**Interview Questions:Restful Web Services**

**1.What is Restful Web Service?**

1. **REST stands for REpresentational State Transfer.**
2. **REST is an architectural style not a protocol.**
3. **The underlying protocol for REST is HTTP.**
4. **Restful Web Service is a lightweight, maintainable, and scalable service that is built on the REST architecture.**
5. **Here web services are viewed as resources and can be identified by their URIs.**
6. **Web services clients use that URI to access the resource.**
7. **It consists of two components: a REST server which provides access to the resources and a REST client which accesses and modifies the REST resources.**

**2. Rest Architecture**

**In restful architecture,**

**1.every resource should be accessible via the normal HTTP commands of GET, POST, PUT, or DELETE. So if someone wanted to get a file from a server, they should be able to issue the GET request and get the file.**

**2.Client-server is the typical architecture where the server can be the web server hosting the application, and the client can be as simple as the web browser.**

**3.Stateless means that the state of the application is not maintained in REST. For example, if you delete a resource from a server using the DELETE command, you cannot expect that delete information to be passed to the next request.**

**3. Important features of Restful web services**

**Some important features of Restful web services are:**

1. **Resource identification through URI: Resources are identified by their URIs .So, a client can directly access a RESTful Web Services using the URIs of the resources**
2. **Uniform interface: Resources are manipulated using a fixed set of four create, read, update, delete operations: PUT, GET, POST, and DELETE.**
3. **Client-Server: A clear separation concerns is the reason behind this constraint. Separating concerns between the Client and Server helps improve portability in the Client and Scalability of the server components.**
4. **Stateless: each request from client to server must contain all the information necessary to understand the request, and cannot take advantage of any stored context on the server.**
5. **Cache: to improve network efficiency responses must be capable of being labeled as cacheable or non-cacheable.**
6. **Named resources – the system is comprised of resources which are named using a URL.**
7. **Interconnected resource representations – the representations of the resources are interconnected using URLs, thereby enabling a client to progress from one state to another.**
8. **Layered components – intermediaries, such as proxy servers, cache servers, gateways, etc, can be inserted between clients and resources to support performance, security, etc.**
9. **Self-descriptive messages: Resources are decoupled from their representation so that their content can be accessed in a variety of formats, such as HTML, XML, plain text, PDF, JPEG, JSON, and others.**

**4.What are the methods used in Restful services**

1. **GET : It defines a reading access of the resource without side-effects.This operation is idempotent i.e.they can be applied multiple times without changing the result**
2. **PUT : It is generally used for updating resource. It must also be idempotent.**
3. **DELETE : It removes the resources. The operations are idempotent i.e. they can get repeated without leading to different results.**
4. **POST : It is used for creating a new resource. It is not idempotent.**

#### **5. What do you mean by Idempotent and which HTTP methods are idempotent? – video required**

* **An Idempotent HTTP method is a HTTPS method that can be called many times without different outcomes.**
* **For example : Delete is an idempotent method because when you first use delete, it will delete the resource (initial application) but after that, all other requests will have no result because the resource is already deleted. Get, put and delete are HTTP Idempotent methods.**

#### **6. What are differences between Post and Put Http methods?**

**POST :**

1. **It is used for creating a new resource.**
2. **It is not idempotent.**

**PUT :**

1. **It is generally used for updating resources.**
2. **It is idempotent.**

**7. What happens if resources are shared by multiple clients**

**New resource instance is created for each request, so we dont to implement thread safety or synchronization. It is default thread safe.**

**8.what is JAX-RS?**

**Java API for RESTful Web Services (JAX-RS), is a set of APIs to develop REST services.**

**JAX-RS is part of the Java EE6, and make developers to develop REST web application easily**

#### **9. What are REST frameworks that you are aware of and which can be used to create Restful web services?**

**There are multiple Rest framework that can be used to create Restful web services such as**

* **Jersey**
* **RestEasy**
* **Restlet**
* **CFX**
* **Spring Rest web services**

**10. What are some important annotations which you use to create Restful web services?**

**Some of important annotations which are used for creating web services are:**

* **@Path : This is used to set path for URI at class level or method level**
* **@GET,@POST,@PUT,@DELETE : There are annotations corresponds to HTTP methods**
* **@Produces(MediaType.TEXT\_XML [, more-types ]): @Produces defines which MIME type is delivered by a method**
* **@PathParam: Used to inject values from the URL into a method parameter.**
* **@Consumes(MediaType.TEXT\_XML) : @Cosumes defines which MIME type will be consumed by the method .**
* **@QueryParam - used to inject query values from the URI into a method,**

#### **11. Can you use get method to create Resources rather than post?**

**No, Get should be used only for resource retrieval and not for resource creation.**

#### **12. What are ways to test Restful web services?**

**You require a restful client to test restful web services. You can use:**

* **Postman for chrome browser**
* **poster for firefox**

**13. principles of Restful web services**

1. **RESTFul Client-Server**

* **This is the most fundamental requirement of a REST based architecture.**
* **It means that the server will have a RESTful web service which would provide the required functionality to the client.**
* **The client send's a request to the web service on the server.**
* **The server would either reject the request or comply and provide an adequate response to the client.**

1. **Stateless**

* **The concept of stateless means that it's up to the client to ensure that all the required information is provided to the server.**
* **This is required so that server can process the response appropriately.**
* **The server should not maintain any sort of information between requests from the client. It's a very simple independent question-answer sequence.**
* **The client asks a question, the server answers it appropriately.**
* **The client will ask another question. The server will not remember the previous question-answer scenario and will need to answer the new question independently.**

**3. cache**

* **The Cache concept is to help with the problem of stateless which was described in the last point.**
* **Since each server client request is independent in nature, sometimes the client might ask the server for the same request again. This is even though it had already asked for it in the past.**
* **This request will go to the server, and the server will give a response.**
* **This increases the traffic across the network.**
* **The cache is a concept implemented on the client to store requests which have already been sent to the server.**
* **So if the same request is given by the client, instead of going to the server, it would go to the cache and get the required informa tion.**
* **This saves the amount of time and network traffic from the client to the server.**

1. **Layered System**

* **The concept of a layered system is that any additional layer such as a middleware layer can be inserted between the client and the actual server hosting the RESTFul web service (The middleware layer is where all the business logic is created. This can be an extra service created with which the client could interact with before it makes a call to the web service.).**
* **But the introduction of this layer needs to be transparent so that it does not disturb the interaction between the client and the server.**

**5. Interface/Uniform Contract**

**This is the underlying technique of how RESTful web services should work. RESTful basically works on the HTTP web layer and uses the below key verbs to work with resources on the server**

* **POST - To create a resource on the server**
* **GET - To retrieve a resource from the server**
* **PUT - To change the state of a resource or to update it**
* **DELETE - To remove or delete a resource from the server**

**14.what is URI how do you define. Restful web services for uri?**

**Rest API’s use Uniform resource identifiers(URI) to address resources. Below is the syntax to define the URI.**

**Syntax: URI= schema:[//authority]path[?query][#fragment]**

**authority=[userinfo]host[:port]**

**Schema: domain(http)**

**Eg:** [**https://john.deo@www.example.com:123/forum/questions/?tag=networking&order=newst#top**](https://john.deo@www.example.lcom:123/forum/questions/?tag=networking&order=newst#top)

**Schema - https**

**authority=** [**john.deo@www.example.com:123**](https://john.deo@www.example.lcom:123/forum/questions/?tag=networking&order=newst#top)

**Path =**[**forum/questions/**](https://john.deo@www.example.lcom:123/forum/questions/?tag=networking&order=newst#top)

**query= tag=[networking&order=newst](https://john.deo@www.example.lcom:123/forum/questions/?tag=networking&order=newst" \l "top)**

**fragment = top**

**15.how to upload the document using rest ful webservices**

**@Post**

**@Consumes(MediaType.Multipart\_form\_data)**

**Public Response uploadFile{**

**@FormDataParam(“file”)InputStream uploadedInputStream,**

**@FormDataParam(“file”)FormDataContentDisposition fileDetail{**

**}**

**16. What is the difference between @QueryParam and @PathParam?**

**@QueryParam - Binds the values of a Http query parameter to a resource method parameter, resource class field or resource class bean property.**

**Query param is optional.**

**URI: users/query?from=100**

**@path(“/users)**

**Public class USerService{**

**@GET**

**@Path(“/query”)**

**publicResponse getUsers(@QueryParam(“from”) int from)**

**}**

**@PathParam - Binds the value of a URI template path parameter to a resource method parameter,resource class field, or resource class bean property**

**Path param is mandatory. It should map to method**

**@Path(“/uers/{username}”)**

**Public class UserService{**

**@GET**

**@Produces(“text/xml”)**

**Public string getUsers(@PathParam(“username”) String username)**

**}**

**17.what is the difference b/w produces and consumers**

**1. @Produces**

* **@Produces annotation is used to define the mediatype which resource (method) can produces and send back to the client.**
* **The value of @Produces is array of mimetypes**
* **Eg: @Produces({“images/jpg,images/png”)**
* **We can declare @Produces at class level or method level**

**2.@Consumes**

* **used to specify the media type which resource can accept.**
* **Value is array of string acceptable mime types**
* **Eg: @Consumes({“text/plain,text/html”})**
* **We can declare @Consumes at class level or method level**

**18.which framework used to test integrated test cases**

* **Rest Assured**
* **Cucumber**
* **SpringTest**

**19. Status Codes in Http**

1. **1XX- Informational - 100 -continue, 102-processing**
2. **2XX - Success- 200-Ok,201-Created,202-Accepted**
3. **3XX- Redirection - 301- moved permanently**
4. **4XX - Client error**
5. **5XX - Server error**

**20 How to secure rest API/web services**

**Since restful web services are stateless we need to make sure every request go after proper authentication**

1. **Basic authentications- credential will be encoded in base 64**
2. **OAuth/sso**
3. **HMAC -**