**1. JSON**

JSON stands for JavaScript Object Notation. It is a lightweight format for storing and transporting data. It is often used when data is sent from a server to a web page. It is "self-describing" and easy to understand

***JSON Syntax Rules***

* Data is in name/value pairs
* Data is separated by commas
* Curly braces hold objects
* Square brackets hold arrays

**2. JSON Array**

JSON array represents ordered list of values. JSON array can store multiple values. It can store string, number, boolean or object in JSON array. In JSON array, values must be separated by comma. The **[** (square bracket) represents JSON array.

**3. Controller and RestController**

***Controller***

we write this controller class to handle requests coming from the client. Then, the controller invokes a business class to process business-related tasks, and then redirects the client to a logical view name

***RestController***

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.

**4. Soap and Restful Webservices**

Simple Object Access Protocol(SOAP) and Representational State Transfer (REST) are by far the **most used options for accessing web services**

**Soap:**

* SOAP is a **protocol**. **It can't use REST** because it is a protocol.
* **JAX-WS** is the java API for SOAP web services.
* **defines standards**to be strictly followed.
* SOAP **permits XML** data format only.
* SOAP is **less preferred** than REST.

**Restful:**

* REST is an **architectural style**. It **can use SOAP** web services because it is a concept and can use any protocol like HTTP, SOAP.
* **JAX-RS** is the java API for RESTful web services.
* REST does not define too much standards like SOAP.
* RESTful web services **inherits security measures** from the underlying transport.
* REST **permits different** data format such as Plain text, HTML, XML, JSON etc.

**5. What is difference between web application and webservice application**

***Web Application***

* Web Application is a complete Application with a Graphical User Interface (GUI), however, web services do not necessarily have a user interface since it is used as a component in an application.
* Web Application can be access through browsers.

**Web** applications are websites with functionality and interactive elements. Gmail, Facebook, YouTube, Twitter, etc. are all web apps that are dynamic, and built for user engagement.

***Webservice Application***

* Web Services can be used to transfer data between Web Applications.
* Web Services can be accessed from any languages or platform.

***types of web services:*** XML-RPC, UDDI, SOAP, and REST: XML-RPC (Remote Procedure Call) is the most basic XML protocol to exchange data between a wide variety of devices on a network.

**6. response body and response entity**

**@ResponseEntity** represents an HTTP response, including headers, body, and status. As a result, we can use it to fully configure the HTTP response. If we want to use it, we have to return it from the endpoint; Spring takes care of the rest. ResponseEntity is a generic type.

**@ResponseBody** is a Spring annotation which binds a method return value to the web response body. It is not interpreted as a view name. It uses HTTP Message converters to convert the return value to HTTP response body, based on the content-type in the request HTTP header.

**7. what is ddl and what is ddl-auto**

Spring Boot offers Schema and Data initialization through the JPA module. We can control this behavior using the following properties.

Spring JPA uses hibernate as its vendor.

1. spring.jpa.generate-ddl – This configuration takes a boolean value to either enable or disable schema initialization.
2. spring.jpa.hibernate.ddl-auto – This property takes an enum that controls the schema generation in a more controlled way.

The ***spring.jpa.hibernate.ddl-auto*** takes one of ***none***, ***validate***, ***update***, ***create***, and ***create-drop***. By explicitly specifying one of these options, you are instructing Spring Boot on how to initialize the schema.

***None:*** No database Schema initialization

***Create:*** Drops and creates the schema at the application startup. With this option, all your data will be gone on each startup.

***create-drop:*** Creates schema at the startup and destroys the schema on context closure. Useful for unit tests.

***Validate:*** Only checks if the Schema matches the Entities. If the schema doesn’t match, then the application startup will fail. Makes no changes to the database.

***Update:*** Updates the schema only if necessary. For example, If a new field was added in an entity, then it will simply alter the table for a new column without destroying the data.

**8. JPARepository, PagingAndSortingRepository and CRUDRepositoy**

***JPA Repository***

JPA repositories are created by extending the JpaRepository library consisting of implementation of different functions, methods, and other related dependent data types to enable persistence in web or desktop applications designed using JAVA.

***PagingAndSortingRepository***

[PagingAndSortingRepository](https://docs.spring.io/spring-data/commons/docs/current/api/org/springframework/data/repository/PagingAndSortingRepository.html) is an extension of [CrudRepository](https://docs.spring.io/spring-data/commons/docs/current/api/org/springframework/data/repository/CrudRepository.html) to provide additional methods to retrieve entities using the pagination and sorting abstraction. It provides two methods :

* Page findAll(Pageable pageable) – returns a Page of entities meeting the paging restriction provided in the Pageable object.
* Iterable findAll(Sort sort) – returns all entities sorted by the given options. No paging is applied here.

**CRUDRepositoy**

The CrudRepository interface provides methods for CRUD operations, so it allows you to create, read, update and delete records without having to define your own methods. The PagingAndSortingRepository provides additional methods to retrieve entities using pagination and sorting.

**9. what are mappings one-to-one, one-to-many, many-to-one, many-to-many**

***one-to-one***

The One-To-One mapping represents a single-valued association where an instance of one entity is associated with an instance of another entity. In this type of association one instance of source entity can be mapped atmost one instance of target entity.

**one-to-many**

The One-To-Many mapping comes into the category of collection-valued association where an entity is associated with a collection of other entities. Hence, in this type of association the instance of one entity can be mapped with any number of instances of another entity.

**many-to-one**

The Many-To-One mapping represents a single-valued association where a collection of entities can be associated with the similar entity. Hence, in relational database any more than one row of an entity can refer to the similar rows of another entity.

**many-to-many**

The Many-To-Many mapping represents a collection-valued association where any number of entities can be associated with a collection of other entities. In relational database any number of rows of one entity can be referred to any number of rows of another entity.

**10.what are relations in parent child table is-a, uses-a, has-a**

Child tables and parent tables are just normal database tables, but they're linked in a way that's described by a parent–child relationship. It's usually used to specify where one table's value refers to the value in another table (usually a primary key of another table).

**11.what is transient in jpa**

@Transient annotation is used to mark a field to be transient for the mapping framework, which means the field marked with @Transient is ignored by mapping framework and the field not mapped to any database column (in RDBMS) or Document property