

1. Describe the problem statement of your Project briefly.

- Nowadays drones are in very trend people wanted to catch their special moments or events via drones also by research we found that there is no website made in india who is giving their drone on rent on scheduled time also with services by thinking this we implemented our project and future scope of our project is
- We can provide drone training and guidance for user from drone experts.
- We can arrange online drone events.
- At the time of emergencies, we can make sure of drone availability in bulk quantity.
- We can provide service centres for customers.

2. Explain the flow of your project?

- ⇒ **Admin Panel:** First Admin is hard coded added using SQL query. Admin will login to our web application using role based login. (1) Admin can add/delete/update another admin's, dealer's, agent's information.
- (2) Admin can permit or reject the customers booking request and service request
- (3) Admin can check the status of agent
- (4) Admin can add/delete/update new stock
- (5) Admin will keep the records of everything like registered user and all
- ⇒ **Customer Panel :** At first customer will enter in our web application then home page of our web application will get loaded which will show him available drones if he wanted buy a drone we will find

two options on home page register and login we have provided role base login there. If customer is already registered then he will go for login and if not he has to register first.

After login customer can buy drone or to take drone on rent or want service for his already borrowed drones or can go for orders where he can see his orders

If customer wants buy a drone he can view products, its specification and category after deciding which drone he wants to buy he will book the drone or will add to cart the drone. **In add to cart he can** remove from cart, or can book drone also.

After booking he has to update his personal details like address mobile no etc. After that this request of booking will go to admin panel where admin can see availability of particular order if it is available he will allow the customer booking request. After that customer will select the payment mode if he selects online payment mode then after payment one invoice will generate in which payment details and on message “payment done successfully now wait for delivery” will get shown after delivery there is feedback form also available for customer where he can give his feedback and if he selects cash on delivery option again one invoice will generate in which booking details and on message “ your booking is done wait for delivery” will get shown.

After delivery he has to do payment and after payment he will get invoice of payment details and on message “payment done successfully and please give your feedback”.

If customer wants to take drone on rent he will go in rent section where rental drone home page will get loaded where he can see rental drone products if customers decides to take particular drone on rent he will select take drone on rent then he has to update his personal details like address mobile no etc. Also he has to mention particular scheduled time on he want drone. After filling this all detail he has to do payment online only. After payment one invoice will get generate in which payment details and one message “payment done successfully wait for scheduled time.” will get shown. At schedule time our agent will go to customer address with drone then he will perform his task. After performing task he will update to admin that task is done. Then customer can see feedback form.

If customer wants to return a drone then he will go for orders where he can see his recent orders. From there he can return his recently ordered drone if **he selects return drone** one query form will generate where customer has to select reason for return order. After filling it properly pick up of drone with verification as per selected reason will get done and if return order get successful one invoice will get generate in which drone details and one message “your drone amount N/A has been transferred to respected bank” .

If customer wants service for his drone then he will click on service then there is repairing service is available for only those who are our existing customer. Then customer will enter his drone’s product id and will wait for confirmation after that product Id will get matched in database, if match not found admin will reject for service and if match get found admit will fetch warranty details of this product and if the product is in warranty then only free service will get available for customer if not then customer have to pay for service After getting matched product key admin will permit the customer for service. After that one invoice will get generate on customer side in which product warranty details with book your free service for in warranty product or do payment for your service for not in warranty product one of these two buttons will get displayed on customers side. If customer selects for free service. Service will get booked and one invoice will get generated in which service details and product details with one message “Service booked now wait for service” will get shown. and if he selects for do payment again one invoice will get generate in which payment details and on message “payment done successfully now wait for service” will get shown. After completion of service then customer can see feedback form.

- ⇒ **Agent panel:** Agent will login and will check for scheduled task if there is task he will fetch customer details after that we will reached the destination at scheduled time and will perform his task throughout this he will keep updating every status to admin
- ⇒ **Dealer Panel:** Dealer will login and then he can see two options where he can update his new available stock and order request. If he has ne stock then he will update it so that admin can see it and make available it for customer

3. Introduction on project

⇒ Drone Galaxy is a web application build on React JS and J2EE platform in which user can buy new drone and also have rental facilities so that they can book it on special events with services. As you see there is not much websites made in India on drone with services. Our web application will be the first application because of its both selling and renting drone with services functionality. In which customer, admin, agent and dealer these are the modules and these application is fully responsive for front end we have used React JS , java script , html and css and for back end we have used spring boot , hibernate , JPA

4. Describe your Role in Project.

⇒ My role in the project was back end developer . Firstly we implemented all the entities of our project together after that we segregated our task one will make one controller , one repository an one service like this

5. Latest spring boot version, which version you have used ?

⇒ V2.5.4

6. Latest React version, which version you have used ?

⇒ 16.7.0

7. What are the limitations of your project?

- ⇒ Talking about limitations as in our project we have not implemented synchronization part as a result if one customer is trying to book a particular drone and another customer is also trying to book same drone then it is not possible there will be some ambiguity error will come . These is the limitation of our project sir

8. How will you improve the performance of your project? (memory related and response time)?

- ⇒ Using load balancing
Using network cache
Http caching
Reverse proxy server catching
Using database cache
Optimizing the application
Storing less data in session
Avoiding running out of memory
Using native SQL instead of running queries in a loop

9. Validation in project?

- ⇒ We have implemented validation in our project at front end using Java script and for back end we added one dependency called spring boot starter validation and after that in entity we added annotation like @not null @email @not empty @size etc

10. Where we add Configurations in spring project and list few names?

⇒ We added configuration in application.properties. To change the port number we used server.port , configured database using driver name , to change the user name we used spring.datasource.username , to change the password we used spring.datasource.password , to send mail of login info to the user we used SMTP server

11. Mention the Technology used in Project.

⇒ For front end we used react js and for back end we used hibernate, spring boot and JPA

12. What are the objectives of your project?

⇒ objectives of the project are customer satisfaction , cost reduction by providing some discounts and offers to customer, also services

13. Describe your Project schema.

⇒ In our project schema there are 9 tables user, rental user, address, product, category, user orders, order cart, payment and transaction

One to one

User to address

User to drone user

User to agent

Transaction to payment

Rental user to products

One to many

User to order cart
User to orders
User to transaction
Orders to products
Order cart to products
Category to products

Many to one

Products to orders
Products to category
Products order cart
Order cart to user
Orders to user
Transaction to user

Many to many

Products to transactions
Orders to user
Transactions to user
Products to category

14. Then how you did mapping with each other ?

⇒ using hibernate sir

15. Entities which you used in your project

1) Admin details

- 2) Dealer/supplier
- 3) Service provider
- 4) Safety agent
- 5) Products
- 6) Product category
- 7) Customer Orders
- 8) Order cart and wish list
- 9) Rental drone customer data
- 10) Transaction reports(shipping)
- 11) Payment
- 12) Customer details
- 13) Customer address details

16. How you all done the project module distribution?

⇒ My role in the project was back end developer . Firstly we implemented all the entities of our project together after that we segregated our task one will make one controller , one repository an one service like this

17. As a leader which are the project leader qualities are?

⇒ I believe that by sharing knowledge , you learn more. Problem solving , helpful attitude, team management like delegate tasks to the appropriate team member, Develop team schedules and assist in the successful on boarding and training of team members....These qualities should be in project leader

18. Who are the target beneficiary of your project?

⇒ All clients like event handlers , travellers , photographers and for security agents also

19. As a Team members what should be the role towards Project leader?

⇒ Executing all tasks assigned by the team leader diligently, on schedule, and to the highest standard. Working with team members to achieve daily, weekly, and monthly targets. Participating in meetings and voicing concerns as well as suggestions for improvement.

20. Any issue or discrepancy in the project where you disagree?

⇒ Yes sir , We were found disagreement while selecting project topic but then we solved it by holding grudges and giving chance to everyone for speak also and listening problem deeply and carefully and then finally we came to the solution

21. Have you used any Software Life cycle Model in your project?

⇒ Yes, we have used agile model in our project. We used four sprints. In first sprint we made ER-diagram, tables and analysed that what will be the flow of our project. In second sprint we made back end. In third sprint we made front end and in last sprint we did connectivity and implemented the part which we left to implement for some time

22. Is it possible to convert your project from Java to .NET or vice versa?

⇒ Yes sir , It is possible using tool like Java Language Conversion Assistant (JLCA) we can convert java project into dot net and for Stryon is providing iNet product so that no need to rewrite the code.

23. If you make use of Angular technology in your UI designing, what possible changes are required to be done?

- 1) React allows developers full freedom to define the structure of their projects. They can combine the templates, styles and business logic into a single JavaScript file, separate each into different files, or set an entirely new structure.

In other hand Angular has a defined code structure, as components are broken into separate templates, styles, and JavaScript files. The template uses HTML, the styles use CSS/SCSS/SASS, and the JavaScript file uses Typescript. This structure allows backend developers to generate users interfaces with code that is familiar to them.

we have to change this after that

- 2) Our second step is to translate React JSX syntax to Angular HTML syntax . React simply returns JSX without creating any extra wrapper elements. The Angular HTML syntax is not valid JSX and will throw an error. Therefore, we will need to rewrite this component to work with angular

24. Have you used any OOPs concepts in your Project?

- ⇒ Yes sir , our entity class which we made in our project describes encapsulation , interfaces which we made describes abstraction and whole project describes modularity and also in our project we are extending some super classes that shows polymorphism

25. Have you used any APIs in your project?

- ⇒ Mail sender api is used to send email

Axios is used for connectivity of back end and front end

Restful API like Do, Get, Post, Put to retrieve data

26. Have you used any Payment Gateway in your Project?

⇒ No sir, we have not used any payment gateway because to use this we have to borrow it so In our project when customers clicks on pay simply it shows one invoice .

27. Any errors that you face during project?

⇒ Spring boot allows adding Dependencies at the start but when we use to need any of the dependencies while making project then we have to add it manually , also mapping related errors and stack overflow error we faced .

28. What are the plus point of your team?

⇒ Overcome Obstacles :- When a team faces a challenge, they can utilize their varied learnings to come up with multiple solutions to tackle the problem. Teammates also help each other through their difficulties thus are better able to **handle any hiccups** along the way and could even proactively warn each other of foreseeable risks.

Trust :- Teamwork creates a strong bond and a team that enjoys working together. If you are able to trust a teammates, it provides a feeling of safety that enables teammates to open up and encourage each other. Trust in teammates also assists in open communication which could indirectly lead to increased trust in the organization and management too.

These are the plus points of our team

29. What is the minus point of your team?

⇒ Minus point of our team is that we were taking too much time on discussion on particular concept

30. How you call api or make request?

⇒ First we Find the URI of the external server or program.

Add an HTTP verb.

Include a header.

Include an API key or access token. then

Wait for the response.

31. What is hibernate why u have used hibernate in project? and why not jdbc?

⇒ Hibernate can perform automatic object mapping. It maps the object model's data to the schema of the database itself with the help of annotations.

In JDBC, one needs to write code to map the object model's data representation to the schema of the relational model.

Whereas Hibernate is database independent and the same code can work for many databases with minor changes.

JDBC is database-dependent, meaning that developers are required to write different codes for different databases.

Hibernate provides good support for lazy loading. JDBC does not support lazy loading.

Hibernate manages exceptions itself by marking them as unchecked.

JDBC code needs to be written in a try-catch block as it throws checked exceptions.

32. Did u know JPA what is it and why u haven't used it?

⇒ **Spring Boot JPA** is a Java specification for managing **relational** data in Java applications. It allows us to access and persist data between Java object/ class and relational database. JPA follows **Object-Relation Mapping** (ORM). It is a set of interfaces. It also provides a runtime **EntityManager** API for processing queries and transactions on the objects against the database. It uses a platform-independent object-oriented query language JPQL (Java Persistent Query Language).

33. How u have secured the apis in your project?

⇒ No sir , security we didn't use

34. Are the APIs from your project globally accecible?

⇒ No sir ,

35. What is controller in MVC?

⇒ The Controller **acts as an interface between both Model and View components**. This helps by processing the data using model to present those data as a presentation in view components

36. What is @service?

⇒ It is **used to mark the class as a service provider**. So overall @Service annotation is used with classes that provide some business functionalities.

37. Have you use any Business Model in Project?

⇒ Yes sir, As we have following MVC design pattern in between controllers and dao layer or persistent we have business logic layer where we validate the data before entering the data into database

38. What is the Front end used in Project.

⇒ We used react , Java script , axios , html , css in our project

39. What is the Back end used in Project?

⇒ We used spring boot, hibernate, JPA for back end

40. Which UML Diagrams you have designed and why?

⇒ Class diagram , use case diagram , sequence diagram , data flow diagram ,activity diagram because it saves time and Provides a better understanding of a system

41. Normalization is applicable to your project? (Yes/no). If so up to what level.

⇒ Yes sir , we have applied normalization till 1NF and 2NF

1NF – Each table cell should contain a single value.

Each record needs to be unique.

2NF – Eliminate partial functional dependency

3NF – A relation will be in 3NF if it is in 2NF and not contain any transitive partial dependency.

- 3NF is used to reduce the data duplication. It is also used to achieve the data integrity.
- If there is no transitive dependency for non-prime attributes, then the relation must be in third normal form.

NORMAL FORM	CHARACTERISTIC	SECTION
First normal form (1NF)	Table format, no repeating groups, and PK identified	6.3.1
Second normal form (2NF)	1NF and no partial dependencies	6.3.2
Third normal form (3NF)	2NF and no transitive dependencies	6.3.3
Boyce-Codd normal form (BCNF)	Every determinant is a candidate key (special case of 3NF)	6.6.1
Fourth normal form (4NF)	3NF and no independent multivalued dependencies	6.6.2

42. Describe database connection of your project?

⇒ **Driver class:** The driver class for the mysql database is **com.mysql.jdbc.Driver**.

Connection URL: The connection URL for the mysql database is **jdbc:mysql://localhost:3306/sonoo** where jdbc is the API, mysql is the database, localhost is the server name on which mysql is running, we may also use IP address, 3306 is the port number and sonoo is the database name. We may use any database, in such case, we need to replace the sonoo with our database name.

Username: The default username for the mysql database is **root**.

Password: It is the password given by the user at the time of installing the mysql database. In this example, we are going to use root as the password.

43. Differentiate between .NET and JAVA Project database connections?

⇒ **Driver class:** The driver class for the mysql database is **com.mysql.jdbc.Driver**.

Connection URL: The connection URL for the mysql database is **jdbc:mysql://localhost:3306/sonoo** where jdbc is the API, mysql is the database, localhost is the server name on which mysql is running, we may also use IP address, 3306 is the port number and sonoo is the database name. We may use any database, in such case, we need to replace the sonoo with our database name.

Username: The default username for the mysql database is **root**.

Password: It is the password given by the user at the time of installing the mysql database. In this example, we are going to use root as the password

That's how we do database connection in java

- Create Visual Basic .NET Windows application.
- Create ADO.NET objects.
- Use the SqlConnection object to open SQL Server connection.
- Use the SqlDataReader object to retrieve data from SQL Server.
- View database in Server Explorer.
- Use Server Explorer to open SQL Server connection.

That's how we do database connection in dotnet

44. Already so many applications are in market, how your application is different from market applications?

⇒ I think there is no application which is providing drones on rent so this may be this makes our application different from another applications

45. What all concepts of PG-DAC courses are applied to your Project?

⇒ OOPS, Collection, MySQL, React JS, CSS , HTML, Javascript , axios, Bootstrap, Spring boot , hibernate , JDBC , JPA

46. Is there any feature you have planned but not able to complete it?

⇒ Yes sir, we have not implemented synchronization part

47. Annotation of hibernate

⇒ @Entity Annotation, @Table Annotation @Id and @GeneratedValue @Column Annotation

48. Annotation of spring

⇒ @Controller @RequestMapping @PathVariable @RequestParam @ModelAttribute @RequestBody @ResponseBody @RequestHeader and @ResponseHeader.

49. Annotation of http

⇒ @GET maps to the HTTP GET method.

@HEAD maps to HTTP HEAD method.

@POST maps to HTTP POST method.

@PUT maps to HTTP PUT method.

@PATCH maps to HTTP PATCH method.

@DELETE maps to HTTP DELETE method.

50. Annotations used in project ?

⇒ @Entity @Table @Id and @GeneratedValue @Column @GET @POST @PUT @Controller @crossorigin @RestController @RequestMapping @RequestParam @RequestBody and @ResponseBody.

51. What is serialization and deserialization? have you used in your project ?

⇒ Serialization is the process of converting an object in to bytes, so that it can be transmitted over the network, or stored in a flat file and can be recreated later. Serialized object is an object represented as sequence of bytes that includes objects data, object type, and the types of data stored in the object.

Yes sir we have used response body that is nothing but the serialization

The @ResponseBody annotation tells a controller that the object returned is automatically serialized into JSON and passed back into the HttpServletResponse object.

Deserialization is the reverse process where the byte stream is used to recreate the actual Java object in memory

Yes sir we have used request body that's nothing but the deserialization

The @RequestBody annotation maps the Http Request body to a transfer or domain object, enabling automatic deserialization of the inbound HttpRequest body onto a Java object

52. How you handle the session in spring boot ?

⇒ Create Spring Boot project from Spring Initializer.

Add Spring Session jdbc dependency in pom.xml.

Add spring jdbc properties in application. properties.

Create rest end points to save, destroy/invalidate session.

53. What is session ?

⇒ A session is some data that is stored on the server. The server then provides an ID to the client, which the client can use to make requests back to the server.

54. How you did session management in react?

⇒ We used local storage to maintain the session. When user logged in customer id will get stored into local storage using

`localStorage.setItem("customerId", Id);` method

Key value

And that Id will get retrieve from local storage using

`localStorage.getItem("customerId");` method

Key

To clear the session we used

`local Storage. remove Item (" customerId");` If customer logged out .

55. Library used in react ?

⇒ react router dom , axios , sweet alert, bootstrap

56. What is ER diagram ?

⇒ **ER Diagram** stands for Entity Relationship Diagram, that displays the relationship of entity sets stored in a database

57. What is synchronization ? have you used in your project ?

- ⇒ synchronization is a technique used to ensure that when multiple threads are sharing same resources, there shouldn't be a "Race Condition" scenario. In other words, synchronization ensures that the lock on an object is acquired by only one thread at a time.
- No sir we have not used it in our project

58. What is SRS table ?

- ⇒ Software Requirements Specifications, also known as SRS, is the term used to describe an in-depth description of a software product to be developed. It's considered one of the initial stages of [the software development lifecycle \(SDLC\)](#).

59. What is restful API ?

- ⇒ A Restful API is an architectural style for an application program interface (API) that uses HTTP requests to access and use data. That data can be used to GET, PUT, POST and DELETE data types, which refers to the reading, updating, creating and deleting of operations concerning resources.

60. What is rest controller ?

- ⇒ Spring RestController annotation is used to create RESTful web services using Spring MVC. Spring RestController takes care of mapping request data to the defined request handler method. Once response body is generated from the handler method, it converts it to JSON or XML response

61. What is REST ?

- ⇒ REST is an acronym for Representational State Transfer and an architectural style for distributed hypermedia systems.

REST is web standards based architecture and uses HTTP Protocol for data communication. It revolves around resource where every component is a resource and a resource is accessed by a common interface using HTTP standard methods.

62. What is REDUX?

- ⇒ Redux is a library used for front end development. It is a state container for JavaScript applications which should be used for the applications state management. You can test and run an application developed with Redux in different environments

63. What is web service?

- ⇒ A web service is a collection of open protocols and standards used for exchanging data between applications and applications or systems. Software applications written in various programming languages and running on various platforms can use web services to exchange data over computer networks like the Internet in a manner similar to inter-process communication on a single computer

64. Diff between soap and rest ?

No.	SOAP	REST
1)	SOAP is a protocol .	REST is an architectural style .
2)	SOAP stands for Simple Object Access Protocol .	REST stands for REpresentational State Transfer .
3)	SOAP can't use REST because it is a protocol.	REST can use SOAP web services because it is a concept and can use any protocol like HTTP, SOAP.
4)	SOAP uses services interfaces to expose the business logic .	REST uses URI to expose business logic .
5)	JAX-WS is the java API for SOAP web services.	JAX-RS is the java API for RESTful web services.
6)	SOAP defines standards to be strictly followed.	REST does not define too much standards like SOAP.
7)	SOAP requires more bandwidth and resource than REST.	REST requires less bandwidth and resource than SOAP.
8)	SOAP defines its own security .	RESTful web services inherits security measures from the underlying transport.
9)	SOAP permits XML data format only.	REST permits different data format such as Plain text, HTML, XML, JSON etc.
10)	SOAP is less preferred than REST.	REST more preferred than SOAP.

65. Why singleton object is made ?

⇒ The primary purpose of a Singleton class is to restrict the limit of the number of object creation to only one. This often ensures that there is access control to resources, for example, socket or database connection.

The memory space wastage does not occur with the use of the singleton class because it restricts the instance creation. As the object creation will take place only once instead of creating it each time a new request is made.

66. What are scopes in spring ?

⇒ What is bean scope in spring?

In spring framework all bean has ascope , means every bean has its own visibility.

When we declare a class as a bean then by default the bean will be created under singleton scope.

How many types of bean scope

1. Singleton
2. Prototype
3. Request
4. Session
5. Global Session

⇒ What is singleton scope?

When will try to create multiple object of same bean then spring IoC container will return same object.

Singleton scope is default scope in bean. Singleton scope functionality is same as singleton class in java.

⇒ What is prototype scope?

When we will declare bean as prototype scope then every time new object will be return by spring IOC container.

⇒ **What is request scope?**

When we will declare a bean scope as request then for every HTTPRequest a new bean instance will be injected.

⇒ **What is session scope?**

When we will declare a bean scope as session then for every new HttpSession new bean instance will be injected.

67. How to create Restful web service ?

⇒ Step 1: Initializing a Spring Boot Project

Step 2: Connecting Spring Boot to the Database

Step 3: Creating a User Model

Step 4: Creating Repository Classes

Step 5: Creating a Controller applying Rest controller, request mapping, get, put, post, delete annotations in controller

Step 6: Compile, Build and Run

Step 7: Testing the Spring Boot REST APIs

This is how we can create Restful web Service

68. What is API methods?

Method	Description
GET	Retrieve information about the REST API resource
POST	Create a REST API resource
PUT	Update a REST API resource
DELETE	Delete a REST API resource or related component

69. In project on which method you applied rest controller?

⇒ Rest controller in a annotation and we used it for every controller in our project

70. Methods in http?

HTTP Method	CRUD operation
GET	Read
POST	Create
PATCH	Update
DELETE	Delete
PUT	Update/Replace

71. Can you make html dynamic ? if yest then how ?

⇒ Yes, we can make it using Java script

72. What is internet?

⇒ The Internet (or internet) is the global system of interconnected computer networks that uses the Internet protocol suite (TCP/IP) to communicate between networks and devices.

73. What is spring boot?

⇒ Spring Boot is an open source Java-based framework used to create a micro Service. It is developed by Pivotal Team and is used to build stand-alone and production ready spring applications.

74. What is spring ?

⇒ Spring is an open source development framework for enterprise Java. The core features of the Spring Framework can be used in developing any Java application, but there are extensions for building web applications on top of the Java EE platform. Spring framework targets to make J2EE development easier to use and promote good programming practice by enabling a POJO-based programming programming model

75. What is pom.xml?

⇒ A Project Object Model or POM is the fundamental unit of work in Maven. It is an XML file that contains information about the project and configuration details used by Maven to build the project. It contains default values for most projects. Some of the configuration that can be specified in the POM are the project dependencies, the plugins or goals that can be executed, the build profiles, and so on.

76. What is maven?

⇒ Maven is Java Build Tool.

Download dependencies from central/remote repository into local repository.

Compile source code.

Package compiled code into JAR/WAR files.

Install the packaged code. This things are followed by maven.

POM.XML is the Heart of Maven

77. Difference between Spring vs Spring boot?

Spring	Spring Boot
Spring Framework is a widely used Java EE framework for building applications.	Spring Boot Framework is widely used to develop REST APIs .
It aims to simplify Java EE development that makes developers more productive.	It aims to shorten the code length and provide the easiest way to develop Web Applications .
The primary feature of the Spring Framework is dependency injection .	The primary feature of Spring Boot is Autoconfiguration . It automatically configures the classes based on the requirement.
It helps to make things simpler by allowing us to develop loosely coupled applications.	It helps to create a stand-alone application with less configuration.
The developer writes a lot of code (boilerplate code) to do the minimal task.	It reduces boilerplate code.
To test the Spring project, we need to set up the sever explicitly.	Spring Boot offers embedded server such as Jetty and Tomcat , etc.
It does not provide support for an in-memory database.	It offers several plugins for working with an embedded and in-memory database such as H2 .

Developers manually define dependencies for the Spring project in **pom.xml**.

Spring Boot comes with the concept of **starter** in pom.xml file that internally takes care of downloading the dependencies **JARs** based on Spring Boot Requirement.

78. Difference between spring boot and spring mvc ?

Spring Boot	Spring MVC
Spring Boot is a module of Spring for packaging the Spring-based application with sensible defaults.	Spring MVC is a model view controller-based web framework under the Spring framework.
It provides default configurations to build Spring-powered framework.	It provides ready to use features for building a web application.
There is no need to build configuration manually.	It requires build configuration manually.
There is no requirement for a deployment descriptor.	A Deployment descriptor is required .
It avoids boilerplate code and wraps dependencies together in a single unit.	It specifies each dependency separately.
It reduces development time and increases productivity.	It takes more time to achieve the same.

79. Difference between spring and spring MVC ?

⇒ **Spring** is an open-source, lightweight comprehensive inversion of control (IoC) and aspect-oriented container framework

For building or developing applications, the Spring framework is considered to be the most widely used Java EE framework.

In pom.xml, the dependencies of our Spring project are manually defined by the developers

⇒ **Spring MVC** is considered to be the model view controller-based web framework under the Spring framework.

For building web applications, ready-to-use features are provided by it.

Each dependency is specified separately by it.

80. Spring MVC flow

⇒ All the incoming requests are intercepted by the DispatcherServlet that works as the front controller. The DispatcherServlet then gets an entry of handler mapping from the XML file and forwards the request to the controller.

The object of ModelAndView is returned by the controller.

The DispatcherServlet checks the entry of the view resolver in the XML file and invokes the appropriate view component.

81. Different between rest-controller and controller

@Controller	@RestController
@Controller is used to mark classes as Spring MVC Controller.	@RestController annotation is a special controller used in RESTful Web services, and it's the combination of @Controller and @ResponseBody annotation.
It is a specialized version of @Component annotation.	It is a specialized version of @Controller annotation.
In @Controller, we can return a view in Spring Web MVC.	In @RestController, we can not return a view.
@Controller annotation indicates that the class is a "controller" like a web controller.	@RestController annotation indicates that class is a controller where @RequestMapping methods assume @ResponseBody semantics by default.
In @Controller, we need to use @ResponseBody on every handler method.	In @RestController, we don't need to use @ResponseBody on every handler method.

82. Different between servlet and JSP

Servlet

Servlet is a java code.

Writing code for servlet is harder than JSP as it is HTML in java.

Servlet plays a controller role in the hasMVC approach.

Servlet is faster than JSP.

Servlet can accept all protocol requests.

In Servlet, we can override the service() method.

In Servlet by default session management is not enabled, user have to enable it explicitly.

In Servlet we have to implement everything like business logic and presentation logic in just one servlet file.

JSP

JSP is a HTML based code.

JSP is easy to code as it is java in HTML.

JSP is the view in the MVC approach for showing output.

JSP is slower than Servlet because the first step in the hasJSP lifecycle is the translation of JSP to java code and then compile.

JSP only accepts HTTP requests.

In JSP, we cannot override its service() method.

In JSP session management is automatically enabled.

In JSP business logic is separated from presentation logic by using JavaBeansclient-side.

Servlet

Modification in Servlet is a time-consuming task because it includes reloading, recompiling, JavaBeans and restarting the server.

JSP

JSP modification is fast, just need to click the refresh button.

83. What is web container

⇒ Web Container is a java application that controls servlet. Servlet does not have a main() method, So they require a container to load them.

Container is a place where servlet gets deployed.

When a client sends a request to web server that contains a servlet, the server sends that request to the container rather than to servlet directly. Container then finds out the requested servlet and pass the Http Request and response to servlet and loads the servlet methods i.e. doGet() or doPost().

84. React lifecycle?

⇒ **Initialization:** This is the stage where the component is constructed with the given Props and default state. This is done in the constructor of a Component Class.

⇒ **Mounting:** Mounting is the stage of rendering the JSX returned by the render method itself.

componentWillMount() Function: As the name clearly suggests, this function is invoked right before the component is mounted on the DOM i.e. this function gets invoked once before the render() function is executed for the first time.

componentDidMount() Function: Similarly as the previous one this function is invoked right after the component is mounted on the DOM i.e. this function gets invoked once after the render() function is executed for the first time

⇒ **Updating:** Updating is the stage when the state of a component is updated and the application is repainted.

componentWillUpdate() Function: As the name clearly suggests, this function is invoked before the component is rerendered i.e. this function gets invoked once before the render() function is executed after the updation of State or Props.

componentDidUpdate() Function: Similarly this function is invoked after the component is rerendered i.e. this function gets invoked once after the render() function is executed after the updation of State or Props.

- ⇒ **Unmounting:** As the name suggests Unmounting is the final step of the component lifecycle where the component is removed from the page.
- **componentWillUnmount() Function:** This function is invoked before the component is finally unmounted from the DOM i.e. this function gets invoked once before the component is removed from the page and this denotes the end of the lifecycle.

85. What is gradle?

- ⇒ Gradle is a build automation tool known for its flexibility to build software. A build automation tool is used to automate the creation of applications. The building process includes compiling, linking, and packaging the code. The process becomes more consistent with the help of build automation tools.

86. What is gradle.build?

- ⇒ The **Gradle build** is a process of creating a Gradle project. When we run a gradle command, it will look for a file called **build.gradle** in the current directory. This file is also called **the Gradle build script**.



87. What is thymeleaf?

The **Thymeleaf** is an open-source Java library that is licensed under the **Apache License 2.0**. It is a **HTML5/XHTML/XML** template engine. It is a **server-side Java template** engine for both web (servlet-based) and non-web (offline) environments. It is perfect for modern-day HTML5 JVM web development. It provides full integration with Spring Framework.

88. What is react ?

⇒ React is an **open-source front-end JavaScript library** that is used for building user interfaces, especially for single-page applications. It is used for handling view layer for web and mobile apps.

89. React vs angular?

 Angular	 React
Angular architecture is a bit complex; it follows the component-oriented architecture	ReactJS architecture is more simplified; it follows a Model-View-Control (MVC) architecture
Angular is written in Microsoft's TypeScript language	ReactJS is written in JavaScript language
Angular is compatible with any platform	ReactJS is not compatible
The latest version of Angular is Angular 7.2.0	The ReactJS latest version is 16.7.0
Angular supports bidirectional (two-way) data binding	ReactJS supports unidirectional (one-way) data binding
This framework is used in mobile apps and single-page applications	The React framework is used in web applications, hybrid apps, and native applications
In Angular, abstraction is medium	In React, abstraction is strong
Adding a JavaScript library to the source code in Angular is not possible	Adding a JavaScript library to the source code in React is possible
In Angular, documentation is slower	In ReactJS, documentation is faster
A single tool is required for testing a complete project in Angular	More than one tool is required for testing a complete project in React

90. What is component in react?

- ⇒ We can say that every application you will develop in React will be made up of pieces called components. Components make the task of building UIs much easier. You can see a UI broken down into multiple individual pieces called components and work on them independently and merge them all in a parent component which will be your final UI. Components in React basically return a piece of JSX code that tells what should be rendered on the screen
- ⇒ **Class component:** The class components are a little more complex than the functional components. The functional components are not aware of the other components in your program whereas the class components can work with each other. We can pass data from one class component to other class components. We can use JavaScript ES6 classes to create class-based components in React.
- ⇒ **Functional component:** Functional components are simply javascript functions. We can create a functional component in React by writing a javascript function. These functions may or may not receive data as parameters.

91. What are React Hooks ?

- ⇒ Hooks are JavaScript functions, but they impose two additional rules:
- Only call Hooks **at the top level**. Don't call Hooks inside loops, conditions, or nested functions.
- Only call Hooks **from React function components**. Don't call Hooks from regular JavaScript functions.

92. What is axios ?

- ⇒ **Axios:** Axios is a Javascript library used to make HTTP requests from node.js or XMLHttpRequests from the browser and it supports the Promise API that is native to JS ES6. It can be used intercept HTTP requests and responses and enables client-side protection against XSRF. It also has the ability to cancel requests.

93. What is fetch?

- ⇒ **Fetch:** Fetch API provides a **fetch() method** defined on the window object. It also provides a JavaScript interface for accessing and manipulating parts of the HTTP pipeline (requests and responses). The fetch method has one mandatory argument- the URL of the resource to be fetched. This method returns a Promise that can be used to retrieve the response of the request.

94. Axios vs fetch ?

Axios	Fetch
Axios has url in request object.	Fetch has no url in request object.
Axios is a stand-alone third party package that can be easily installed.	Fetch is built into most modern browsers; no installation is required as such.
Axios enjoys built-in XSRF protection.	Fetch does not.
Axios uses the data property.	Fetch uses the body property.
Axios' data contains the object .	Fetch's body has to be stringified .
Axios request is ok when status is 200 and statusText is 'OK'.	Fetch request is ok when response object contains the ok property .
Axios performs automatic transforms of JSON data .	Fetch is a two-step process when handling JSON data- first, to make the actual request; second, to call the <code>.json()</code> method on the response.
Axios allows cancelling request and request timeout .	Fetch does not.
Axios has the ability to intercept HTTP requests .	Fetch, by default, doesn't provide a way to intercept requests.
Axios has built-in support for download progress .	Fetch does not support upload progress.
Axios has wide browser support .	Fetch only supports Chrome 42+, Firefox 39+, Edge 14+, and Safari 10.1+ (This is known as Backward Compatibility).
Axios "GET" call can have body Content	Fetch "GET" call cannot have body Content

95. Wildcards in java ?

The **question mark (?)** is known as the **wildcard** in generic programming. It represents an unknown type. The wildcard can be used in a variety of situations such as the type of a parameter, field, or local variable; sometimes as a return type. Unlike arrays, different instantiations of a generic type are not compatible with

each other, not even explicitly. This incompatibility may be softened by the wildcard if ? is used as an actual type parameter.

Types of Wildcards:

Upper bound wildcard – If a variable is of **in** category, use extends keyword with wildcard.

Lower bound wildcard – If a variable is of **out** category, use super keyword with wildcard.

Unbounded wildcard – If a variable can be accessed using Object class method then use an unbound wildcard.

No wildcard – If code is accessing variable in both **in** and **out** category then do not use wildcards.

96. What is difference between jsp and servlet.

JSP	Servlet
JSP program code is tag based code.	Servlet program code is pure java code.
JSP run slower compared to Servlet as it takes compilation time to convert into Java Servlets.	Servlets run faster than JSP.
Coding of JSP is easier than Servlet because it is tag based.	Coding of Servlet is harder than JSP.
JSP will accept only http protocol request.	Servlet accepts all protocol requests.
In MVC pattern, JSP is used for showing output data i.e. in MVC it is a view.	In MVC pattern, Servlet plays a controller role.
We can build custom tags in JSP.	There is no such facility in Servlet.
We can achieve functionality of JSP at client side by running JavaScript at client side.	There are no such methods for servlets.
JSP are generally preferred when there is not much processing of data required.	Servlets are best for use when there is more processing and manipulation involved.

97. Explain dependency injection and IOC.

Dependency Injection: Dependency Injection is the main functionality provided by [Spring](#) IOC(Inversion of Control). The Spring-Core module is responsible for injecting dependencies through either Constructor or Setter methods. The design principle of Inversion of Control emphasizes keeping the Java classes independent of each other and the container frees them from object creation and maintenance. These classes, managed by [Spring](#), must adhere to the standard definition of Java-Bean. Dependency Injection in [Spring](#) also ensures loose-coupling between the classes.

IOC: The IOC container is responsible to instantiate, configure and assemble the objects. The IoC container gets informations from the XML file and works accordingly. The main tasks performed by IoC container are:

- to instantiate the application class
- to configure the object
- to assemble the dependencies between the objects

Spring IoC (Inversion of Control)

Spring Dependency Injection

Spring IoC Container is the core of Spring Framework. It creates the objects, configures and assembles their dependencies, manages their entire life cycle.

Spring Dependency injection is a way to inject the dependency of a framework component by the following ways of spring: Constructor Injection and Setter Injection

Spring helps in creating objects, managing objects, configurations, etc. because of IoC (Inversion of Control).

Spring framework helps in the creation of loosely-coupled applications because of Dependency Injection.

Spring IoC is achieved through Dependency Injection.

Dependency Injection is the method of providing the dependencies and Inversion of Control is the end result of Dependency Injection.

IoC is a design principle where the control flow of the program is inverted.

Dependency Injection is one of the subtypes of the IOC principle.

Aspect-Oriented Programming is one way to implement Inversion of Control.

In case of any changes in business requirements, no code change is required.

98. Scriptlet tags in jsp.

This tag allow user to insert java code in JSP. The statement which is written will be moved to jspservice() using JSP container while generating servlet from JSP. When client make a request, JSP service method is invoked and after that the content which is written inside the scriptlet tag executes.

99. What are Implicit objects?

Implicit objects are a set of Java objects that the JSP Container makes available to developers on each page. These objects may be accessed as built-in variables via scripting elements and can also be accessed programmatically by JavaBeans and Servlets. JSP provide you Total 9 implicit objects which are as follows :

1. *request*: This is the object of **HttpServletRequest** class associated with the request.
2. *response*: This is the object of **HttpServletResponse** class associated with the response to the client.
3. *config*: This is the object of **ServletConfig** class associated with the page.
4. *application*: This is the object of **ServletContext** class associated with the application context.
5. *session*: This is the object of **HttpSession** class associated with the request.
6. *page context*: This is the object of **PageContext** class that encapsulates the use of server-specific features. This object can be used to find, get or remove an attribute.
7. *page object*: The manner we use the keyword **this** for current object, page object is used to refer to the current translated servlet class.
8. *exception*: The exception object represents all errors and exceptions which is accessed by the respective jsp. The exception implicit object is of type **java.lang.Throwable**.
9. *out*: This is the **PrintWriter** object where methods like print and println help for displaying the content to the client.

100. What is Dispatcher servlet?

The DispatcherServlet is the front controller in Spring web applications. It's used to create web applications and REST services in Spring MVC. In a traditional Spring web application, this servlet is defined in the *web.xml* file.

101. Servlet and spring difference?

- ⇒ Servlet is the first step to learn java Web application development
- ⇒ Servlet: a server side java class to produce the html content.
- ⇒ Spring is a framework for J2EE application development famous now a days
- ⇒ Spring: A framework to develop Big Enterprise Application which include your servlet as well.

102. Referential integrity i.e foreign key how you achieve in your database ?

- ⇒ Using Hibernate for that we have applied hibernate dependency called hibernate entity manager and then we used @FK annotation to achieve foreign key in our database

103. If login then how you retrieved that data from database n what they called ?

- ⇒ Using JPA repository we retrieved the data from database for that we applied method called `findByEmailAndPassword(String email,String password);`

104. How many types of index explain ?

The types of indexes are:

- 1. Clustered:** Clustered index sorts and stores the rows data of a table / view based on the order of clustered index key. Clustered index key is implemented in B-tree index structure.
- 2. Nonclustered:** A non clustered index is created using clustered index. Each index row in the non clustered index has non clustered key value and a row locator. Locator positions to the data row in the clustered index that has key value.
- 3. Unique:** Unique index ensures the availability of only non-duplicate values and therefore, every row is unique.
- 4. Full-text:** It supports is efficient in searching words in string data. This type of indexes is used in certain database managers.
- 5. Spatial:** It facilitates the ability for performing operations in efficient manner on spatial objects. To perform this, the column should be of geometry type.
- 6. Filtered:** A non clustered index. Completely optimized for query data from a well defined subset of data. A filter is utilized to predicate a portion of rows in the table to be indexed.

105. Spring boot components ?

- ⇒ Spring Boot Framework has mainly four major Components.
- ⇒ Spring Boot Starters:- The main responsibility of Spring Boot Starter is to combine a group of common or related dependencies into single dependencies.
- ⇒ Spring Boot AutoConfigurator:- The main responsibility of Spring Boot AutoConfigurator is to reduce the Spring Configuration. If we develop Spring applications in Spring Boot, then we don't need to define single XML configuration and almost no or minimal Annotation configuration. Spring Boot AutoConfigurator component will take care of providing those information.
- ⇒ Spring Boot CLI:- Spring Boot CLI(Command Line Interface) is a Spring Boot software to run and test Spring Boot applications from command prompt. When we run Spring Boot applications using CLI, then it internally uses Spring Boot Starter and Spring Boot AutoConfigure components to resolve all dependencies and execute the application.
- ⇒ Spring Boot Actuator:- Providing Management EndPoints to Spring Boot Applications.
Spring Boot Applications Metrics are the main features

When we run our Spring Boot Web Application using CLI, Spring Boot Actuator automatically provides hostname as "localhost" and default port number as "8080". We can access this application using "<https://localhost:8080/>" end point. We actually use HTTP Request methods like GET and POST to represent Management EndPoints using Spring Boot Actuator.

106. Lifecycle of spring boot?

Bean life cycle is **managed by the spring container**. When we run the program then, first of all, the spring container gets started. After that, the container creates the instance of a bean as per the request, and then dependencies are injected. And finally, the bean is destroyed when the spring container is closed

107. What is @Component

⇒ @Component is **an annotation that allows Spring to automatically detect our custom beans**. In other words, without having to write any explicit code, Spring will: Scan our application for classes annotated with @Component. Instantiate them and inject any specified dependencies into them.

108. Bean class?

JavaBeans are **classes that encapsulate many objects into a single object** (the bean). It is a java class that should follow following conventions: Must implement Serializable. It should have a public no-arg constructor. All properties in java bean must be private with public getters and setter methods

109. What is lifecycle of functional component?

The component's lifecycle consists of three phases:

- **Mounting lifecycle methods:** that is inserting elements into the DOM.
- **Updating:** which involves methods for updating components in the DOM.
- **Unmounting:** that is removing a component from the DOM.

110. How to test website when you are given with the link only?

Entering credentials on login or register page we can check validations provided to the website or not ? After logged in taking back and again go to forward if the website page is loading it means we can say that session management is not done for website . Also we can check responsiveness of the website by running website on different different system. Making unnecessary load on website we can check performance of website.

111. JSP life cycle

- ⇒ **Translation of JSP page to Servlet** : This is the first step of the JSP life cycle. This translation phase deals with the Syntactic correctness of JSP. Here test.jsp file is translated to test.java.
- ⇒ **Compilation of JSP page** : Here the generated java servlet file (test.java) is compiled to a class file (test.class).
- ⇒ **Classloading** : Servlet class which has been loaded from the JSP source is now loaded into the container.
- ⇒ **Instantiation** : Here an instance of the class is generated. The container manages one or more instances by providing responses to requests.
- ⇒ **Initialization** : `jspInit()` method is called only once during the life cycle immediately after the generation of Servlet instance from JSP.
- ⇒ **Request processing** : `_jspService()` method is used to serve the raised requests by JSP. It takes request and response objects as parameters. This method cannot be overridden.
- ⇒ **JSP Cleanup** : In order to remove the JSP from the use by the container or to destroy the method for servlets `jspDestroy()` method is used. This method is called once, if you need to perform any cleanup task like closing open files, releasing database connections `jspDestroy()` can be overridden.

112. State and props

- ⇒ **State is the local state of the component which cannot be accessed and modified outside of the component.** It's equivalent to local variables in a function.
- ⇒ Props, on the other hand, make components reusable by giving components the ability to receive data from their parent component in the form of props.

113. Features in ES 6

- ⇒ Default parameters
- ⇒ Template literals (Template strings)
- ⇒ Tagged Templates
- ⇒ Destructuring assignment
- ⇒ Arrow function expressions
- ⇒ let and const
- ⇒ Spread and Rest syntaxes (...)
- ⇒ Object.assign() and Object.is()
- ⇒ Classes

114. Arrow function

- ⇒ Arrow functions are introduced in [ES6](#), which provides you a more accurate way to write the [functions in JavaScript](#). They allow us to write smaller function syntax. Arrow functions make your code more readable and structured.

Arrow functions are **anonymous functions** (the functions without a name and not bound with an identifier). They don't return any value and can declare without the function keyword. Arrow

functions cannot be used as the constructors. The context within the arrow functions is lexically or statically defined. They are also called as **Lambda Functions** in different languages.

Arrow functions do not include any prototype property, and they cannot be used with the new keyword.

115. Virtual dom?

The virtual DOM (VDOM) is a programming concept where an ideal, or “virtual”, representation of a UI is kept in memory and synced with the “real” DOM by a library such as ReactDOM. This process is called reconciliation.

116. Waterfall model and agile model?

S.no.	Purpose	Agile model	Waterfall model
1.	Definition	Agile model follows the incremental approach, where each incremental part is developed through iteration after every timebox.	Waterfall model follows a sequential design process.
2.	Progress	In the agile model, the measurement of progress is in terms of developed and delivered functionalities.	In the waterfall model, generally the measurement of success is in terms of completed and reviewed artifacts.
3.	Nature	Agile model is flexible as there is a possibility of changing the requirements even after starting the development process.	On the other hand, the waterfall model is rigid as it does not allow to modify the requirements once the development process starts.

4.	Customer interaction	In Agile model, there is a high customer interaction. It is because, after every iteration, an incremental version is deployed to the customer.	Customer interaction in waterfall model is very less. It is because, in a waterfall model, the product is delivered to the customer after overall development.
5.	Team size	It has a small team size. As smaller is the team, the fewer people work on it so that they can move faster.	In the waterfall model, the team may consist more members.
6.	Suitability	Agile model is not a suitable model for small projects. The expenses of developing the small projects using agile is more than compared to other models.	Waterfall model works well in smaller size projects where requirements are easily understandable. But waterfall model is not suitable for developing the large projects.
7.	Test plan	The test plan is reviewed after each sprint.	Test plan is reviewed after complete development.
8.	Testing	Testing team can take part in the requirements change phase without problems.	It is difficult for the testing team to initiate any change in needs.

117. software development life cycle (SDLC)

SDLC is a process followed for a software project, within a software organization. It consists of a detailed plan describing how to develop, maintain, replace and alter or enhance specific software. The life cycle defines a methodology for improving the quality of software and the overall development process.

118. Explain registration page sign up and sign in in your project ?

For registration and login we have created components called register.js and login.js inside register.js we have taken some entities like name , email, phone number, gender, DOB, security question and also added a checkbox of accept policy , terms and condition so it is mandatory to new user to fill these all details also we

have provided validation to our page using java script. For login page we have taken entities like email and password so that the old user can sign in easily.

119. Which are stereotype annotations ?

⇒ Spring Framework provides us with some special annotations. These annotations are used to create Spring beans automatically in the application context. `@Component` annotation is the main Stereotype Annotation. There are some Stereotype meta-annotations which is derived from `@Component` those are

1. `@Service`
2. `@Repository`
3. `@Controller`

1: @Service: We specify a class with `@Service` to indicate that they're holding the business logic. Besides being used in the service layer, there isn't any other special use for this annotation. The utility classes can be marked as Service classes.

2: @Repository: We specify a class with `@Repository` to indicate that they're dealing with **CRUD operations**, usually, it's used with DAO (Data Access Object) or Repository implementations that deal with database tables.

3: @Controller: We specify a class with `@Controller` to indicate that they're front controllers and responsible to handle user requests and return the appropriate response. It is mostly used with REST Web Services.

120. In how many ways we can start spring boot application ?

- ⇒ Running from an IDE
- ⇒ Running as a Packaged Application
- ⇒ Using the Maven Plugin
- ⇒ Using External Tomcat

⇒ Using the Gradle Plugin

121. Best practice to avoid as a developer to avoid null pointer exception ?

⇒ some of the best practices to avoid NullPointerException are:

- Use **equals()** and equalsIgnoreCase() method with String literal instead of using it on the unknown object that can be null.
- Use valueOf() instead of toString() ; and both return the same result.
- Use Java annotation @NotNull and @Nullable.

122. Mention all controllers in your project & Have you saved history of user ?,

⇒ Admin controller

User controller

product controller

booking controller

Pricing controller

No sir, we have not saved the history of user

123. Which properties we need to mention for database connectivity?,

⇒ dbName dbSchema dbConnectId dbConnectPwd enableSQLCheckData dbConnectionWaitTime
dbLoginTimeout dbConnectionMode

124. How swiggy works actually?

- ⇒ When you open the Swiggy application on your mobile phones or visit the Swiggy website then you will find a whole list of restaurants to order from.

Once the restaurant is selected and the order placed, the restaurant which has its own Swiggy application receives the order details and starts preparing for the order.

A broadcast signal is then sent to all the drivers in the vicinity who have their own driver application. Those willing to accept the order can choose to accept and deliver it.

Like other on-demand delivery startups, Swiggy also has integrated the Google Maps API which lets the customers know where their order is and the amount of time it will take for their order to get delivered in real-time.

125. Favourite app and what you want to make changes?

- ⇒ My favourite app is B612 app which is used for photo or video editing . I really loved this app but so many times the problem I have faced in this app is slow response. Its performance is quite low even I am having good processor and RAM , So by
- ⇒ Optimizing the application,
 - load balancing ,
 - Using network cache
 - Http caching
 - Reverse proxy server catching
 - Storing less data in session
 - We can change the performance of this app

126. Have you provided any type of security to your web application ?

⇒ Yes sir , We have provided normal spring security. For that we have added dependency called spring boot starter security and did some configurations in application.properties like spring.security.user.name and spring.security.user.password by using this security we have created one type of filter in front of our web application which was securing our web application and because of this whatever request will come it has to go through that filter first then our web application .

127. You did mail invoice ? Explain ?

For Email invoice we took reference from you tube and google. Firstly We have taken one dependency called spring-boot-starter-mail after that we created one email service class inside that we used java mail sender API and then created some methods like `SendSimpleEmail(String toEmail,String body, String subject)` and `sendMailWithInlineResources(String to, String subject, String fileToAttach)` which was throwing Messaging exception after that in user controller we imported email service class and overrided his methods and implemented code for it and finally in application.properties we added some configurations of SMTP server like `spring.mail.host`
`spring.mail.port`
`spring.mail.properties.mail.smtp.auth=true`
`spring.mail.properties.mail.smtp.starttls.enable=true`

128. How you perform role based login?

For role based login we have added extra columns in our user table i.e is_admin , is_agent, is_dealer and inside that column we put some admins, dealers and agents information. So at the time of login whoever user

is, he will put his email address and password in login page then that email and password will get match with database credentials if we found particular matched admin or agent or dealer of our staff then respected mapped home page will get open for that user. That's how we done role based login in our project

129. Which advanced features have you used in your project ?

⇒ We used react for view, axios for front end and back end connectivity , we provided role based login , mail sender API to send mail , real time reporting , invoicing , to make our web application responsive we have used media min and max queries

130. What are cookies and how you created it ?

⇒ Cookies are the textual information that is stored in key-value pair format to the client's browser during multiple requests. It is one of the state management techniques in session tracking. Basically, the server treats every client request as a new one so to avoid this situation cookies are used. When the client generates a request, the server gives the response with cookies having an id which are then stored in the client's browser. Thus if the client generates a second request, a cookie with the matched id is also sent to the server. The server will fetch the cookie id, if found it will treat it as an old request otherwise the request is considered new.

- We use a Cookie class that is present in **javax.servlet.http** package.
- After that we created an object of Cookie class and passed a name and its value.
- To add cookie in response, we used `addCookie(Cookie)` method of `HttpServletResponse` interface.
- To fetch the cookie, We used `getCookies()` method of `Request` Interface is used.

131. Which Design pattern are used in your project?

- ⇒ Singleton pattern:- The singleton pattern is a mechanism that ensures only one instance of an object exists per application. This pattern can be useful when managing shared resources or providing cross-cutting services, such as logging.

132. Which database is used in your Project? Why?

- ⇒ We used SQL database because it provides
- High speed, Using the SQL queries, the user can quickly and efficiently retrieve a large amount of records from a database.
 - No coding needed
 - Well defined standards
 - Portability
 - Interactive language
 - Multiple data view

133. Explain data access layer of your database?

- ⇒ Data Access Layer is the repository layer in the backend. In the Repository package, every interface interacts with Database. Either to save data or to fetch data from the Database. For that we implement JPA Repository interface which has some inbuilt methods like `save()`, `findById()`, `findAll()`, `delete()`, `deleteById()`, etc. We can also write our own queries using annotations like `@Query`, `@Param` and `@Procedure` especially for Select queries.

134. How to write stored procedure in your database? How to call from your data access layer?

135. How many web pages are present in your project? And in each module of your project?

⇒ There are total 18 Pages are present in our project

Login page, Register page, Admin Home Page, agent Home Page, Dealer Home page, order page, Invoice page, Services page, about page , feedback page, Gallery page, Contact page, testimonial page, pricing page, booking page, terms and condition page, view card page, update page.

136. How did you implement look and feel of your web pages? Have you used any framework and why?

⇒ Look and feel basically means UI and we have used React JS to build our UI. Concepts like Routing, Nested-Routing, Functional components, Hooks, different libraries such as sweet alert etc. CSS properties, Bootstrap and React Bootstrap. Layout of the page were made through HTML(called JSX in React) and then applying different CSS properties, using inbuilt classes of Bootstrap and components in React Bootstrap, This is how we built UI

137. Explain me any one part of your project from frontend to the database .

⇒ In our Signup part there are various input field where we input all the appropriate data and when we click on Signup Button, an API call is made to the backend using post method of axios library. the control goes to the backend and it searches for the specified API in the Controller. after finding the API, it will execute that method and call another method from the Service class. The Service class will again call a method from the repository layer and execute that method. After which the repository layer will deal with the database and perform the mentioned statements

138. Now explain me the reverse flow of the data from database to UI

⇒ When we make an API call to get list of drones using get method of axios, control goes to the controller and then to the Service layer and in turn goes to the repository layer. In repository layer, it executes the

method `findByItems(String items)`; Which executes the query mentioned in it and fetches list of drones from a particular location. This list is returned to the Service layer and then again passed on to the Controller and after that this List is sent as a response to the Frontend using `ResponseEntity<List<drone>` (`droneList, HttpStatus.OK`). This List is received at the FrontEnd in `response.data` which is then used to update the UI to fill the information

139. Write User Controller , Service , Entity

⇒ User Controller:

```
package com.cdac.mumbai.controller;

import java.util.List;

import javax.mail.MessagingException;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.CrossOrigin;
import org.springframework.web.bind.annotation.DeleteMapping;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.PutMapping;
import org.springframework.web.bind.annotation.RequestBody;

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

import com.cdac.mumbai.entity.User;
import com.cdac.mumbai.service.EmailService;
import com.cdac.mumbai.service.UserService;

@CrossOrigin
@RestController
public class UserController {
    @Autowired
    private EmailService emailService;
```

```

@Autowired
private UserService userService;

@GetMapping("/getallusers")
public List<User> getAllUsers(User user){
    return userService.getAllUsers();
}

@PostMapping("/forget")
public List<User> forgetPassword(@RequestBody User user){
    return userService.forgetPassword(user.getEmail(),user.getSecurityQues(), user.getSecurityAns());
}

@PostMapping("/findbyphone")
public List<User> findByPhone(@RequestBody User user) {
    return userService.findByPhone(user.getPhone());
}

@PostMapping("/findbyemail")
public List<User> findByEmail(@RequestBody User user) {
    return userService.findByEmail(user.getEmail());
}

@PostMapping("/login")
public List<User> getUserByEmailAndPassword(@RequestBody User user){
    return userService.getUserByEmailAndPassword(user.getEmail(),user.getPassword());
}

@PostMapping("/register")
public ResponseEntity<?> insertUser(@RequestBody User user) throws MessagingException {

    System.out.println("in registerUser controller method");
    System.out.println("in registerUser controller method"+user);

    User data=userService.insertUser(user);
    if(data==null)
        throw new MessagingException();
    String str=data.getEmail();
    String pass=user.getPassword();
}

```

```

        emailService.sendSimpleEmail(str,"You have registered successfully!\nusername = "+str+"\nPassword = "+pass ,"\nWelcome To DRONE GALAXY SERVICES !!");

        System.out.println("mail send succesffully with username :"+str);
        return ResponseEntity.status(HttpStatus.CREATED).body(data);

        //return userService.insertUser(user);
    }

    @PutMapping("/updatepassword")
    public User updatePassword(@RequestBody User user) {
        return userService.updatePassword(user);
    }

    @DeleteMapping("deleteuser/{uemail}")
    public String deleteUser(@PathVariable String uemail) {
        userService.deleteUser(uemail);
        return "Deleted";
    }
}

```

⇒ Booking Repository using JPA query:

```

package com.cdac.mumbai.repository;
import java.util.List;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.data.jpa.repository.Modifying;
import org.springframework.data.jpa.repository.Query;
import org.springframework.data.repository.query.Param;
import org.springframework.stereotype.Repository;
import com.cdac.mumbai.entity.Booking;

@Repository
public interface BookingDAO extends JpaRepository<Booking, Integer>{

    public List<Booking> findByEmail(String email);

    //public List<Booking> findByBookingId(int bookingId);
    public Booking findByBookingId(int bookingId);
}

```

```

    @Query("select r from Booking r where email=:e and status=:s")
    public List<Booking> pendingBookings(@Param("e") String email,@Param("s") boolean status);

    @Query("select r from Booking r where email=:e and status=:s")
    public List<Booking> viewBookings(@Param("e") String email,@Param("s") boolean status);

    @Query("select r from Booking r where status=:s")
    public List<Booking> viewAllPendingBookings(@Param("s") boolean status);

    @Query("select r from Booking r where status=:s")
    public List<Booking> viewAllBookings(@Param("s") boolean status);

    public List<Booking> findByCityAndDate(String city, String date);

}

```

⇒ User Repository using JPA:

```

package com.cdac.mumbai.repository;

import java.util.List;

import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.stereotype.Repository;

import com.cdac.mumbai.entity.User;

@Repository
public interface UserDao extends JpaRepository<User, String>{

    public List<User> findByEmailAndPassword(String email,String password);

    public List<User> findByEmail(String email);
    public List<User> findByPhone(String phone);

    public List<User> findByEmailAndSecurityQuesAndSecurityAns(String email,String securityQues,String securityAns);

}

```

⇒ User Entity:

```
package com.cdac.mumbai.entity;

import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.OneToOne;

@Entity
public class User {

    @Id
    private String email;
    private String password;
    private String name;
    private String city;
    private String phone;
    private String securityQues;
    private String securityAns;
    private String address;

    public String getAddress() {
        return address;
    }
    public void setAddress(String address) {
        this.address = address;
    }
    private boolean isAdmin=false;

    public String getEmail() {
        return email;
    }
    public void setEmail(String email) {
        this.email = email;
    }
    public String getPassword() {
        return password;
    }
    public void setPassword(String password) {
        this.password = password;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
}
```

```

    }
    public String getCity() {
        return city;
    }
    public void setCity(String city) {
        this.city = city;
    }
    public String getPhone() {
        return phone;
    }
    public void setPhone(String phone) {
        this.phone = phone;
    }
    public String getSecurityQues() {
        return securityQues;
    }
    public void setSecurityQues(String securityQues) {
        this.securityQues = securityQues;
    }
    public String getSecurityAns() {
        return securityAns;
    }
    public void setSecurityAns(String securityAns) {
        this.securityAns = securityAns;
    }
    public boolean isAdmin() {
        return isAdmin;
    }
    public void setAdmin(boolean isAdmin) {
        this.isAdmin = isAdmin;
    }
    @Override
    public String toString() {
        return "User [email=" + email + ", password=" + password + ", name=" + name + ", city=" + city + ", phone="
            + phone + ", securityQues=" + securityQues + ", securityAns=" + securityAns + ", isAdmin=" + isAdmin
            + ",address=" + address
            + " ]";
    }
}

```

=====

==

