic compute cloud

-> It is the AWS service that gives us the capability to launch virtual servers, to run our applications
on the cloud.

-> AWS calls these instances instead of virtual machines or virtual servers.
----
3) What is a AMI

-> What is a AMI?

-> AMI stands for Amazon Machine Image, it acts as a template to launch our EC2 instances.
It will have the software like the operating system, if it is Java, will have the Java runtime.
Python will have the Python interpreter.

-> If we require MySQL database, that can be a part of a AMI and if we want Docker, Docker be a part
of the AMI as well.

-> Once we pick a Amazon machine image or AMI to launch a instance, we can launch any number of instances
using the same AMI if we want similar instances to run our applications.

-> Also, we can create a AMI from an existing running instance and use it to launch future instances as
well.
----
4) What are Spot Instances

-> What are ec2 spot instances when we launch a ec2 instances, we have the option of requesting
for a spot instance, by default, it'll be a on demand instance if you don't choose this option and
will be charged based on the usage.

-> But if you pick this option of spot instances, it will save us a lot of money because we get to choose
a price, the current price and you can offer a price.It's like bidding.

-> And once you get a spot instance, you get it for a very low rate.

-> As it says here, you can save up to 90 percent compared to OnDemand instances.
But the thing is, when you have a spot instance, A.W.S can take back that spot instance by giving you
just a two minute notice.

-> So if you are OK with that, then you can go for spot instances, for doing some quick, dirty work.
----
5) Public vs Elastic IP

-> What is the difference between a public IP address and elastic IP address on a ec2 instance?

-> When we create A launch ec2 instance, it will be allocated a private IP address and a public IP address,
this public IP address will keep changing every time we stop and start the instance, which could be
a pain for the client applications which are using it.That is where elastic IP address comes in.

-> Once we create, the elastic IP address will have a kind of a static IP,

which will not change.
    But of course we will be charged for it.

----
6) What are EC2 instance States

-> What are the different EC2 instance states? when we launch ec2 instance, it will go to the running
    state, when we stop it, it will go to the stop state and we can start it at some point.
    Again, we can also reboot the instance when it is up and running.

-> We can hibernate the instance temporarily and we can terminate the instance.

-> The difference between stopping and terminating instances that if we stop, we can restart that instance
    or start that instance again at a later point in time.

-> But if we terminate the instance it is gone, it will be deleted permanently.

-> Any volumes associated with that instance will also be deleted unless those volumes are marked not to
    be deleted during termination and will not be charged to any other instance is not running, will not
    be charged for that duration.

-> So if you stop the instance, you will not be charged for it.

----
7) How to Connect to a linux instance

-> How can we connect to a remote ec2 linux instance from our machine?

-> We can use ssh to connect to a remote, ec2 linux instance when we launch ec2 instance
    The last step of it asks us to download a key file.

-> This is where two keys are generated.

-> The public key is stored on aws. and the private key is what we download to connect to the ec2 linux
    instance.

-> We use the ssh command the IP address of that remote instance and then this key, the private key
    that we download.

----
8) How to Secure Ec2 instance

-> How do we secure a ec2 instance, to secure a ec2 instance, go to the security tab and we have
    a security group under the security details, which defines the inbound and outbound rules, for that ec2
    instance, if you go to the security group, will have inbound rules which will decide what kind of
    traffic can come into that easy to instant security group acts like a firewall and we can edit inbound
    rules and create our own rules or delete the existing rules to create a rule.

-> You click on ADD Rule and you can select a protocol that needs to be allowed the port range for protocol
    and the IP addresses that are allowed to access the ec2 instance.

-> Through that protocol you can have anywhere if you want to allow anybody over the Internet or you can
use custom IP ranges as well.
----
9) How to do Load Balancing

-> How can we balance the load across ec2 instances when we have multiple instances running our application,
we can do that using the load balancer service, we create a load balancer.

-> And when we do that, we get to pick different types of load balance that you can create application
load balancer network load balancer gateway load balancer, or classic load balancer.

-> Once you pick that, you can select the instances across which you want to balance the load and will
create a load balancer.

-> Once that is done, the load balancer will automatically spread the traffic across those instances and
during the creation process we will also specify on which ports the load balancer should listen and to
which ports on the instance, it should redirect that traffic to.

-> For example, 80 is what it should listen to and then it should redirect the traffic to 9091 on
the instances.
----
10) How to use Auto Scaling

-> How can we configure auto scaling for ec2 instances,that is when the load is high, the instance number
should go up and when there is no load on the applications, the instances should be taken down or terminated.
We can use auto scaling groups for this to use a auto scaling group.
It is a two step process.

-> The first step is to create a launch configuration.
This is where we specify that type of AMI we want to use and all that configuration.
This is just like launching a ec2 instances

-> Once we have that AMI information, we can create a auto scaling group where will use this launch
configuration and then we specify a scaling policy within the scaling policy will define the rules on
how the scaling should happen when the CPU is used 60 to 70 percent.

-> I want a new instance to be launched when the disk space is 80 percent.
I want to launch a new ec2 intance and when it goes below 50 percent, that ec2 instance should
be terminated.

-> All those can be defined using the scaling policies.

-> So it's a two step process first you go to auto scaling, create a launch configuration, use that launch
configuration, create a auto scaling group by specifying scaling policy where all the rules, like
CPU usage disc usage network usage can go in based on which the scaling

will automatically happen, both
scaling up and scaling down the instances.
----

11) Create custom user

-> Can you create a developer login that will give him access on the console only to the ec2 and S3 services to do that will go to the Identity and Access Management Service IAM create a new user, give a user name, for example.

-> Dev user give him the console access and we can either auto generate or give him a custom password. Once we give the password, we go to permissions section where we can assign him several inbuilt policies, or we can attach custom policies as well through the policies we can give him permissions, for example, ec2 full access, S3, full access and so on.

-> Once we create this user, we can share the user id and password to him through email and he'll be able to login and access just the ec2 and S3 services based on the access we give him through policies.
----

12) What is SNS

-> What is SNS? SNS stands for simple notification service, it allows us to react to events happening in other aws components and send out notifications.

-> There are two components, in SNS topics and subscriptions.

-> Topics are virtual channels which receive messages, and all the subscriptions to the topic will receive those messages on the topic and they can send out notifications by default.

-> They support email notifications, http notifications and also they can trigger a LAMBDA function. They can even put a message on a queue as well.

-> Using the SQS service in AWS. the senders of the message can be cloud watch alarms or any other component on AWS., or it can be our own micro service application which wants to use the topic to communicate with other micro services or simply send out notifications.
----

13) How to Send Notifications

-> If a devops engineer wants to receive an email every time the instances CPU usage or disk space is being used heavily, how can we accomplish that?

-> We can accomplish it. Using the simple notification service first will create a topic and a subscription, which can send out emails to a particular email id and then will create a cloud watch alarm within this alarm will specify our metrics.

-> When the CPU usage or the network traffic or the disk space goes above a particular number that alarm will

put a message onto the topic, which will eventually send out the notification because the subscription
will receive that message.
----
14) What is Cloudwatch

-> What is AWS cloud watch? AWS cloud watch service can continuously collect and monitor data from various AWS
services that our applications are using, like CPU usage, disk space, network traffic and more.

-> It can analyze all that data and also take actions based on that data.

-> It integrates beautifully with almost all the AWS services that are out there, starting from EBS to SNS
ec2 RDS and more.
----
15) S3 vs EBS vs EFS

-> What is the difference between S3 EBS and EFS? S3 which stands for Simple Storage Service, is a
object based storage service where we create buckets and store data or files inside those buckets.

-> It is good for storing artifacts like Jar's wars, are you and our application log files, etc., but
not for operating system or software.That is where EBS and EFS come in.

-> EBS stands for elastic block storage.
It allows reading and writing blocks of data.

-> We can mount it to EC2 instance, we can do the same with EFS, which stands for elastic file
system.But the key difference is in EBS

-> We can mount it only to one particular EC2 instance within a region, whereas in EFS we can mount it to
any number of instances across the regions.

-> Both EBS and EFS are good for storing operating system and other softwares as well.
----
16) What are the S3 Storage Classes

-> What are the different S3 storage classes and which one should we use?

-> The different S3 storage classes are standard standard IA. one zone IA , Glacier Intelligentsia and Outpost's.
Standard offers durability, availability and great performance, we should choose this if the data
is being accessed frequently, the data is stored across three availability zones.

-> Next is the standard infrequently accessed standard IA, which we should use if the data is infrequently
accessed, but it provides rapid access of data when required.

-> Next is one zone IA
the name itself says, our data will be stored in only one availability zone, reducing the cost
by 20 percent.

-> But if that availability zone goes down, the data will be lost.
Glacier provides us with the deep archiving and data can be retrieved within a few minutes or hours
if required, and will be charged a nominal fee for that.
S3 intelligent tier should be picked.

-> If we cannot decide which storage class we should go with for our applications, it will use artificial
intelligence to pick one of the storage classes based on how our applications are using the data.

-> And, of course, will be charged a little fee for that intelligence last is the outpost's outposts
Allow us to have S3 objects storage right on our premise, very close to our applications, running
on premise within our organization.And we can have a lifecycle policy.

-> We can define a lifecycle policy which will automatically transition the objects across these storage
classes as required.

----

17) What is CloudFormation

-> What is cloud formation service? cloud formation, is a infrastructure as code service using which
we can create the entire infrastructure required for our projects through simple yaml or Json template
files, we can use the cloud formation template designer to create our templates as well.

----

18) RDS vs DynamoDB

-> What is the difference between RDS and Dynamo DB in AWS? RDS stands for relational database services,
which makes it super easy to work with relational databases like MySQL, Postgre SQL, MSSQL, Maria
DB.

-> Amazon aurora which is a version of Mysql on the cloud.

-> When we use RDS, we don't have to worry about data replication in these database servers.
Automatically backups can be enabled and recovery can happen and also caching features can be simply
enabled all that will be taken care of for us by RDS.

-> On the other hand, Dynamo DB is the no sql database, just like Mongo DB.
It is a AWS version of no sql databases and it is good for unstructured data where we store
data as key value pairs dynamically.

----

19) What is Serverless

-> What does it mean to go Serverless, serverless is using all the components in the cloud to solve a business
problem without worrying about the non-functional requirements like scalability, reliability, etc,
while paying only for what we use and how much we use it?

-> It uses speed to market, the infrastructure costs will be very less, and we

get all the non functional
requirements for free because all the components on the cloud will have these non-functional requirements
like scalability and reliability implemented into them and will also pay based on demand.
That is how much ever we use it and how long we use it.

-> Based on that, we are going to pay for these services.

----

20) What is AWS Lambda

-> What is the AWS Lamda? AWS severless platform is comprised of a number of components S3 SNS, dynamo,
DB SQS and more AWS Lambda is a compute service
to write our own code, which will run in response to the events that happen within other serverless
components, just like how Java has a main method.

-> AWS Lambda will have a handler method and this method will be invoked by the lambda Service. AWS Lambda
can interact with other serverless components to get its work done.

Spring Cloud

1) What is Spring Cloud

-> What is spring cloud?

-> When we develop micro service applications using Springboot, these micro service applications need
to meet certain non-functional requirements like service, registration and discovery.

-> This is where one micro service will register itself so that other micro services can dynamically discover
it at runtime and use it load balancing both on the client side as well as on the provider or server side
fault tolerance

-> When something goes wrong within a micro service, it should handle it gracefully.
Easy integration among micro services as they communicate with each other.

-> Crosscutting concerns which are common across micro services like security, logging, etc., should
be handled in a single place instead of repeating them across micro services and distributed tracing
as we will Have many micro services within an organization.

-> We need to have distributed tracing as the client request goes from one micro service to another so
that we configure out what is going on in the system when something goes wrong.

-> So to address all these non-functional requirements, sprind cloud comes with a group or collection
of opensource components which will address all these non-functional requirements for our micro services
created using Spring boot.

----

2) What is Service Registration and Discovery

-> What is service, registration and discovery and how does springcloud address it when we have multiple
      micro service applications that communicate with each other instead of each of these micro service applications,
      remembering the urls and port numbers of other micro services will use the naming server component.

-> Once we have the naming server, these micro services will register themselves with this naming server
      by providing a unique application name or application I.D. and the urls and port numbers
      on which they can be communicated with.

-> So the naming server will act as a registry of application IDs and their communication information the
      url port numbers and micro services which want to communicate with these micro services can then
      talk to the naming server using just the application ID and they will get their addresses and the port numbers
      and they will be able to communicate with other micro services.

-> This naming server component in spring cloud is the Eureka server component, all the micro applications
      in spring cloud will register themselves with Eureka server and another micro services can discover them
      dynamically and communicate through the Eurekha Server

---

3) How to use Eureka Server

-> How do you use Eureka Server in your projects? To use Eureka Server
      The first step is to create Eureka Sever project and add the spring cloud Eureka
      Server dependancy to it.

-> Once you do that, you go to the application class of that project and mark it with @ enable Eureka
      Server you also configure the port on which the Eurekha server will listen for the claims on which the client
      can register with Eurekha Server.

-> Once we do that, we can start using Eurekha Server within our Micro service project by adding the
    Eureka client dependency to those projects.

-> So in the client project, we use Eurekha client.

-> Once you add the dependency in the application dot properties of our micro service projects.
      We will configure the application name by which the application should be recognized by Eureka Server
      and the url on which the Micro service applications can connect to the Eureka server and register themselves
      with it other micro services, can communicate with the Eureka server using this application name and fetch
      the url and other details of micro service.

---

4) How to do Client side Load Balancing

-> What is Client side load balancing and how did you configure it for your spring cloud project? client side load balancing is where the distribution of load to a provider micro service is done right on the client side.

-> In case of spring cloud, we use the load balancer component to accomplish it. It is super easy to configure the load balancer component.

-> We simply add the spring cloud starter load balancer dependency to our project. That's it.

-> Once we do that, the load balancer will automatically look at the application dot properties, figures out where the Eureka

-> Server is running and whenever we try to make a restful call using the fame client fame client beautifully integrates with load balancer automatically that restful call.

-> When that happens, the load balancer will receive it.

-> It will fetch the details of the provider micros service, the multiple instances that are running from the Eureka server, and it will automatically distribute the load across these instances.

-> Each request will be sent to a different.Service or provider instance for us.

---

5) What is API Gateway

-> What is the API gateway component in spring cloud?

-> When we have micro service applications that will have crosscutting concerns like security, logging, tracing, applying the rate limits on these micro services instead of repeating these non-functional requirements or crosscutting concerns across the micro service applications, we put them in one single place called the API Gateway in Spring cloud.This component is called API Gateway.

-> And once we have it in place, all the micro service applications will have to go through this API gateway and we can write our own custom filters, which can be pre filter post filter error filter route filter and more to apply our own logic while this routing happens from one micro service to another so we can address all the crosscutting concerns like security, etc. in one single place, instead of repeating that logic across micro services using the API gateway component.

---

6) How to use API Gateway

-> How did you use API gateway component in your project? using API Gateway is super simple, just like the Eureka Server will create an API Gateway project with the Spring cloud Starter Gateway dependency after that will configure the micro service routes which have to go through this API gateway and the API Gateway will act as a client to the Eureka server to fetch the url

port number and all those
      details for these micro service applications.

-> So once this is in place, all the requests for these micro services will have to go through the API
      gateway.

-> And if you want to address any crosscutting concern, you will write your own filter logic.
----

7) What are Sleuth and Zipkin

-> Water sleuth and zipkin for in spring cloud, as the micro service requests flow from one, micro service
      leads to another while serving a client's request.

-> We need to trace these requests to ensure that we can figure out if something goes wrong.
      We know where exactly went wrong and which application is responsible for it.
      That is where Sleuth comes in and it will add application name Trace ID and more to the logs.

-> So once you start using sleuth in micro service projects, it will automatically add a application
    name and a trace ID, which will be present in every log.

-> So using that log, we can trace the entire request from the client all the way to the First Micro service
    then the second micro service and so on.

-> So we can check each of the micro service logs and see if that is present and what went wrong with that
    particular request.

-> Zipkin compliments Sluthe and it provides a beautiful UI, a dashboard where we can do this tracing to
    see where the request has failed and why it has failed.
