# InterviewBit: 43Q

### 1. What do you understand by RESTful Web Services?

RESTful web services are services that follow REST architecture. REST stands for Representational State Transfer and uses HTTP protocol (web protocol) for implementation. These services are lightweight, provide maintainability, scalability, support communication among multiple applications that are developed using different programming languages. They provide means of accessing resources present at server required for the client via the web browser by means of request headers, request body, response body, status codes, etc.

### 2. What is a REST Resource?

Every content in the REST architecture is considered a resource. The resource is analogous to the object in the object-oriented programming world. They can either be represented as text files, HTML pages, images, or any other dynamic data.

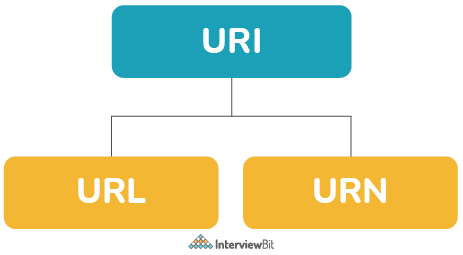
* The REST Server provides access to these resources whereas the REST client consumes (accesses and modifies) these resources. Every resource is identified globally by means of a URI.

### 3. What is URI?

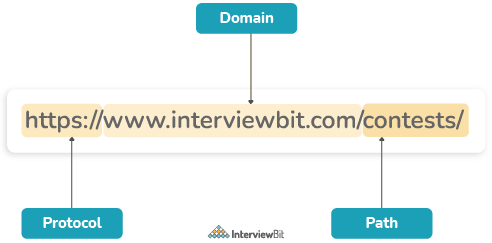
**Uniform Resource Identifier** is the full form of URI which is used for identifying each resource of the REST architecture. URI is of the format:

<protocol>://<service-name>/<ResourceType>/<ResourceID>

There are 2 types of URI:



* **URN:**Uniform Resource Name identifies the resource by means of a name that is both unique and persistent.
  + URN doesn’t always specify where to locate the resource on the internet. They are used as templates that are used by other parsers to identify the resource.
  + These follow the urn scheme and usually prefixed with urn:. Examples include
    - urn:isbn:1234567890 is used for identification of book based on the ISBN number in a library application.
    - urn:mpeg:mpeg7:schema:2001 is the default namespace rules for metadata of MPEG-7 video.
  + Whenever a URN identifies a document, they are easily translated into a URL by using “resolver” after which the document can be downloaded.
* **URL:**Uniform Resource Locator has the information regarding fetching of a resource from its location.
  + Examples include:
    - http://abc.com/samplePage.html
    - ftp://sampleServer.com/sampleFile.zip
    - file:///home/interviewbit/sampleFile.txt
  + URLs start with a protocol (like ftp, http etc) and they have the information of the network hostname (sampleServer.com) and the path to the document(/samplePage.html). It can also have query parameters.



**You can download a PDF version of Rest Api Interview Questions.**

[**Download PDF**](javascript:void(0))

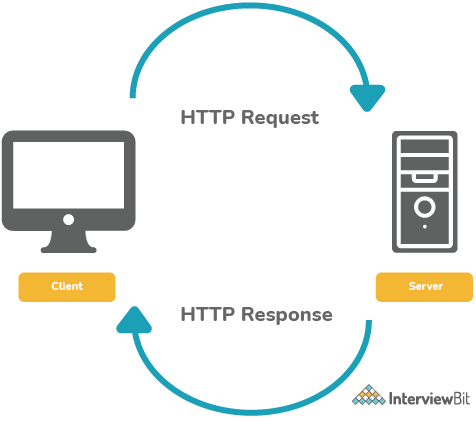
### 4. What are the features of RESTful Web Services?

Every RESTful web service has the following features:

* The service is based on the Client-Server model.
* The service uses HTTP Protocol for fetching data/resources, query execution, or any other functions.
* The medium of communication between the client and server is called “Messaging”.
* Resources are accessible to the service by means of URIs.
* It follows the statelessness concept where the client request and response are not dependent on others and thereby provides total assurance of getting the required data.
* These services also use the concept of caching to minimize the server calls for the same type of repeated requests.
* These services can also use SOAP services as implementation protocol to REST architectural pattern.

### 5. What is the concept of statelessness in REST?

The REST architecture is designed in such a way that the client state is not maintained on the server. This is known as statelessness. The context is provided by the client to the server using which the server processes the client’s request. The session on the server is identified by the session identifier sent by the client.



### 6. What do you understand by JAX-RS?

As the name itself stands (JAX-RS= Java API for RESTful Web Services) is a Java-based specification defined by JEE for the implementation of RESTful services. The JAX-RS library makes usage of annotations from Java 5 onwards to simplify the process of web services development. The latest version is 3.0 which was released in June 2020. This specification also provides necessary support to create REST clients.

### 7. What are HTTP Status codes?

These are the standard codes that refer to the predefined status of the task at the server. Following are the status codes formats available:

* 1xx - represents informational responses
* 2xx - represents successful responses
* 3xx - represents redirects
* 4xx - represents client errors
* 5xx - represents server errors

Most commonly used status codes are:

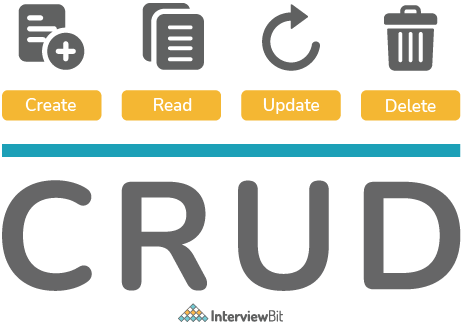
* 200 - success/OK
* 201 - CREATED - used in POST or PUT methods.
* 304 - NOT MODIFIED - used in conditional GET requests to reduce the bandwidth use of the network. Here, the body of the response sent should be empty.
* 400 - BAD REQUEST - This can be due to validation errors or missing input data.
* 401- UNAUTHORIZED - This is returned when there is no valid authentication credentials sent along with the request.
* 403 - FORBIDDEN - sent when the user does not have access (or is forbidden) to the resource.
* 404 - NOT FOUND - Resource method is not available.
* 500 - INTERNAL SERVER ERROR - server threw some exceptions while running the method.
* 502 - BAD GATEWAY - Server was not able to get the response from another upstream server.

### 8. What are the HTTP Methods?

HTTP Methods are also known as HTTP Verbs. They form a major portion of uniform interface restriction followed by the REST that specifies what action has to be followed to get the requested resource. Below are some examples of HTTP Methods:

* GET: This is used for fetching details from the server and is basically a read-only operation.
* POST: This method is used for the creation of new resources on the server.
* PUT: This method is used to update the old/existing resource on the server or to replace the resource.
* DELETE: This method is used to delete the resource on the server.
* PATCH: This is used for modifying the resource on the server.
* OPTIONS: This fetches the list of supported options of resources present on the server.

The POST, GET, PUT, DELETE corresponds to the create, read, update, delete operations which are most commonly called **CRUD Operations**.



GET, HEAD, OPTIONS are safe and idempotent methods whereas PUT and DELETE methods are only idempotent. POST and PATCH methods are neither safe nor idempotent.

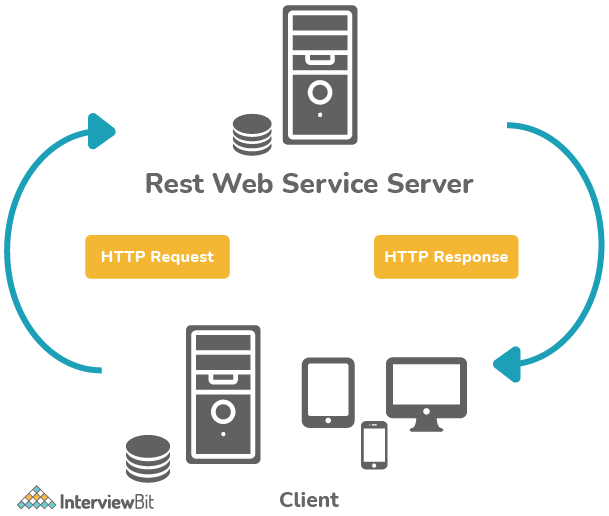
### 9. Can you tell the disadvantages of RESTful web services?

The disadvantages are:

* As the services follow the idea of statelessness, it is not possible to maintain sessions. (Session simulation responsibility lies on the client-side to pass the session id)
* REST does not impose security restrictions inherently. It inherits the security measures of the protocols implementing it. Hence, care must be chosen to implement security measures like integrating SSL/TLS based authentications, etc.

### 10. Define Messaging in terms of RESTful web services.

The technique of sending a message from the REST client to the REST server in the form of an HTTP request and the server responding back with the response as HTTP Response is called Messaging. The messages contained constitute the data and the metadata about the message.



## REST API Experienced Interview Questions

### 11. Differentiate between SOAP and REST?

| **SOAP** | **REST** |
| --- | --- |
| SOAP - Simple Object Access Protocol | REST - Representational State Transfer |
| SOAP is a protocol used to implement web services. | REST is an architectural design pattern for developing web services |
| SOAP cannot use REST as it is a protocol. | REST architecture can have SOAP protocol as part of the implementation. |
| SOAP specifies standards that are meant to be followed strictly. | REST defines standards but they need not be strictly followed. |
| SOAP client is more tightly coupled to the server which is similar to desktop applications having strict contracts. | The REST client is more flexible like a browser and does not depend on how the server is developed unless it follows the protocols required for establishing communication. |
| SOAP supports only XML transmission between the client and the server. | REST supports data of multiple formats like XML, JSON, MIME, Text, etc. |
| SOAP reads are not cacheable. | REST read requests can be cached. |
| SOAP uses service interfaces for exposing the resource logic. | REST uses URI to expose the resource logic. |
| SOAP is slower. | REST is faster. |
| Since SOAP is a protocol, it defines its own security measures. | REST only inherits the security measures based on what protocol it uses for the implementation. |
| SOAP is not commonly preferred, but they are used in cases which require stateful data transfer and more reliability. | REST is commonly preferred by developers these days as it provides more scalability and maintainability. |

### 12. While creating URI for web services, what are the best practices that needs to be followed?

Below is the list of best practices that need to be considered with designing URI for web services:

* While defining resources, use plural nouns. Example: To identify user resource, use the name “users” for that resource.
* While using the long name for resources, use underscore or hyphen. Avoid using spaces between words. For example, to define authorized users resource, the name can be “authorized\_users” or “authorized-users”.
* The URI is case-insensitive, but as part of best practice, it is recommended to use lower case only.
* While developing URI, the backward compatibility must be maintained once it gets published. When the URI is updated, the older URI must be redirected to the new one using the HTTP status code 300.
* Use appropriate HTTP methods like GET, PUT, DELETE, PATCH, etc. It is not needed or recommended to use these method names in the URI. Example: To get user details of a particular ID, use /users/{id} instead of /getUser
* Use the technique of forward slashing to indicate the hierarchy between the resources and the collections. Example: To get the address of the user of a particular id, we can use: /users/{id}/address

### 13. What are the best practices to develop RESTful web services?

RESTful web services use REST API as means of implementation using the HTTP protocol. REST API is nothing but an application programming interface that follows REST architectural constraints such as statelessness, cacheability, maintainability, and scalability. It has become very popular among the developer community due to its simplicity. Hence, it is very important to develop safe and secure REST APIs that follow good conventions. Below are some best practices for developing REST APIs:

* Since REST supports multiple data formats, it is however good practice to develop REST APIs that accept and responds with JSON data format whenever possible. This is because a majority of the client and server technologies have inbuilt support to read and parse JSON objects with ease, thereby making JSON the standard object notation.
  + To ensure that the application responds using JSON data format, the response header should have Content-Type set to as application/JSON, this is because certain HTTP clients look at the value of this response header to parse the objects appropriately.
  + To ensure that the request sends the data in JSON format, again the Content-Type must be set to application/JSON on the request header.
* While naming the resource endpoints, ensure to use plural nouns and not verbs. The API endpoints should be clear, brief, easy to understand, and informative. Using verbs in the resource name doesn’t contribute much information because an HTTP request already has what the request is doing in its HTTP method/verb. An appropriate HTTP verb should be used to represent the task of the API endpoint.
  + Below are the most commonly used HTTP methods to define the verb:
    - GET - indicates get/retrieve the resource data
    - POST - indicates create new resource data
    - PUT - indicates update the existing resource data
    - DELETE - indicates remove the resource data
* To represent the hierarchy of resources, use the nesting in the naming convention of the endpoints. In case, you want to retrieve data of one object residing in another object, the endpoint should reflect this to communicate what is happening. For example, to get the address of an author, we can use the GET method for the URI /authors/:id/address'
  + Please ensure there are no more than 2 or 3 levels of nesting as the name of the URI can become too long and unwieldy.
* Error Handling should be done gracefully by returning appropriate error codes the application has encountered. REST has defined standard HTTP Status codes that can be sent along with the response based on the scenario.
  + Error codes should also be accompanied by appropriate error messages that can help the developers to take corrective actions. However, the message should not be too elaborate as well which can help the hacker to hack your application.
  + Common status codes are:
    - 400 - Bad Request – client-side error - failed input validation.
    - 401 - Unauthorized – The user is not authenticated and hence does not have authority to access the resource.
    - 403 - Forbidden – User is authenticated but is not authorized to access the resource.
    - 404 - Not Found – The resource is not found.
    - 500 - Internal server error – This is a very generic server-side error that is thrown when the server goes down. This shouldn’t be returned by the programmer explicitly.
    - 502 - Bad Gateway – Server did not receive a valid response from the upstream server.
    - 503 - Service Unavailable – Some unexpected things happened on the server such as system failure, overload, etc.
* While retrieving huge resource data, it is advisable to include filtering and pagination of the resources. This is because returning huge data all at once can slow down the system and reduce the application performance. Hence, filter some items reduces the data to some extent. Pagination of data is done to ensure only some results are sent at a time. Doing this can increase the server performance and reduce the burden of the server resources.
* Good security practices are a must while developing REST APIs. The client-server communication must be private due to the nature of data sensitivity. Hence, incorporating SSL/TLS becomes the most important step while developing APIs as they facilitate establishing secure communication. SSL certificates are easier to get and load on the server.
  + Apart from the secure channels, we need to ensure that not everyone should be able to access the resource. For example, normal users should not access the data of admins or another user. Hence, role-based access controls should be in place to make sure only the right set of users can access the right set of data.
* Since REST supports the feature of caching, we can use this feature to cache the data in order to improve the application performance. Caching is done to avoid querying the database for a request repeated times. Caching makes data retrieval fast. However, care must be taken to ensure that the cache has updated data and not outdated ones. Frequent cache update measures need to be incorporated. There are many cache providers like Redis that can assist in caching.
* API Versioning: Versioning needs to be done in case we are planning to make any changes with the existing endpoints. We do not want to break communication between our application and the apps that consume our application while we are working on the API release. The transition has to be seamless. Semantic versioning can be followed. For example, 3.0.1 represents 3rd major version with the first patch. Usually, in the API endpoints, we define /v1,/v2, etc at the beginning of the API path.

### 14. What are Idempotent methods? How is it relevant in RESTful web services domain?

The meaning of idempotent is that even after calling a single request multiple times, the outcome of the request should be the same. While designing REST APIs, we need to keep in mind to develop idempotent APIs. This is because the consumers can write client-side code which can result in duplicate requests intentionally or not. Hence, fault-tolerant APIs need to be designed so that they do not result in erroneous responses.

* Idempotent methods ensure that the responses to a request if called once or ten times or more than that remain the same. This is equivalent to adding any number with 0.
* REST provides idempotent methods automatically. GET, PUT, DELETE, HEAD, OPTIONS, and TRACE are the idempotent HTTP methods. POST is not idempotent.
  + POST is not idempotent because POST APIs are usually used for creating a new resource on the server. While calling POST methods N times, there will be N new resources. This does not result in the same outcome at a time.
  + Methods like GET, OPTIONS, TRACE, and HEAD are idempotent because they do not change the state of resources on the server. They are meant for resource retrieval whenever called. They do not result in write operations on the server thereby making it idempotent.
  + PUT methods are generally used for updating the state of resources. If you call PUT methods N times, the first request updates the resource and the subsequent requests will be overwriting the same resource again and again without changing anything. Hence, PUT methods are idempotent.
  + DELETE methods are said to be idempotent because when calling them for N times, the first request results in successful deletion (Status Code 200), and the next subsequent requests result in nothing - Status Code 204. The response is different, but there is no change of resources on the server-side.
    - However, if you are attempting to delete the resource present, at last, every time you hit the API, such as the request DELETE /user/last which deletes the last user record, then calling the request N times would delete N resources on the server. This does not make DELETE idempotent. In such cases, as part of good practices, it is advisable to use POST requests.

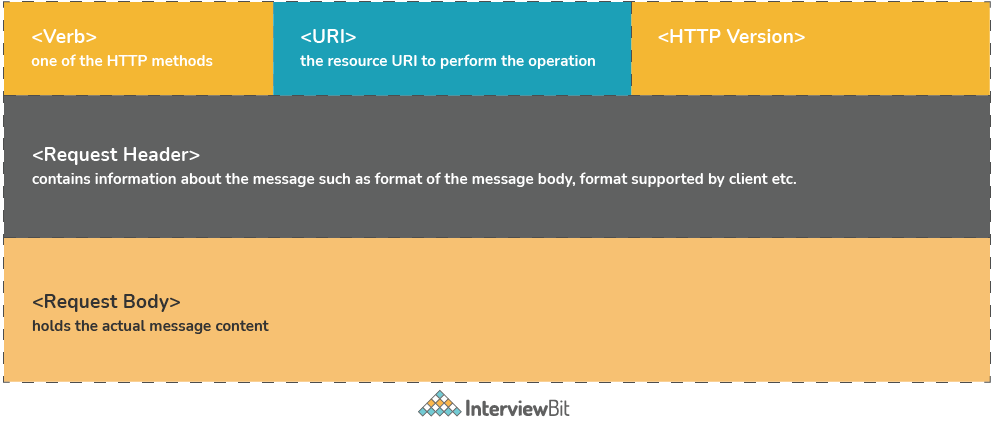
### 15. What are the differences between REST and AJAX?

| **REST** | **AJAX** |
| --- | --- |
| REST- Representational State Transfer | AJAX - Asynchronous javascript and XML |
| REST has a URI for accessing resources by means of a request-response pattern. | AJAX uses XMLHttpRequest object to send requests to the server and the response is interpreted by the Javascript code dynamically. |
| REST is an architectural pattern for developing client-server communication systems. | AJAX is used for dynamic updation of UI without the need to reload the page. |
| REST requires the interaction between client and server. | AJAX supports asynchronous requests thereby eliminating the necessity of constant client-server interaction. |

### 16. Can you tell what constitutes the core components of HTTP Request?

In REST, any HTTP Request has 5 main components, they are:

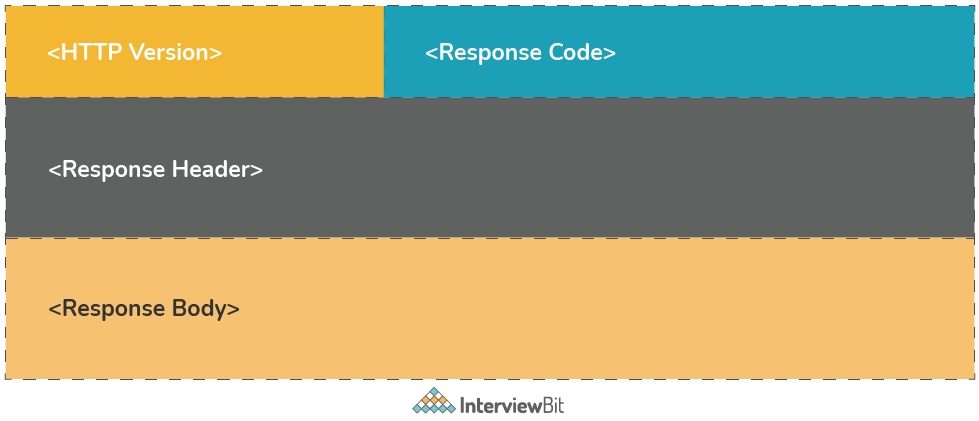
* Method/Verb − This part tells what methods the request operation represents. Methods like GET, PUT, POST, DELETE, etc are some examples.
* URI − This part is used for uniquely identifying the resources on the server.
* HTTP Version − This part indicates what version of HTTP protocol you are using. An example can be HTTP v1.1.
* Request Header − This part has the details of the request metadata such as client type, the content format supported, message format, cache settings, etc.
* Request Body − This part represents the actual message content to be sent to the server.



### 17. What constitutes the core components of HTTP Response?

HTTP Response has 4 components:

* Response Status Code − This represents the server response status code for the requested resource. Example- 400 represents a client-side error, 200 represents a successful response.
* HTTP Version − Indicates the HTTP protocol version.
* Response Header − This part has the metadata of the response message. Data can describe what is the content length, content type, response date, what is server type, etc.
* Response Body − This part contains what is the actual resource/message returned from the server.



### 18. Define Addressing in terms of RESTful Web Services.

Addressing is the process of locating a single/multiple resources that are present on the server. This task is accomplished by making use of URI (Uniform Resource Identifier). The general format of URI is

<protocol>://<application-name>/<type-of-resource>/<id-of-resource>

### 19. What are the differences between PUT and POST in REST?

| **PUT** | **POST** |
| --- | --- |
| PUT methods are used to request the server to store the enclosed entity in request. In case, the request does not exist, then new resource has to be created. If the resource exists, then the resource should get updated. | POST method is used to request the server to store the enclosed entity in the request as a new resource. |
| The URI should have a resource identifier. Example: PUT /users/{user-id} | The POST URI should indicate the collection of the resource. Example: POST /users |
| PUT methods are idempotent. | POST methods are not idempotent. |
| PUT is used when the client wants to modify a single resource that is part of the collection. If a part of the resource has to be updated, then PATCH needs to be used. | POST methods are used to add a new resource to the collection. |
| The responses are not cached here despite the idempotency. | Responses are not cacheable unless the response explicitly specifies Cache-Control fields in the header. |
| In general, PUT is used for UPDATE operations. | POST is used for CREATE operations. |

### 20. What makes REST services to be easily scalable?

REST services follow the concept of statelessness which essentially means no storing of any data across the requests on the server. This makes it easier to scale horizontally because the servers need not communicate much with each other while serving requests.

### 21. Based on what factors, you can decide which type of web services you need to use - SOAP or REST?

REST services have gained popularity due to the nature of simplicity, scalability, faster speed, improved performance, and multiple data format support. But, SOAP has its own advantages too. Developers use SOAP where the services require advanced security and reliability.

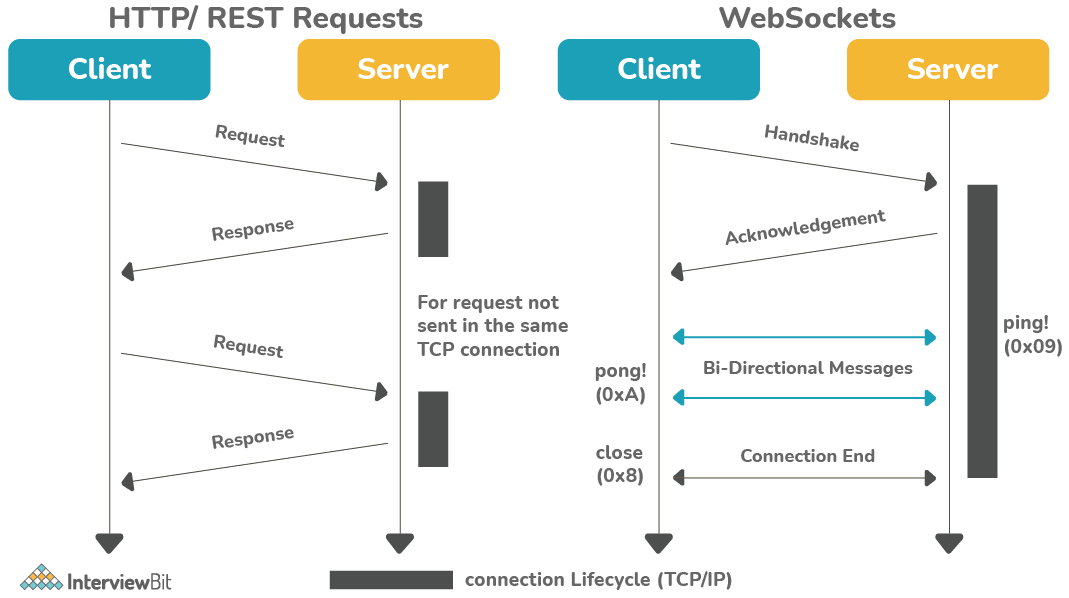
Following are the questions you need to ask to help you decide which service can be used:

* Do you want to expose resource data or business logic?
  + SOAP is commonly used for exposing business logic and REST for exposing data.
* Does the client require a formal strict contract?
  + If yes, SOAP provides strict contracts by using WSDL. Hence, SOAP is preferred here.
* Does your service require support for multiple formats of data?
  + If yes, REST supports multiple data formats which is why it is preferred in this case.
* Does your service require AJAX call support?
  + If yes, REST can be used as it provides the XMLHttpRequest.
* Does your service require both synchronous and asynchronous requests?
  + SOAP has support for both sync/async operations.
  + REST only supports synchronous calls.
* Does your service require statelessness?
  + If yes, REST is suitable. If no, SOAP is preferred.
* Does your service require a high-security level?
  + If yes, SOAP is preferred. REST inherits the security property based on the underlying implementation of the protocol. Hence, it can’t be preferred at all times.
* Does your service require support for transactions?
  + If yes, SOAP is preferred as it is good in providing advanced support for transaction management.
* What is the bandwidth/resource required?
  + SOAP involves a lot of overhead while sending and receiving XML data, hence it consumes a lot of bandwidth.
  + REST makes use of less bandwidth for data transmission.
* Do you want services that are easy to develop, test, and maintain frequently?
  + REST is known for simplicity, hence it is preferred.

### 22. We can develop webservices using web sockets as well as REST. What are the differences between these two?

| **REST** | **Web Socket** |
| --- | --- |
| REST follows stateless architecture, meaning it won’t store any session-based data. | Web Socket APIs follow the stateful protocol as it necessitates session-based data storage. |
| The mode of communication is uni-directional. At a time, only the server or the client will communicate. | The communication is bi-directional, communication can be done by both client or server at a time. |
| REST is based on the Request-Response Model. | Web Socket follows the full-duplex model. |
| Every request will have sections like header, title, body, URL, etc. | Web sockets do not have any overhead and hence suited for real-time communication. |
| For every HTTP request, a new TCP connection is set up. | There will be only one TCP connection and then the client and server can start communicating. |
| REST web services support both vertical and horizontal scaling. | Web socket-based services only support vertical scaling. |
| REST depends on HTTP methods to get the response. | Web Sockets depend on the IP address and port number of the system to get a response. |
| Communication is slower here. | Message transmission happens very faster than REST API. |
| Memory/Buffers are not needed to store data here. | Memory is required to store data. |

The request flow difference between the REST and Web Socket is shown below:



### 23. Can we implement transport layer security (TLS) in REST?

Yes, we can. TLS does the task of encrypting the communication between the REST client and the server and provides the means to authenticate the server to the client. It is used for secure communication as it is the successor of the Secure Socket Layer (SSL). HTTPS works well with both TLS and SSL thereby making it effective while implementing RESTful web services. One point to mention here is, the REST inherits the property of the protocol it implements. So security measures are dependent on the protocol REST implements.

### 24. Should we make the resources thread safe explicitly if they are made to share across multiple clients?

There is no need to explicitly making the resources thread-safe because, upon every request, new resource instances are created which makes them thread-safe by default.

### 25. What is Payload in terms of RESTful web services?

Payload refers to the data passes in the request body. It is not the same as the request parameters. The payload can be sent only in POST methods as part of the request body.

### 26. Is it possible to send payload in the GET and DELETE methods?

No, the payload is not the same as the request parameters. Hence, it is not possible to send payload data in these methods.

### 27. How can you test RESTful Web Services?

RESTful web services can be tested using various tools like Postman, Swagger, etc. Postman provides a lot of features like sending requests to endpoints and show the response which can be converted to JSON or XML and also provides features to inspect request parameters like headers, query parameters, and also the response headers. Swagger also provides similar features like Postman and it provides the facility of documentation of the endpoints too. We can also use tools like Jmeter for performance and load testing of APIs.

### 28. What is the maximum payload size that can be sent in POST methods?

Theoretically, there is no restriction on the size of the payload that can be sent. But one must remember that the greater the size of the payload, the larger would be the bandwidth consumption and time taken to process the request that can impact the server performance.

### 29. How does HTTP Basic Authentication work?

While implementing Basic Authentication as part of APIs, the user must provide the username and password which is then concatenated by the browser in the form of “username: password” and then perform base64 encoding on it. The encoded value is then sent as the value for the “Authorization” header on every HTTP request from the browser. Since the credentials are only encoded, it is advised to use this form when requests are sent over HTTPS as they are not secure and can be intercepted by anyone if secure protocols are not used.

### 30. What is the difference between idempotent and safe HTTP methods?

* Safe methods are those that do not change any resources internally. These methods can be cached and can be retrieved without any effects on the resource.
* Idempotent methods are those methods that do not change the responses to the resources externally. They can be called multiple times without any change in the responses.

According to [restcookbook.com](https://restcookbook.com/), the following is the table that describes what methods are idempotent and what is safe.

| **HTTP Methods** | **Idempotent** | **Safe** |
| --- | --- | --- |
| OPTIONS | yes | yes |
| GET | yes | yes |
| HEAD | yes | yes |
| PUT | yes | no |
| POST | no | no |
| DELETE | yes | no |
| PATCH | no | no |

## JAX-RS Interview Questions

### 31. What are the key features provided by JAX-RS API in Java EE?

JAX-RS stands for Java API for RESTful Web services. They are nothing but a set of Java-based APIs that are provided in the Java EE which is useful in the implementation and development of RESTful web services.

Features of JAX-RS are:

* **POJO-based**: The APIs in the JAX-RS is based on a certain set of annotations, classes, and interfaces that are used with POJO (Plain Old Java Object) to expose the services as web services.
* **HTTP-based:** The JAX-RS APIs are designed using HTTP as their base protocol. They support the HTTP usage patterns and they provide the corresponding mapping between the HTTP actions and the API classes.
* **Format Independent**: They can be used to work with a wide range of data types that are supported by the HTTP body content.
* **Container Independent**: The APIs can be deployed in the Java EE container or a servlet container such as Tomcat or they can also be plugged into JAX-WS (Java API for XML-based web services) providers.

### 32. Define RESTful Root Resource Classes in the JAX-RS API?

* A resource class is nothing but a Java class that uses JAX-RS provided annotations for implementing web resources.
* They are the POJOs that are annotated either with @Path or have at least one method annotated with @Path, @GET, @POST, @DELETE, @PUT, etc.

Example:

**import** javax.ws.rs.Path;

/\*\*

\* InterviewBitService is a root resource class that is exposed at 'resource\_service' path

\*/

@Path('resource\_service')

**public** **class** **InterviewBitService** {

// Defined methods

}

### 33. What do you understand by request method designator annotations?

They are the runtime annotations in the JAX-RS library that are applied to Java methods. They correspond to the HTTP request methods that the clients want to make. They are @GET, @POST, @PUT, @DELETE, @HEAD.

Usage Example:

**import** javax.ws.rs.Path;

/\*\*

\* InterviewBitService is a root resource class that is exposed at 'resource\_service' path

\*/

@Path('resource\_service')

**public** **class** **InterviewBitService** {

@GET

**public** String **getRESTQuestions**() {

// some operations

}

}

### 34. How can the JAX-RS applications be configured?

JAX-RS applications have the root resource classes packaged in a war file. There are 2 means of configuring JAX-RS applications.

1. Use @ApplicationPath annotation in a subclass of javax.ws.rs.core.Application that is packaged in the WAR file.
2. Use the <servlet-mapping> tag inside the web.xml of the WAR. web.xml is the deployment descriptor of the application where the mappings to the servlets can be defined.

### 35. Is it possible to make asynchronous requests in JAX-RS?

Yes. the JAX-RS Client API provides a method called Invocation.Builder.async() that is used for constructing client requests that need to be executed asynchronously. Invoking a request asynchronously does the task of returning the control to the caller by returning with datatype java.util.concurrent.Future whose type is set to return the service call type. Future objects are used because they have the required methods to check whether the asynchronous calls have been completed and if yes, then retrieve the responses. They also provide the flexibility to cancel the request invocations and also check if the cancellation has been successful.

Let us understand this with the help of a random example. We know that the Future interface from the java.util.concurrent has the below functions available:

**package** java.util.concurrent;

**public** **interface** **Future**<**V**> {

// informs the executor to stop the thread execution

**boolean** **cancel**(**boolean** mayInterruptIfRunning);

// indicates whether the Future was cancelled or not

**boolean** **isCancelled**();

// indicates if the executor has completed the task

**boolean** **isDone**();

// gets the actual result from the process.

// This blocks the program execution until the result is ready.

V **get**() **throws** InterruptedException, ExecutionException;

// also gets actual result from the process but it throws

// the TimeoutException in case the result is not obtained before specified timeout

V **get**(**long** timeout, TimeUnit unit)

**throws** InterruptedException, ExecutionException, TimeoutException;

}

Let us consider we have this function below which is used for processing 2 Ids parallelly.

**public** **void** **processIds**(String userId1, String questionId){

Client client = ClientBuilder.newClient();

Future<Response> futureResponse1 = client.target("http://interviewbitserver.com/users/"+userId).request().async().get();

Future<Order> futureResponse2 = client.target("http://interviewbitserver.com/questions/"+questionId).request().async().get(Question.class);

// block the process until complete

Response response1 = futureResponse1.get();

User userObject = response1.readEntity(User.class);

//Do processing of userObject

// Wait for 2 seconds before fetching record

**try** {

Question question = futureResponse2.get(2, TimeUnit.SECONDS);

//Do Processing of question

} **catch** (TimeoutException timeoutException ) {

//handle exceptions

}

**return**;

}

In the above example, we see that there are 2 separate requests getting executed parallelly. For the first future object, we await the javax.ws.rs.core.Response indefinitely using the get() method until we get the response. For the second future object, we wait for the response only for 2 seconds and if we do not get within 2 seconds, then the get() method throws TimeoutException. We can also use the isDone() method or isCancelled() method to find out whether the executors have completed or cancelled.

### 36. List the key annotations that are present in the JAX-RS API?

* @Path - This specifies the relative URI path to the REST resource.
* @GET - This is a request method designator which is corresponding to the HTTP GET requests. They process GET requests.
* @POST - This is a request method designator which is corresponding to the HTTP POST requests. They process POST requests.
* @PUT - This is a request method designator which is corresponding to the HTTP PUT requests. They process PUT requests.
* @DELETE - This is a request method designator which is corresponding to the HTTP DELETE requests. They process DELETE requests.
* @HEAD - This is a request method designator which is corresponding to the HTTP HEAD requests. They process HEAD requests.
* @PathParam - This is the URI path parameter that helps developers to extract the parameters from the URI and use them in the resource class/methods.
* @QueryParam - This is the URI query parameter that helps developers extract the query parameters from the URI and use them in the resource class/methods.
* @Produces - This specifies what MIME media types of the resource representations are produced and sent to the client as a response.
* @Consumes - This specifies which MIME media types of the resource representations are accepted or consumed by the server from the client.

## Spring RESTful Web Services Interview Questions

### 37. Define RestTemplate in Spring.

The RestTemplate is the main class meant for the client-side access for Spring-based RESTful services. The communication to the server is accomplished using the REST constraints. This is similar to other template classes such as JdbcTemplate, HibernateTemplate, etc provided by Spring. The RestTemplate provides high-level implementation details for the HTTP Methods like GET, POST, PUT, etc, and gives the methods to communicate using the URI template, URI path params, request/response types, request object, etc as part of arguments.

* Commonly used annotations like @GetMapping, @PostMapping, @PutMapping, etc are provided by this class from Spring 4.3. Prior to that, Spring provided (and still provides) @RequestMapping annotation to indicate what methods were being used.

### 38. What is the use of @RequestMapping?

* The annotation is used for mapping requests to specific handler classes or methods.
* In spring, all the incoming web request routing is handled by Dispatcher Servlet. When it gets the request, it determines which controller is meant for processing the request by means of request handlers. The Dispatcher Servlet scans all the classes annotated with @Controller. The process of routing requests depends on @RequestMapping annotations that are declared inside the controller classes and their methods.

### 39. What are the differences between the annotations @Controller and @RestController?

| **@Controller** | **@RestController** |
| --- | --- |
| Mostly used traditional Spring MVC service. | Represents RESTful web service in Spring. |
| It is mostly used in Spring MVC service where model data needs to rendered using view. | It is used in case of RESTful web service that returns object values bound to response body. |
| If response values need to be converted through HttpMessageConverters and sent via response object, extra annotation @ResponseBody needs to be used on the class or the method handlers. | The default behavior of the @RestController needs to be written on the response body because it is the combination of @Controller and @ResponseBody. |
| @Controller provides control and flexibility over how the response needs to be sent. | @RestController annotation has no such flexibility and writes all the results to the response body. |

### 40. What does the annotation @PathVariable do?

@PathVariable annotation is used for passing the parameter with the URL that is required to get the data. Spring MVC provides support for URL customization for data retrieval using @PathVariable annotation.

### 41. Is it necessary to keep Spring MVC in the classpath for developing RESTful web services?

Yes. [**Spring MVC**](https://www.interviewbit.com/spring-interview-questions/) needs to be on the classpath of the application while developing RESTful web services using Spring. This is because, the Spring MVC provides the necessary annotations like @RestController, @RequestBody, @PathVariable, etc. Hence the spring-mvc.jar needs to be on the classpath or the corresponding Maven entry in the pom.xml.

### 42. Define HttpMessageConverter in terms of Spring REST?

HttpMessageConverter is a strategic interface that specified a converter for conversion between HTTP Requests and responses. Spring REST uses the HttpMessageConverter for converting responses to various data formats like JSON, XML, etc. Spring makes use of the “Accept” header for determining the type of content the client expects. Based on this, Spring would find the registered message converter interface that is capable of this conversion.

## Conclusion

### 43. Conclusion

We have seen what are the most commonly asked questions on RESTful web services during an interview. REST APIs have become a very important tool in the software industry. Developing RESTful web services that are scalable and easily maintainable is considered an art. As the industry trends increase, the REST architecture would become more concrete and the demand for developers who know the development of RESTful web services would increase steadily.

# 2.Guru99: 15Q

**1) Explain what is REST and RESTFUL?**

REST represents REpresentational State Transfer; it is a relatively new aspect of writing web API.

RESTFUL is referred for web services written by applying REST architectural concept are called RESTful services, it focuses on system resources and how state of resource should be transported over HTTP protocol to different clients written in different language. In RESTFUL web service HTTP methods like GET, POST, PUT and DELETE can be used to perform CRUD operations.

**2) Explain the architectural style for creating web API?**

The architectural style for creating web api are

* HTTP for client server communication
* XML/JSON as formatting language
* Simple URI as the address for the services
* Stateless communication

**3) Mention what tools are required to test your web API?**

SOAPUI tool for SOAP WS and Firefox “poster” plugin for RESTFUL services.

**4) Mention what are the HTTP methods supported by REST?**

HTTP methods supported by REST are:

* **GET:**It requests a resource at the request URL. It should not contain a request body as it will be discarded. Maybe it can be cached locally or on the server.
* **POST:**It submits information to the service for processing; it should typically return the modified or new resource
* **PUT:**At the request URL it update the resource
* **DELETE:**At the request URL it removes the resource
* **OPTIONS:**It indicates which techniques are supported
* **HEAD:**About the request URL it returns meta information

**5) Mention whether you can use GET request instead of PUT to create a resource?**

No, you are not supposed to use PUT for GET. GET operations should only have view rights, while PUT resource is used for updating a data.

**6) Mention what are resources in a REST architecture?**

Resources are identified by logical URLs; it is the key element of a RESTful design. Unlike, SOAP web services in REST, you view the product data as a resource and this resource should contain all the required information.

**7) Mention what is the difference between AJAX and REST?**

| **AJAX** | **REST** |
| --- | --- |
| * In Ajax, the request are sent to the server by using XMLHttpRequest objects. The response is used by the JavaScript code to dynamically alter the current page. | * REST have a URL structure and a request/response pattern the revolve around the use of resources. |
| * Ajax is a set of technology; it is a technique of dynamically updating parts of UI without having to reload the page. | * REST is a type of software architecture and a method for users to request data or information from servers. |
| * Ajax eliminates the interaction between the customer and server asynchronously. | * REST requires the interaction between the customer and server. |

**7) Mention some key characteristics of REST?**

Some key characteristics of REST includes

* REST is stateless, therefore the SERVER has no state (or session data)
* With a well-applied REST API, the server could be restarted between two calls as every data is passed to the server
* Web service mostly uses POST method to make operations, whereas REST uses GET to access resources

**8) Mention what are the different application integration styles?**

The different integration styles include

* Shared database
* Batch file transfer
* Invoking remote procedure (RPC)
* Swapping asynchronous messages over a message oriented middle-ware (MOM)

**9) Explain how JAXB related to RESTful web API?**

JAXB stands for java arch for XML binding.

**10) Mention what is the difference between PUT and POST?**

“PUT” puts a file or resource at a particular URI and exactly at that URI. If there is already a file or resource at that URI, PUT changes that file or resource. If there is no resource or file there, PUT makes one

POST sends data to a particular URI and expects the resource at that URI to deal with the request. The web server at this point can decide what to do with the data in the context of specified resource

PUT is idempotent meaning, invoking it any number of times will not have an impact on resources.

However, POST is not idempotent, meaning if you invoke POST multiple times it keeps creating more resources

**11) Mention which markup language can be used in restful web api?**

JSON and XML are the two markup language that can be used in restful web api

**12) Mention what is the difference between RPC or document style web services? How you determine to which one to choose?**

In document style web services, we can transport an XML message as part of SOAP request which is not possible in RPC style web service. Document style web service is most appropriate in some application where XML message behaves as document and content of that document can alter and intention of web service does not rely on the content of XML message.

**13) Mention what is JAX-WS and JAX-RS?**

Both JAX-WS and JAX-RS are libraries (APIs) for doing communication in various ways in Java. JAX-WS is a library that can be used to do SOAP communication in JAVA, and JAX-RS lets you do the REST communication in JAVA.

**14) List out the tools or API for developing or testing web api?**

Testing tools for web services for REST APIs includes

* Spring REST web service using MVC
* Jersey API
* CXF
* Axis
* Restlet,

**15) Mention what is the difference between SOAP and REST?**

| **SOAP** | **REST** |
| --- | --- |
| * SOAP is a protocol through which two computer communicates by sharing XML document. | * Rest is a service architecture and design for network-based software architectures. |
| * SOAP permits only XML | * REST supports many different data formats |
| * SOAP based reads cannot be cached | * REST reads can be cached |
| * SOAP is like custom desktop application, closely connected to the server | * A REST client is more like a browser; it knows how to standardized methods and an application has to fit inside it |
| * SOAP is slower than REST | * REST is faster than SOAP |
| * It runs on HTTP but envelopes the message | * It uses the HTTP headers to hold meta information |

# Educative: 13Q

### 1. What is REST?

REST stands for Representational State Transfer. REST is an architectural style for web development. REST architecture lays out guidelines for the transfer of resource representations between clients and servers on the web.

### 2. What is a REST API?

A REST API or RESTful API is a web API that conforms to the REST architecture style.

### 3. Describe the 5 constraints of the REST architectural style, and their benefits.

In case they ask for 6 constraints, see the following question about the optional constraint.

A truly RESTful API must conform to the five REST architectural constraints:

* **Uniform interface**:
  + Interface between client and server that allows for standardized client-server communication in a single language
  + Necessary for the decoupling of client and server
* **Client-server**:
  + Client-server model, for separation of concerns between client and server
  + Permits client and server to operate and evolve independently
  + Supports portability and scalability
* **Stateless**:
  + Refers to stateless communication protocol, wherein the server stores no information about session states
  + Improves performance by reducing server load
* **Cacheable**:
  + Servers mark their responses as cacheable or non-cacheable
  + Clients and intermediaries are able to cache server responses
  + Reduces client-server interaction, supports scalability and performance
* **Layered system**:
  + Layers between client and server, can consist of intermediaries such as proxy servers or load balancers
  + Layers have separate responsibilities but are able to interact with each other
  + Supports system scalability and security

### 4. What is the optional architectural constraint of REST?

**Code on demand** is the optional constraint of RESTful architecture. Code on demand allows the server to send executable code (scripts or applets) to a client upon client request.

**Advantage**: Extends client functionality, since client can download features after deployment  
**Disadvantage**: Reduces visibility, which is why it’s considered optional  
**Examples**: Java applets and JavaScript

### 5. Explain the constraints of a uniform interface.

A uniform interface is needed to decouple the client from the server.

There are four necessary constraints to achieving uniform interface:

* **Identification of resources**: Client requests must identify resources using uniform resource identifiers (URIs)
* **Manipulation of resources through these representations**: When clients receive a resource representation from the server, they have all information necessary to be able to modify resource state
* **Self-descriptive messages**: Messages contain all information necessary for recipient to interpret it, including metadata
* **Hypermedia as the engine of application state**: Hypermedia (such as HTML) is the medium for client-server interaction, and the client requires no API-specific documentation to understand server responses

### 6. What is CRUD?

[CRUD](https://www.educative.io/blog/crud-operations) is an acronym for the four basic operations used in relational database management system (RDBMS).

Each operation in CRUD relates to an HTTP method that REST supports.

* **Create**: POST
* **Read**: GET
* **Update**: PUT
* **Delete**: DELETE

#### Get hands-on with interview prep.

Join the 800,000 developers who advanced their careers with Educative.

[Get started](https://learn.educative.io/interview-prep)

### 7. Explain the HTTP request methods supported by REST, and when they are used.

REST APIs are based on [HTTP requests](https://www.educative.io/blog/behind-the-screens-what-happens-when-you-type-a-url-in-a-browser#http) or verbs, which each perform a different task.

REST supports the following HTTP requests:

* **GET** method: Request data from server
* **POST** method: Submit data to create new resource on server-defined URL
* **PUT** method: Submit data to create new resource at client-defined URL
* **DELETE** method: Remove resource from server
* **OPTIONS** method: Return request methods supported by a service
* **HEAD** method: Return meta information such as response headers
* **PATCH** method: Modify part of the resource on the server

### 8. What’s the difference between PUT and POST methods?

This question can stump some developers. Being able to explain this will help you stand out as someone who actually knows what they’re talking about.

Here are the differences between PUT and POST:

**PUT**:

* Idempotent (i.e. multiple requests will yield same result)
* PUT responses aren’t cacheable
* Updates or replaces target resource with request’s payload

**POST**:

* Not idempotent (i.e. multiple requests will yield multiples of the same resource)
* POST responses can be cacheable, provided proper cache-control header
* Request’s payload is processed by the web server based on target resource

**Understanding idempotency**: An example of an idempotent operation would be the operation of multiplying a number by one. No matter how many times you multiply five by one, you’ll get the same result.

### 9. Explain what statelessness means in REST.

Statelessness means that the client and server don’t store information about each other’s state. Since the server stores no information, it treats each client request as a new request.

As a consequence, the following conditions would apply:

* The client request contains all information required for the server to process the request
* Client application is responsible for storing session state

### 10. What are the advantages and disadvantages of a REST API?

It’s important to know the pros and cons of a RESTful API.

**Advantages include**:

* Designed for high performance, portability, reliability, and scalability
* Client-server separation allows each to individually operate and scale
* Easy to test and adapt to various environments
* Easy to learn as it uses HTTP protocol
* Supports various data transfer technologies including JSON, XML, YAML, images, and more
* Uses less bandwidth than other methods, such as Simple Object Access Protocol (SOAP) technology

**Disadvantages include**:

* Doesn’t enforce security practices
* HTTP method limits you to synchronous requests
* Due to statelessness, you might be unable to maintain state (e.g. in sessions)

### 11. What’s the difference between AJAX and REST?

The distinction can confuse beginner developers, so it’s helpful to know the difference.

An AJAX client can make a RESTful request to a REST API (e.g. a get request), but **AJAX isn’t an architectural style**. It’s a web development technique for client-side applications. REST APIs can be accessed by AJAX clients, but they aren’t inherently implemented with AJAX.

### 12. What’s the difference between SOAP and REST?

Although some REST APIs use SOAP protocols, REST and SOAP are entirely different approaches to building APIs. Interviewers may ask this to assess your depth of understanding.

Here are some of the differences between SOAP and REST.

**SOAP**:

* Protocol
* Data format is limited to XML
* Heavyweight and requires more bandwidth
* Calls can’t be cached

**REST**:

* Architectural style
* Allows various data formats including plain text, HTML, XML, JSON, and YAML
* Lightweight and requires less bandwidth
* Calls can be cached

### 13. Explain HTTP response status codes.

HTTP response codes indicate the result of client requests.

Common HTTP status codes include:

* **200**: Successful request
* **201**: Entity or entities created from successful request
* **400**: Bad request. Invalid client request.
* **401**: Unauthorized. User isn’t authorized to access a resource and may be unauthenticated
* **403**: Forbidden. User isn’t authorized to access a resource, user is authenticated
* **404**: Not found. Resource not found
* **500**: Internal server error. Generic server error
* **502**: Bad gateway. Response from upstream server is not valid
* **503**: Service unavailable. Result of server-side issue, including overload or system failure

# FullStackCafe: 21Q

## *Q1*:

## What is REST Web Services?

**Entry**

**[REST & RESTful](https://www.fullstack.cafe/interview-questions/rest-and-restful" \o "REST & RESTful Interview Questions)**[27](https://www.fullstack.cafe/interview-questions/rest-and-restful" \o "REST & RESTful Interview Questions)

Answer

REST is the acronym for REpresentational State Transfer. REST is an architectural style for developing applications that can be accessed over the network. REST architectural style was brought in light by Roy Fielding in his doctoral thesis in 2000.

REST is a stateless client-server architecture where web services are resources and can be identified by their URIs. Client applications can use HTTP GET/POST methods to invoke Restful web services. REST doesn’t specify any specific protocol to use, but in almost all cases it’s used over HTTP/HTTPS.

When compared to SOAP web services, these are lightweight and doesn’t follow any standard. We can use XML, JSON, text or any other type of data for request and response.

*Having Tech or Coding Interview?* Check 👉 [27 REST & RESTful Interview Questions](https://www.fullstack.cafe/interview-questions/rest-and-restful)

*Source:* [journaldev.com](https://www.journaldev.com/9193/web-services-interview-questions-soap-restful)

## *Q2*:

## Mention some key characteristics of REST?

**Junior**

**[API Design](https://www.fullstack.cafe/interview-questions/api-design" \o "API Design Interview Questions)**[46](https://www.fullstack.cafe/interview-questions/api-design" \o "API Design Interview Questions)

Answer

Some key characteristics of REST includes

* REST is stateless, therefore the SERVER has no state (or session data)
* With a well-applied REST API, the server could be restarted between two calls as every data is passed to the server
* Web service mostly uses POST method to make operations, whereas REST uses GET to access resources

*Having Tech or Coding Interview?* Check 👉 [46 API Design Interview Questions](https://www.fullstack.cafe/interview-questions/api-design)

*Source:* [career.guru99.com](https://career.guru99.com/rest-api-interview/)

## *Q3*:

## Mention what is the difference between AJAX and REST?

**Junior**

**[REST & RESTful](https://www.fullstack.cafe/interview-questions/rest-and-restful" \o "REST & RESTful Interview Questions)**[27](https://www.fullstack.cafe/interview-questions/rest-and-restful" \o "REST & RESTful Interview Questions)

Answer

**Ajax**

* In Ajax, the request are sent to the server by using XMLHttpRequest objects. The response is used by the JavaScript code to dynamically alter the current page
* Ajax is a set of technology; it is a technique of dynamically updating parts of UI without having to reload the page
* Ajax eliminates the interaction between the customer and server asynchronously

**REST**

* REST requires the interaction between the customer and server
* REST have a URL structure and a request/response pattern the revolve around the use of resources
* REST is a type of software architecture and a method for users to request data or information from servers
* REST requires the interaction between the customer and server

*Having Tech or Coding Interview?* Check 👉 [27 REST & RESTful Interview Questions](https://www.fullstack.cafe/interview-questions/rest-and-restful)

*Source:* [career.guru99.com](https://career.guru99.com/rest-api-interview/)

## *Q4*:

## What are advantages of REST web services?

**Junior**

**[API Design](https://www.fullstack.cafe/interview-questions/api-design" \o "API Design Interview Questions)**[46](https://www.fullstack.cafe/interview-questions/api-design" \o "API Design Interview Questions)

Answer

Some of the advantages of REST web services are:

* Learning curve is easy since it works on HTTP protocol
* Supports multiple technologies for data transfer such as text, xml, json, image etc.
* No contract defined between server and client, so loosely coupled implementation.
* REST is a lightweight protocol
* REST methods can be tested easily over browser.

*Having Tech or Coding Interview?* Check 👉 [46 API Design Interview Questions](https://www.fullstack.cafe/interview-questions/api-design)

*Source:* [journaldev.com](https://www.journaldev.com/9193/web-services-interview-questions-soap-restful)

## *Q5*:

## What is a Resource in Restful web services?

**Junior**

**[REST & RESTful](https://www.fullstack.cafe/interview-questions/rest-and-restful" \o "REST & RESTful Interview Questions)**[27](https://www.fullstack.cafe/interview-questions/rest-and-restful" \o "REST & RESTful Interview Questions)

Answer

Resource is the fundamental concept of Restful architecture. A resource is an object with:

* a type,
* relationship with other resources and
* methods that operate on it.

Resources are identified with:

* their URI,
* HTTP methods they support and
* request/response data type and format of data.

*Having Tech or Coding Interview?* Check 👉 [27 REST & RESTful Interview Questions](https://www.fullstack.cafe/interview-questions/rest-and-restful)

*Source:* [journaldev.com](https://www.journaldev.com/9193/web-services-interview-questions-soap-restful)

[8 Steps to Increase Your Dev Resume Response Rate by 90%](https://www.fullstack.cafe/blog/full-stack-resume-template)

**[Career](https://www.fullstack.cafe/interview-questions/career" \o "Career Interview Questions)**[1](https://www.fullstack.cafe/interview-questions/career" \o "Career Interview Questions)

## *Q6*:

## What is purpose of a URI in REST based webservices?

**Junior**

**[REST & RESTful](https://www.fullstack.cafe/interview-questions/rest-and-restful" \o "REST & RESTful Interview Questions)**[27](https://www.fullstack.cafe/interview-questions/rest-and-restful" \o "REST & RESTful Interview Questions)

Answer

URI stands for Uniform Resource Identifier. Each resource in REST architecture is identified by its URI. Purpose of an URI is to locate a resource(s) on the server hosting the web service.

A URI is of following format:

<protocol>://<service-name>/<ResourceType>/<ResourceID>

*Having Tech or Coding Interview?* Check 👉 [27 REST & RESTful Interview Questions](https://www.fullstack.cafe/interview-questions/rest-and-restful)

*Source:* [tutorialspoint.com](https://www.tutorialspoint.com/restful/restful_interview_questions.htm)

## *Q7*:

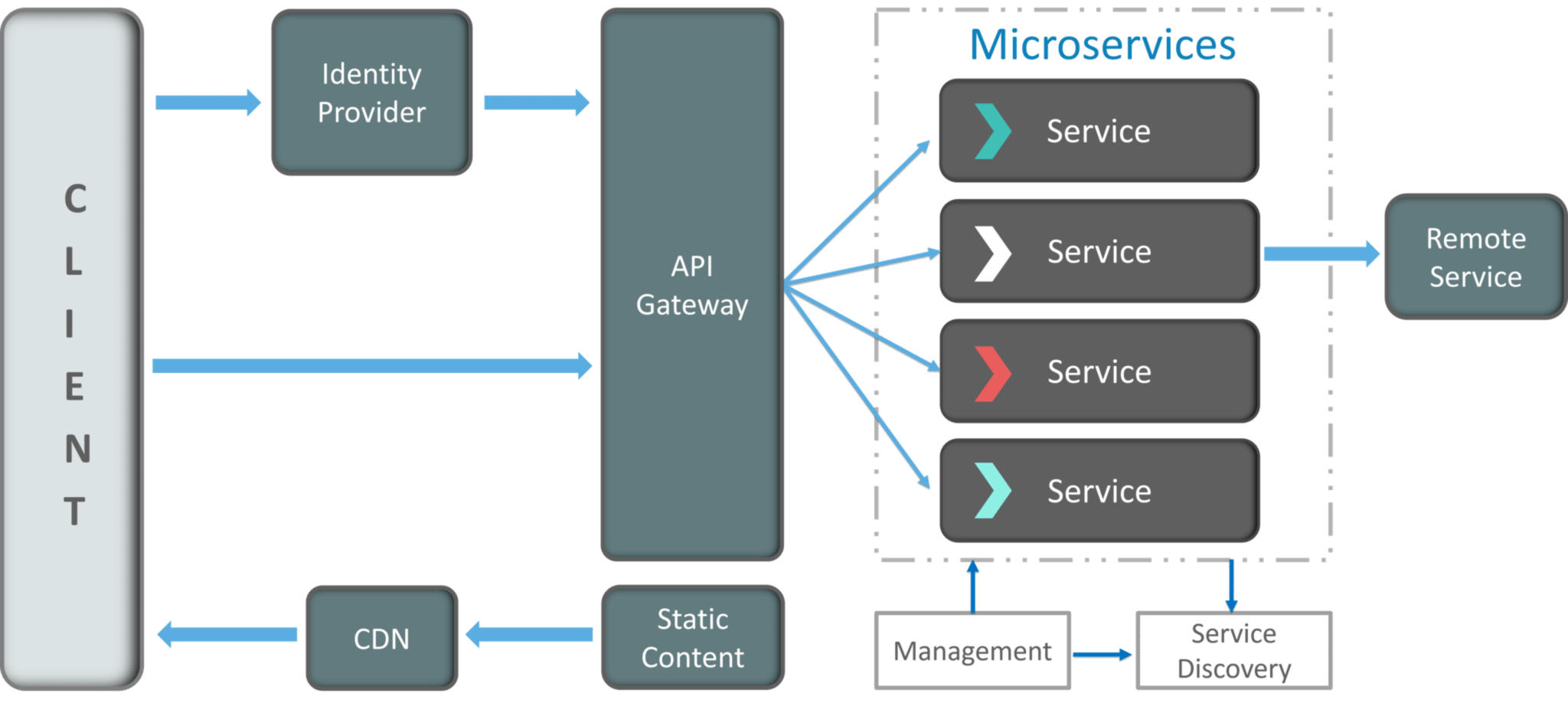
## How does Microservice Architecture work?

**Mid**

**[Microservices](https://www.fullstack.cafe/interview-questions/microservices" \o "Microservices Interview Questions)**[34](https://www.fullstack.cafe/interview-questions/microservices" \o "Microservices Interview Questions)

Answer

* **Clients** – Different users from various devices send requests.
* **Identity Providers** – Authenticates user or clients identities and issues security tokens.
* **API Gateway** – Handles client requests.
* **Static Content** – Houses all the content of the system.
* **Management** –  Balances services on nodes and identifies failures.
* **Service Discovery** – A guide to find the route of communication between microservices.
* **Content Delivery Networks** – Distributed network of proxy servers and their data centers.
* **Remote Service** – Enables the remote access information that resides on a network of IT devices.



*Having Tech or Coding Interview?* Check 👉 [34 Microservices Interview Questions](https://www.fullstack.cafe/interview-questions/microservices)

*Source:* [edureka.co](https://www.edureka.co/blog/interview-questions/microservices-interview-questions/)

## *Q8*:

## Mention what are the HTTP methods supported by REST?

**Mid**

**[REST & RESTful](https://www.fullstack.cafe/interview-questions/rest-and-restful" \o "REST & RESTful Interview Questions)**[27](https://www.fullstack.cafe/interview-questions/rest-and-restful" \o "REST & RESTful Interview Questions)

Answer

HTTP methods supported by REST are:

* GET: It requests a resource at the request URL. It should not contain a request body as it will be discarded. Maybe it can be cached locally or on the server.
* POST: It submits information to the service for processing; it should typically return the modified or new resource
* PUT: At the request URL it update the resource
* DELETE: At the request URL it removes the resource
* OPTIONS: It indicates which techniques are supported
* HEAD: About the request URL it returns meta information

*Having Tech or Coding Interview?* Check 👉 [27 REST & RESTful Interview Questions](https://www.fullstack.cafe/interview-questions/rest-and-restful)

*Source:* [career.guru99.com](https://career.guru99.com/rest-api-interview/)

 Full-Stack, Web & Mobile

 Coding & Data Structures

 System Design & Architecture

**[Ruby](https://www.fullstack.cafe/interview-questions/ruby" \o "Ruby Interview Questions)**[84](https://www.fullstack.cafe/interview-questions/ruby" \o "Ruby Interview Questions)

**[Python](https://www.fullstack.cafe/interview-questions/python" \o "Python Interview Questions)**[112](https://www.fullstack.cafe/interview-questions/python" \o "Python Interview Questions)

**[UX Design](https://www.fullstack.cafe/interview-questions/ux-design" \o "UX Design Interview Questions)**[78](https://www.fullstack.cafe/interview-questions/ux-design" \o "UX Design Interview Questions)

**[Ionic](https://www.fullstack.cafe/interview-questions/ionic" \o "Ionic Interview Questions)**[29](https://www.fullstack.cafe/interview-questions/ionic" \o "Ionic Interview Questions)

**[Spring](https://www.fullstack.cafe/interview-questions/spring" \o "Spring Interview Questions)**[87](https://www.fullstack.cafe/interview-questions/spring" \o "Spring Interview Questions)

**[ASP.NET MVC](https://www.fullstack.cafe/interview-questions/aspnet-mvc" \o "ASP.NET MVC Interview Questions)**[36](https://www.fullstack.cafe/interview-questions/aspnet-mvc" \o "ASP.NET MVC Interview Questions)

**[WPF](https://www.fullstack.cafe/interview-questions/wpf" \o "WPF Interview Questions)**[46](https://www.fullstack.cafe/interview-questions/wpf" \o "WPF Interview Questions)

**[TypeScript](https://www.fullstack.cafe/interview-questions/typescript" \o "TypeScript Interview Questions)**[39](https://www.fullstack.cafe/interview-questions/typescript" \o "TypeScript Interview Questions)

**[.NET Core](https://www.fullstack.cafe/interview-questions/net-core" \o ".NET Core Interview Questions)**[52](https://www.fullstack.cafe/interview-questions/net-core" \o ".NET Core Interview Questions)

**[Node.js](https://www.fullstack.cafe/interview-questions/nodejs" \o "Node.js Interview Questions)**[88](https://www.fullstack.cafe/interview-questions/nodejs" \o "Node.js Interview Questions)

🤖 Having Data Science & ML Interview? Check  MLStack.Cafe - 1299 Data Science & ML Interview Questions & Answers!

## *Q9*:

## Mention what is the difference between PUT and POST?

**Mid**

**[API Design](https://www.fullstack.cafe/interview-questions/api-design" \o "API Design Interview Questions)**[46](https://www.fullstack.cafe/interview-questions/api-design" \o "API Design Interview Questions)

Answer

PUT puts a file or resource at a particular URI and exactly at that URI. If there is already a file or resource at that URI, PUT changes that file or resource. If there is no resource or file there, PUT makes one

POST sends data to a particular URI and expects the resource at that URI to deal with the request. The web server at this point can decide what to do with the data in the context of specified resource

PUT is idempotent meaning, invoking it any number of times will not have an impact on resources.

However, POST is not idempotent, meaning if you invoke POST multiple times it keeps creating more resources

*Having Tech or Coding Interview?* Check 👉 [46 API Design Interview Questions](https://www.fullstack.cafe/interview-questions/api-design)

*Source:* [career.guru99.com](https://career.guru99.com/rest-api-interview/)



[18 Searching Algorithms Interview Questions (SOLVED with CODE) Devs Must Know](https://www.fullstack.cafe/blog/searching-algorithms-interview-questions)

**[Linked Lists](https://www.fullstack.cafe/interview-questions/linked-lists" \o "Linked Lists Interview Questions)**[43](https://www.fullstack.cafe/interview-questions/linked-lists" \o "Linked Lists Interview Questions)

**[Searching](https://www.fullstack.cafe/interview-questions/searching" \o "Searching Interview Questions)**[24](https://www.fullstack.cafe/interview-questions/searching" \o "Searching Interview Questions)

## *Q10*:

## What are disadvantages of REST web services?

**Mid**

**[API Design](https://www.fullstack.cafe/interview-questions/api-design" \o "API Design Interview Questions)**[46](https://www.fullstack.cafe/interview-questions/api-design" \o "API Design Interview Questions)

Answer

Some of the disadvantages of REST are:

* Since there is no contract defined between service and client, it has to be communicated through other means such as documentation or emails.
* Since it works on HTTP, there can’t be asynchronous calls.
* Sessions can’t be maintained.

*Having Tech or Coding Interview?* Check 👉 [46 API Design Interview Questions](https://www.fullstack.cafe/interview-questions/api-design)

*Source:* [journaldev.com](https://www.journaldev.com/9193/web-services-interview-questions-soap-restful)

## *Q11*:

## What are the best practices to create a standard URI for a web service?

**Mid**

**[API Design](https://www.fullstack.cafe/interview-questions/api-design" \o "API Design Interview Questions)**[46](https://www.fullstack.cafe/interview-questions/api-design" \o "API Design Interview Questions)

Answer

Following are important points to be considered while designing a URI:

* **Use Plural Noun** − Use plural noun to define resources. For example, we've used users to identify users as a resource.
* **Avoid using spaces** − Use underscore(\_) or hyphen(-) when using a long resource name, for example, use authorized\_users instead of authorized%20users.
* **Use lowercase letters** − Although URI is case-insensitive, it is good practice to keep url in lower case letters only.
* **Maintain Backward Compatibility** − As Web Service is a public service, a URI once made public should always be available. In case, URI gets updated, redirect the older URI to new URI using HTTP Status code, 300.
* **Use HTTP Verb** − Always use HTTP Verb like GET, PUT, and DELETE to do the operations on the resource. It is not good to use operations names in URI.

*Having Tech or Coding Interview?* Check 👉 [46 API Design Interview Questions](https://www.fullstack.cafe/interview-questions/api-design)

*Source:* [tutorialspoint.com](https://www.tutorialspoint.com/restful/restful_interview_questions.htm)

## *Q12*:

## What are the best practices to design a resource representation?

**Mid**

**[API Design](https://www.fullstack.cafe/interview-questions/api-design" \o "API Design Interview Questions)**[46](https://www.fullstack.cafe/interview-questions/api-design" \o "API Design Interview Questions)

Answer

Following are important points to be considered while designing a representation format of a resource in a RESTful web services −

* **Understandability** − Both Server and Client should be able to understand and utilize the representation format of the resource.
* **Completeness** − Format should be able to represent a resource completely. For example, a resource can contain another resource. Format should be able to represent simple as well as complex structures of resources.
* **Linkablity** − A resource can have a linkage to another resource, a format should be able to handles such situations.

*Having Tech or Coding Interview?* Check 👉 [46 API Design Interview Questions](https://www.fullstack.cafe/interview-questions/api-design)

*Source:* [tutorialspoint.com](https://www.tutorialspoint.com/restful/restful_interview_questions.htm)

## *Q13*:

## What are the disadvantages of statelessness in RESTful Webservices?

**Mid**

**[API Design](https://www.fullstack.cafe/interview-questions/api-design" \o "API Design Interview Questions)**[46](https://www.fullstack.cafe/interview-questions/api-design" \o "API Design Interview Questions)

Answer

Following is the disadvantage of statelessness in RESTful web services:

* Web services need to get extra information in each request and then interpret to get the client's state in case client interactions are to be taken care of.

*Having Tech or Coding Interview?* Check 👉 [46 API Design Interview Questions](https://www.fullstack.cafe/interview-questions/api-design)

*Source:* [tutorialspoint.com](https://www.tutorialspoint.com/restful/restful_interview_questions.htm)



[22 Dart Interview Questions (ANSWERED) Flutter Dev Must Know](https://www.fullstack.cafe/blog/dart-interview-questions)

**[Flutter](https://www.fullstack.cafe/interview-questions/flutter" \o "Flutter Interview Questions)**[68](https://www.fullstack.cafe/interview-questions/flutter" \o "Flutter Interview Questions)

## *Q14*:

## What are the primary security issues of web service?

**Mid**

**[API Design](https://www.fullstack.cafe/interview-questions/api-design" \o "API Design Interview Questions)**[46](https://www.fullstack.cafe/interview-questions/api-design" \o "API Design Interview Questions)

Answer

To ensure reliable transactions and secure confidential information, web services requires very high level of security which can be only achieved through Entrust Secure Transaction Platform. Security issues for web services are broadly divided into three sections as described below

**1) Confidentiality:** A single web service can have multiple applications and their service path contains a potential weak link at its nodes. Whenever messages or say XML requests are sent by the client along with the service path to the server, they must be encrypted. Thus, maintaining the confidentiality of the communication is a must.

**2) Authentication:** Authentication is basically performed to verify the identity of the users as well as ensuring that the user using the web service has the right to use or not? Authentication is also done to track user’s activity. There are several options that can be considered for this purpose

* Application level authentication
* HTTP digest and HTTP basic authentication
* Client certificates

**3) Network Security:** This is a serious issue which requires tools to filter web service traffic.

*Having Tech or Coding Interview?* Check 👉 [46 API Design Interview Questions](https://www.fullstack.cafe/interview-questions/api-design)

*Source:* [softwaretestinghelp.com](https://www.softwaretestinghelp.com/web-services-interview-questions/)

## *Q15*:

## What is addressing in RESTful webservices?

**Mid**

**[REST & RESTful](https://www.fullstack.cafe/interview-questions/rest-and-restful" \o "REST & RESTful Interview Questions)**[27](https://www.fullstack.cafe/interview-questions/rest-and-restful" \o "REST & RESTful Interview Questions)

Answer

Addressing refers to locating a resource or multiple resources lying on the server. It is analogous to locate a postal address of a person.

*Having Tech or Coding Interview?* Check 👉 [27 REST & RESTful Interview Questions](https://www.fullstack.cafe/interview-questions/rest-and-restful)

*Source:* [tutorialspoint.com](https://www.tutorialspoint.com/restful/restful_interview_questions.htm)

 Full-Stack, Web & Mobile

 Coding & Data Structures

 System Design & Architecture

**[T-SQL](https://www.fullstack.cafe/interview-questions/t-sql" \o "T-SQL Interview Questions)**[51](https://www.fullstack.cafe/interview-questions/t-sql" \o "T-SQL Interview Questions)

**[Laravel](https://www.fullstack.cafe/interview-questions/laravel" \o "Laravel Interview Questions)**[41](https://www.fullstack.cafe/interview-questions/laravel" \o "Laravel Interview Questions)

**[LINQ](https://www.fullstack.cafe/interview-questions/linq" \o "LINQ Interview Questions)**[38](https://www.fullstack.cafe/interview-questions/linq" \o "LINQ Interview Questions)

**[MongoDB](https://www.fullstack.cafe/interview-questions/mongodb" \o "MongoDB Interview Questions)**[83](https://www.fullstack.cafe/interview-questions/mongodb" \o "MongoDB Interview Questions)

**[SQL](https://www.fullstack.cafe/interview-questions/sql" \o "SQL Interview Questions)**[42](https://www.fullstack.cafe/interview-questions/sql" \o "SQL Interview Questions)

**[.NET Core](https://www.fullstack.cafe/interview-questions/net-core" \o ".NET Core Interview Questions)**[52](https://www.fullstack.cafe/interview-questions/net-core" \o ".NET Core Interview Questions)

**[Vue.js](https://www.fullstack.cafe/interview-questions/vuejs" \o "Vue.js Interview Questions)**[41](https://www.fullstack.cafe/interview-questions/vuejs" \o "Vue.js Interview Questions)

**[PWA](https://www.fullstack.cafe/interview-questions/pwa" \o "PWA Interview Questions)**[22](https://www.fullstack.cafe/interview-questions/pwa" \o "PWA Interview Questions)

**[Software Testing](https://www.fullstack.cafe/interview-questions/software-testing" \o "Software Testing Interview Questions)**[26](https://www.fullstack.cafe/interview-questions/software-testing" \o "Software Testing Interview Questions)

**[Spring](https://www.fullstack.cafe/interview-questions/spring" \o "Spring Interview Questions)**[87](https://www.fullstack.cafe/interview-questions/spring" \o "Spring Interview Questions)

🤖 Having Data Science & ML Interview? Check  MLStack.Cafe - 1299 Data Science & ML Interview Questions & Answers!

## *Q16*:

## What is statelessness in RESTful Webservices?

**Mid**

**[API Design](https://www.fullstack.cafe/interview-questions/api-design" \o "API Design Interview Questions)**[46](https://www.fullstack.cafe/interview-questions/api-design" \o "API Design Interview Questions)

Answer

As per REST architecture, a RESTful web service should not keep a client state on server. This restriction is called statelessness. It is responsibility of the client to pass its context to server and then server can store this context to process client's further request. For example, session maintained by server is identified by session identifier passed by the client.

*Having Tech or Coding Interview?* Check 👉 [46 API Design Interview Questions](https://www.fullstack.cafe/interview-questions/api-design)

*Source:* [tutorialspoint.com](https://www.tutorialspoint.com/restful/restful_interview_questions.htm)

## *Q17*:

## What is the difference between Monolithic, SOA and Microservices Architecture?

**Mid**

**[Microservices](https://www.fullstack.cafe/interview-questions/microservices" \o "Microservices Interview Questions)**[34](https://www.fullstack.cafe/interview-questions/microservices" \o "Microservices Interview Questions)

Answer

* **Monolithic Architecture** is similar to a big container wherein all the software components of an application are assembled together and tightly packaged.
* A **Service-Oriented Architecture** is a collection of services which communicate with each other. The communication can involve either simple data passing or it could involve two or more services coordinating some activity.
* **Microservice Architecture** is an architectural style that structures an application as a collection of small autonomous services, modeled around a business domain.

*Having Tech or Coding Interview?* Check 👉 [34 Microservices Interview Questions](https://www.fullstack.cafe/interview-questions/microservices)

*Source:* [edureka.co](https://www.edureka.co/blog/interview-questions/microservices-interview-questions/)



[35 Entity Framework Interview Questions (ANSWERED + .NET Core Updates)](https://www.fullstack.cafe/blog/entity-framework-interview-questions)

**[Entity Framework](https://www.fullstack.cafe/interview-questions/entity-framework" \o "Entity Framework Interview Questions)**[57](https://www.fullstack.cafe/interview-questions/entity-framework" \o "Entity Framework Interview Questions)

## *Q18*:

## What is the purpose of HTTP Status Code?

**Mid**

**[API Design](https://www.fullstack.cafe/interview-questions/api-design" \o "API Design Interview Questions)**[46](https://www.fullstack.cafe/interview-questions/api-design" \o "API Design Interview Questions)

Answer

HTTP Status code are standard codes and refers to predefined status of task done at server. For example, HTTP Status 404 states that requested resource is not present on server.

Consider following status codes:

* 200 - OK, shows success.
* **201** - CREATED, when a resource is successful created using POST or PUT request. Return link to newly created resource using location header.
* **304** - NOT MODIFIED, used to reduce network bandwidth usage in case of conditional GET requests. Response body should be empty. Headers should have date, location etc.
* 400 - BAD REQUEST, states that invalid input is provided e.g. validation error, missing data.
* **401** - FORBIDDEN, states that user is not having access to method being used for example, delete access without admin rights.
* 404 - NOT FOUND, states that method is not available.
* **409** - CONFLICT, states conflict situation while executing the method for example, adding duplicate entry.
* 500 - INTERNAL SERVER ERROR, states that server has thrown some exception while executing the method.

*Having Tech or Coding Interview?* Check 👉 [46 API Design Interview Questions](https://www.fullstack.cafe/interview-questions/api-design)

*Source:* [tutorialspoint.com](https://www.tutorialspoint.com/restful/restful_interview_questions.htm)

## *Q19*:

## What is the use of Accept and Content-Type Headers in HTTP Request?

**Mid**

**[API Design](https://www.fullstack.cafe/interview-questions/api-design" \o "API Design Interview Questions)**[46](https://www.fullstack.cafe/interview-questions/api-design" \o "API Design Interview Questions)

Answer

* **Accept headers** tells web service what kind of response client is accepting, so if a web service is capable of sending response in XML and JSON format and client sends Accept header as application/xml then XML response will be sent. For Accept header application/json, server will send the JSON response.
* **Content-Type header** is used to tell server what is the format of data being sent in the request. If Content-Type header is application/xml then server will try to parse it as XML data. This header is useful in HTTP Post and Put requests.

*Having Tech or Coding Interview?* Check 👉 [46 API Design Interview Questions](https://www.fullstack.cafe/interview-questions/api-design)

*Source:* [journaldev.com](https://www.journaldev.com/9193/web-services-interview-questions-soap-restful)

## *Q20*:

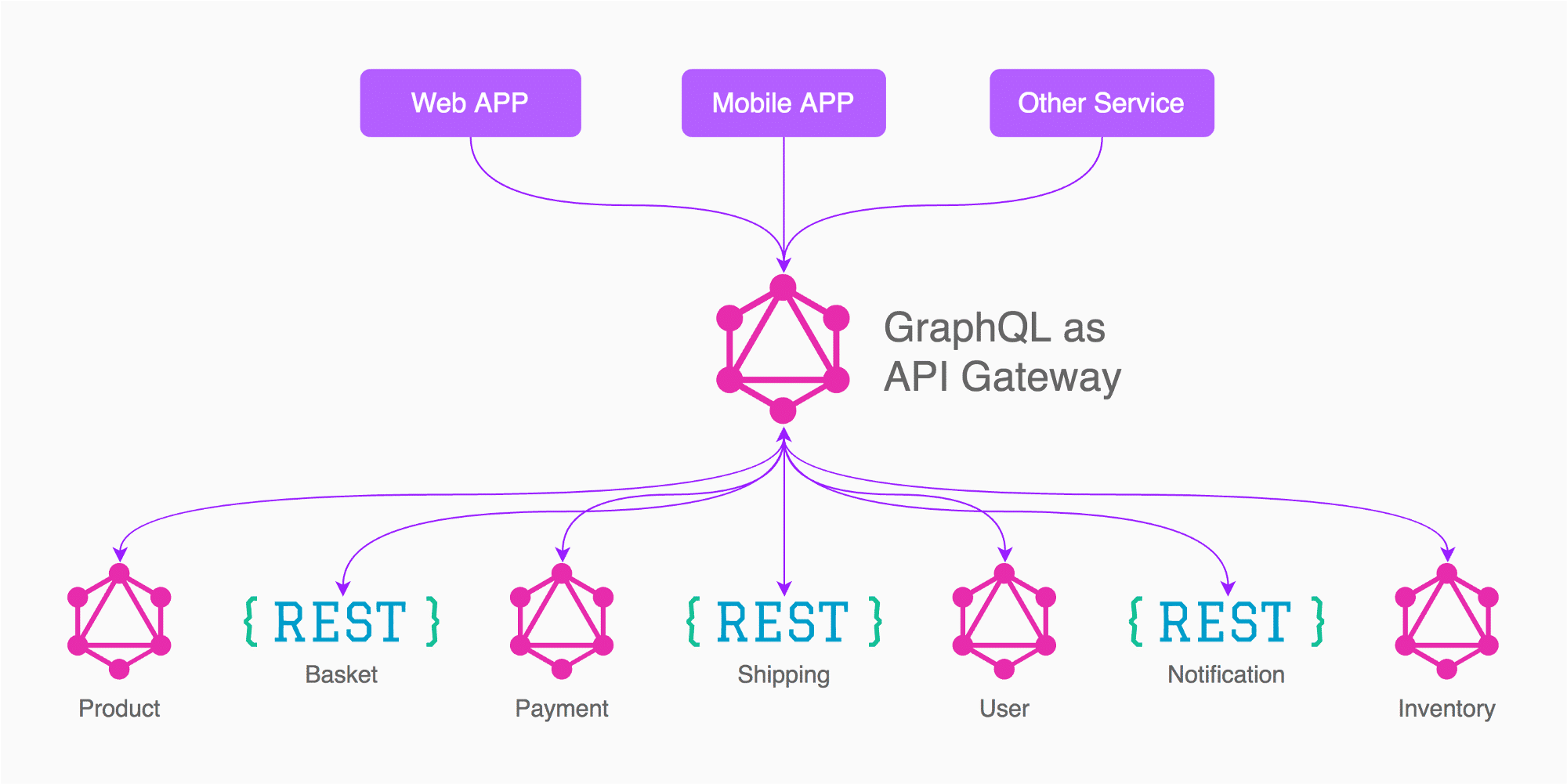
## Whether do you find GraphQL the right fit for designing microservice architecture?

**Mid**

**[Microservices](https://www.fullstack.cafe/interview-questions/microservices" \o "Microservices Interview Questions)**[34](https://www.fullstack.cafe/interview-questions/microservices" \o "Microservices Interview Questions)

Answer

GraphQL and microservices are a perfect fit, because GraphQL hides the fact that you have a microservice architecture from the clients. From a backend perspective, you want to split everything into microservices, but from a frontend perspective, you would like all your data to come from a single API. Using GraphQL is the best way I know of that lets you do both. It lets you split up your backend into microservices, while still providing a single API to all your application, and allowing joins across data from different services.



# STH: 20Q

**Q #1) What is your understanding of what are RESTful web services?**

**Answer:**Just like SOAP (Simple Object Access Protocol), which is used to develop web services by the XML method, RESTful web services use web protocol i.e. HTTP protocol method. They have the feature like scalability, maintainability, help multiple application communication built on various programming languages, etc.

RESTful web service implementation defines the method of accessing various resources that are required by the client and he has sent the request to the server through the web browser.

**The important aspects of this implementation include:**

* Resources
* Request Headers
* Request Body
* Response Body
* Status codes

**Q #2) Name the protocol which is used by RESTful web services.**

**Answer:**RESTful web services use a famous web protocol i.e. HTTP protocol. This serves as a medium of data communication between client and server. HTTP standard methods are used to access resources in RESTful web service architecture.

**Q #3) Explain the term ‘Addressing’ with respect to RESTful WEB service.**

**Answer:**Just like we require an address with postal code to reach any person, in the same way, ‘Addressing’ locates resources that are present on the server for the purpose of hosting web services. This is usually done with URI i.e. Unified Resource Identifier.

**Q #4) Enlist features of RESTful web services.**

**Answer:**Every RESTful web services should have the following features and characteristics that are enlisted below:

* Based on the Client-Server representation.
* Use of HTTP protocol for performing functions like fetching data from the web service, retrieving resources, execution of any query, etc.
* The communication between the server and client is performed through the medium known as ‘messaging’.
* Addressing of resources available on the server through URIs.
* Based on the concept of statelessness where every client request and the response is independent of the other with complete assurance of providing required information.
* Uses the concept of caching.
* Works on the Uniform interface.

**Q #5) Explain messaging technique.**

**Answer:**Messages are the mode of exchanging data for any type of communication to take place. In the same way, HTTP protocol plays the role of message communication between the client and server through HTTP Request and Response methods. HTTP request is sent by the client who contains information about the data and in turn, receives HTTP Response from the server.

Messages are the collection of information about the data i.e. Metadata.

**Q #6) What are the core components of the HTTP request and HTTP response?**

**Answer:**The core components under HTTP Request are:

* **Verb:** Includes methods like GET, PUT, POST, etc.
* Uniform Resource Identifier for identifying the resources available on the server.
* HTTP Version for specifying the HTTP version.
* HTTP Request header for containing the information about the data.
* HTTP Request body that contains the representation of the resources in use.

**The core components under HTTP Response are:**

* **Request Code:** This contains various codes that determine the status of the server response.
* HTTP Version for specifying the HTTP version.
* HTTP Response header for containing the information about the data.
* HTTP Response body that contains the representation of the resources in use.

**Q #7) Explain the term ‘Statelessness’ with respect to RESTful WEB service.**

**Answer:**In REST, ST itself defines State Transfer and Statelessness means complete isolation. This means, the state of the client’s application is never stored on the server and is passed on.

In this process, the clients send all the information that is required for the server to fulfill the HTTP request that has been sent. Thus every client requests and the response is independent of the other with complete assurance of providing the required information.

Every client passes a ‘session identifier’ which also acts as an identifier for each session.

**Q #8) Enlist advantages and disadvantages of ‘Statelessness’.**

**Answer:**In the above question, we have understood the meaning of statelessness with respect to client-server communication. Now, let us see some of its advantages and disadvantages.

**Advantages:**

* Every method required for communication is identified as an independent method i.e. there are no dependencies to other methods.
* Any previous communication with the client and server is not maintained and thus the whole process is very much simplified.
* If any information or metadata used earlier in required in another method, then the client sends again that information with the HTTP request.
* The HTTP protocol and REST web service, both shares the feature of statelessness.

**Disadvantages:**

* In every HTTP request from the client, the availability of some information regarding the client state is required by the web service.

**Q #9) Enlist some important constraints for RESTful web services.**

**Answer:**Every constraint has positive as well as negative impacts and to produce an overall architecture, there should be a balance between both of them.

**Below mentioned are some important constraints for RESTful web service:**

* There should be separate concerns for each server and client which will help to maintain the modularity within the application. This will also reduce the complexity and increase the scalability.
* The client-server communication should be stateless, which means no previous information is used and the complete execution is done in isolation. In cases of failure, it also helps the client to recover.
* In client-server communication, the HTTP response should be cacheable so that when required cached copy can be used which in turn enhances the scalability and performance of the server.
* The fourth constraint is the uniform interface which allows client-server interaction to be easily understood. This constraint is further divided into four sub-constraints as:
  + Resource Identification
  + Resource Manipulation
  + Each message is easily understood and is self-descriptive.
  + Hypermedia, which is defined as the text with hyperlinks and when clicked, it moves to another application state.
* Client-server communication should be done on a layered system and thus the client should only have knowledge about the intermediate level with which communication is being done,

**Q #10) What is a ‘Resource’?**

**Answer:**Just like the ‘Object’ instance, we have learned in Object Orient Programming Language, in the same way, ‘Resource’ is defined as an object of a type which can be an image, HTML file, text data, and any type of dynamic data. There are varieties of representation formats available in order to represent a resource.

**Some most common Resources are enlisted below:**

* JSON
* YAML
* XML
* HTML

**Q #11) Why proper representation of Resource is required?**

**Answer:**Representation is very important because it determines the easy identification of resources. With proper representations of resource in the proper format, allows the client to easily understand the format.

**Q #12) Enlist some important points that should be kept in mind while designing Resources representation for RESTful web services.**

**Answer:**As there are no restrictions on the format in which the resource representation is done but just that the main requirement is the format of the representation should be as per the client requirement.

**A good resource representation is designed by considering the following main points:**

* The resource representation format should be easily understood by the client and server.
* The representation should be complete regardless of its format structure, which may be complex or simple.
* In the case of the link of the resources to other resources, such cases should also be considered and handled.

**Q #13) What is Caching?**

**Answer:**Caching is the process in which server response is stored so that a cached copy can be used when required and there is no need for generating the same response again. This process not only reduces the server load but in turn increase the scalability and performance of the server. Only the client is able to cache the response and that too for a limited period of time.

**Mentioned below are the header of the resources and their brief description so that they can be identified for the caching process:**

* Time and date of resource creation
* Time and date of resource modification that usually stores the last detail.
* Cache-control header
* Time and date at which the cached resource will expire.
* The age which determines the time from when the resource has been fetched.

**Q #14) Explain Cache-control header.**

**Answer:**A standard Cache-control header can help in attaining cache ability. Enlisted below is the brief description of the various cache-control header:

* **Public:** Resources that are marked as the public can be cached by any intermediate components between the client and the server.
* **Private:** Resources that are marked as private can only be cached by the client.
* No cache means that a particular resource cannot be cached and thus the whole process is stopped.

**Q #15) What are the best practices that are to be followed while designing RESTful web services?**

**Answer:**To design a secure RESTful web service, there are some best practices or say points that should be considered.

**These are explained as follows:**

* Every input on the server should be validated.
* Input should be well-formed.
* Never pass any sensitive data through URL.
* For any session, the user should be authenticated.
* Only HTTP error messages should be used for indicating any fault.
* Use message format that is easily understood and is required by the client.
* Unified Resource Identifier should be descriptive and easily understood.

**Q #16) What is Payload?**

**Answer:**The request data which is present in the body part of every HTTP message is referred to as ‘Payload’.  In Restful web service, the payload can only be passed to the recipient through the POST method.

There is no limit of sending data as payload through the POST method but the only concern is that more data will consume more time and bandwidth. This may consume much of the user’s time also.

**Q #17) Enlist some of the HTTP methods with description.**

**Answer: Enlisted below is the list of HTTP methods with their descriptions:**

* **GET:** This is a read-only operation that fetches the list of users on the server.
* **PUT:** This operation is used for the creation of any new resource on the server.
* **POST:** This operation is used for updating an old resource or for creating a new resource.
* **DELETE:** As the name suggests, this operation is used for deleting any resource on the server.
* **OPTIONS:** This operation fetches the list of any supported options of resources that are available on the server.

**Q #18) What is the difference between the PUT method and the POST method?**

**Answer:**The major difference between the PUT and POST method is that the result generated with the PUT method is always the same no matter how many times the operation is performed. On the other hand, the result generated by POST operation is always different every time.

**Q #19) What is your understanding about JAX-RS?**

**Answer:**JAX-RS is defined as the Java API for RESTful web service. Among multiple libraries and framework, this is considered as the most suitable Java programming language based API which supports RESTful web service.

**Some of the implementations of JAX-RS are:**

* Jersey
* RESTEasy
* Apache CFX
* Play

Among these, Jersey is the most popular framework.

**Q #20) What are HTTP status codes? Enlist few with meaning.**

**Answer:**HTTP status codes basically are the representation of the status of the task that has been performed on the server, with the mode of some codes. Every code has their own meaning.

**Some of the HTTP status codes with their meaning are as follows:**

* **Code 200:** This indicates success.
* **Code 201:** This indicates resource has been successfully created.
* **Code 204:** This indicates that there is no content in the response body.
* **Code 404:** This indicates that there is no method available.

There are few more such codes that indicate the status.

# Medium: 45Q

1. **What is an API**

**An API is an interface that allows users to interact with a program through a client.** A client can be a browser that an end-user uses to access a website. For example, when you use your browser to access any website, you’re interacting with that specific website’s API through the browser.

**A client can also be another application.** If you’re a software developer, you might write a program that accesses the website API to pull in information about different things through the client application. Either way, the client provides access to the API and its resources which are the objects that the application stores information about.

**2. Explain what is REST and RESTFUL**

**REST** *(Representational State Transfer)* is web standards based architectural style or approach for communications purpose that is often used in various web services development.It uses **HTTP** Protocol for data communication and a relatively new aspect of writing web API.

**RESTFUL** is referred for web services written by applying REST architectural concept are called RESTFUL services. It focuses on system resources and how state of resource should be transported over HTTP protocol to different clients written in different language.

REST suggests to create an object of the data requested by the client and send the values of the object in response to the user. For example, if the user is requesting for a movie in any country at a certain place and time then we can create an object on the server-side.

So, over here, we have an object and we are sending the state of an object. This is why REST is known as Representational State Transfer.

REST was first introduced by **Roy Fielding** in 2000.

3.**When we will call a web service a RESTFUL service**

REST architecture presents a set of constraints/guiding principles to be used in the creation of web services. The services that use REST constraints are called as RESTful Web Services. To be referred to as RESTful, it should satisfy the six-guiding constraints. These constraints are names as the**client-server**, **stateless**, **cacheable**, **uniform interface**, **layered system**, and **code** **on demand.**

4. **Explain the architectural style for creating web API in REST**

The architectural style for creating web api are:

HTTP for client server communicationXML/JSON as formatting languageSimple URI as the address for the servicesStateless communication

5. **What are Resources in a REST architecture**

In the REST architecture, every content is a resource. It can be a text file, HTML pages, images, videos, or business data. A resource is an object with:

a type,relationship with other resources andmethods that operate on it.

**Resources** are identified with:

their URI(**Uniform Resource Identifiers**),HTTP methods they support andrequest/response data type and format of data.

6. **What is options in REST**

**The options** allows the client of the **REST API** to determine what HTTP methods (GET, HEAD, POST, PUT, DELETE) can be used for the resource identified by the requested URI. The client determines without initiating a resource request.

The REST OPTIONS method is also used for the **CORS** (**Cross**-**Origin** **Resource** **Sharing**) request.

7. **What is an URI in REST**

**URI** (**Uniform Resource Identifiers**) is used to identify each resource in the REST. An HTTP operation is called by the client application to access the resource.

//FORMAT for creating a URI<protocol>://<service-name>/<ResourceType>/<ResourceID>

8. **What are the HTTP methods supported by REST**

HTTP methods supported by REST are:

**GET:** It requests a resource at the request URL. It should not contain a request body as it will be discarded. Maybe it can be cached locally or on the server.**POST:** It submits information to the service for processing; it should typically return the modified or new resource**PUT:** At the request URL it update the resource**DELETE:** At the request URL it removes the resource**OPTIONS:** It indicates which techniques are supported**HEAD:** About the request URL it returns meta information

9. **What is the most popular way to represent a resource in REST**

REST uses different representations to define a resource like text, JSON, and XML. **XML**and**JSON** are the most popular representations of resources.

10. **What do you understand by payload in RESTFul web service**

Request body of every HTTP message includes request data called as Payload. This part of the message is of interest to the recipient.

We can say that we send the payload in the POST method but not in **<GET>**and **<DELETE>** methods.

11. **What is the upper limit for a payload to pass in the POST method**

<GET> appends data to the service URL. But, its size shouldn’t exceed the maximum URL length. However, <POST> doesn’t have any such limit.

So, a user can pass unlimited data as the payload to the POST method. But, if we consider a real use case, then sending a POST with large payload will consume more bandwidth. It’ll take more time and present performance challenges to your server. Hence, a user should take action accordingly.

12. **Explain the caching mechanism**

Caching is a process of storing server response at the client end. It makes the server save significant time from serving the same resource again and again.

The server response holds information which leads a client to perform the caching. It helps the client to decide how long to archive the response or not to store it at all.

13. **Mention whether you can use GET request instead of PUT to create a resource**

No, you are not supposed to use PUT for GET. GET operations should only have view rights, while PUT resource is used for updating a data.

14. **What is the difference between PUT and POST**

**PUT** puts a file or resource at a particular URI and exactly at that URI. If there is already a file or resource at that URI, PUT changes that file or resource. If there is no resource or file there, PUT makes one.

**POST**sends data to a particular URI and expects the resource at that URI to deal with the request. The web server at this point can decide what to do with the data in the context of specified resource.

**PUT** is idempotent meaning, invoking it any number of times will not have an impact on resources.

However, **POST** is not idempotent, meaning if you invoke POST multiple times it keeps creating more resources.

15. **Mention some key characteristics of REST**

Some key characteristics of REST includes

1. REST is stateless, therefore the SERVER has no state (or session data)2. With a well-applied REST API, the server could be restarted between two calls as every data is passed to the server3. Web service mostly uses POST method to make operations, whereas REST uses GET to access resources

16. **Mention what is the difference between AJAX and REST**

**AJAX**

1. In Ajax, the request are sent to the server by using XMLHttpRequest objects. The response is used by the JavaScript code to dynamically alter the current page2. Ajax is a set of technology.It is a technique of dynamically updating parts of UI without having to reload the page3. Ajax eliminates the interaction between the customer and server asynchronously

**REST**

1. REST have a URL structure and a request/response pattern the revolve around the use of resources2. REST is a type of software architecture and a method for users to request data or information from servers3. REST requires the interaction between the customer and server

17. **What are the advantages of Web Services**

Some of the advantages of web services are:

**1. Interoperability**: Web services are accessible over network and runs on HTTP/SOAP protocol and uses XML/JSON to transport data, hence it can be developed in any programming language. Web service can be written in java programming and client can be PHP and vice versa.**2. Reusability**: One web service can be used by many client applications at the same time.**3. Loose Coupling**: Web services client code is totally independent with server code, so we have achieved loose coupling in our application.**4. Easy to deploy and integrate**, just like web applications.**5. Multiple service versions** can be running at same time.

18. **What are advantages of REST web services**

Some of the advantages of REST web services are:

1. Learning curve is easy since it works on HTTP protocol2. Supports multiple technologies for data transfer such as text, xml, json, image etc.3. No contract defined between server and client, so loosely coupled implementation.4. REST is a lightweight protocol5. REST methods can be tested easily over browser.

19.**What are different types of Web Services**

There are two types of web services:

**SOAP Web ServicesRestful Web Services**:

20. **What is SOAP**

**SOAP** stands for Simple Object Access Protocol. SOAP is an XML based industry standard protocol for designing and developing web services.

21. **What is WSDL**

**WSDL** stands for Web Service Description Language. WSDL is an XML based document that provides technical details about the web service. Some of the useful information in WSDL document are:

method name,port types,service end point,binding,method parameters etc.

22.**List the main differences between SOAP and REST**

1. SOAP is a protocol through which two computer communicates by sharing the XML document while Rest is a service architecture and design for network-based software architecture.2. SOAP supports the only XML format while REST supports many different data formats.3. SOAP does not support caching while REST supports caching.4. SOAP is like a custom desktop application, closely connected to the server while A REST client is just like a browser and uses standard methods. An application has to fit inside it.5. SOAP is slower than the REST while REST is faster than SOAP.6. SOAP runs on HTTP but envelopes the message while REST uses the HTTP headers to hold meta information.

23. **State the core components of an HTTP Request**

Each HTTP request includes five key elements.

1. The Verb which indicates HTTP methods such as GET, PUT, POST, DELETE.2. URI stands for Uniform Resource Identifier.It is the identifier for the resource on the server.3. HTTP Version which indicates HTTP version, for example-HTTP v1.1.4. Request Header carries metadata (as key-value pairs) for the HTTP Request message. Metadata could be a client (or browser) type, the format that the client supports, message body format, and cache settings.5. Request Body indicates the message content or resource representation.

24. **State the core components of an HTTP response**

Every HTTP response includes four key elements.

1. Status/Response Code — Indicates Server status for the resource present in the HTTP request. For example, 404 means resource not found, and 200 means response is ok.**2.** HTTP Version — Indicates HTTP version, for example-HTTP v1.1.**3.** Response Header — Contains metadata for the HTTP response message stored in the form of key-value pairs. For example, content length, content type, response date, and server type.**4.** Response Body — Indicates response message content or resource representation.

25. **What are the tools available for testing web services**

Following tools can help in testing the SOAP and RESTful web services.

1. SOAP UI tool.  
**2.** Poster for Firefox browser.  
**3.** The Postman extension for Chrome.

26.**What is Postman**

**Postman** is a popular test and development tool to simplify the API workflow. It provides the tool to manage every stage of the API lifecycle and makes the development of the API simple.

With**postman**, you can design, debug, test, document, monitor, and publish the **API** from one place. It also provides version control and tagging to maintain multiple versions of the API. It also provides a testing tool to automate the testing process.

27. **List major HTTP response codes returned by REST API**

**The status** code in the REST API is divided into five categories. **They are**,

**1xx** — It is used to communicate the transfer protocol-level information.**2xx** — It is used to indicate the request was accepted successfully. Some codes are,200 (**OK**) — It indicates the request is successfully carried out.201 (**Created**) — It is returned when a resource is created inside the collection.202 (**Accepted**) — It indicates the request has been accepted for processing.204 (**No Content**) — It indicates when a request is declined.**3xx** — It indicates the client must take additional action to complete the request.**4xx** — It is the client error status code.**5xx** — It is the server error status code.

28.**What is a stateless server**

**A stateless server** is a server that keeps no state information. Stateless file servers do not store any session state. Therefore, every client request is treated independently and not as a part of a new or existing session. A stateless server does not need a client to first establish a connection to the server. So, it views a client request as an independent transaction and responds to it.

29. **Explain the factors that help to decide about the style of web service to use? SOAP or REST?**

In general, using REST-based web service is preferred due to its simplicity, performance, scalability, and support for multiple data formats.

However, SOAP is favorable to use where service requires an advanced level of security and transactional reliability.

But you can read the following facts before opting for any of the styles.

1. Does the service expose data or business logic? To expose data REST will be a better choice and SOAP for logic.**2.** If the consumer or the service providers require a formal contract, then SOAP can provide such a contract via WSDL.**3.** Need to support multiple data formats. REST supports this.**4.** Support for AJAX calls. REST can use the XMLHttpRequest.**5.** Synchronous and asynchronous calls — SOAP enables both synchronous/asynchronous operations whereas REST has built-in support for synchronous.**6.** Stateless or Stateful calls -REST is suited for stateless operations.

Here are some of the advanced-level facts that you can consider as well.

1. Security requirement — SOAP provides a high level of security.**2.** Transaction support — SOAP has good support for transaction management.**3.** Limited bandwidth — SOAP has a lot of overhead when sending/receiving packets since it’s XML based, requires a SOAP header. However, the REST requires less bandwidth to send requests to the server. Its messages are mostly built using JSON.**4.** Ease of use — It is easy to implement, test, and maintain REST-based application.

30. **What are the best practices to be followed while designing a secure RESTful web service**

Following are the best practices to be followed while designing a RESTful web service −

**Validation** − Validate all inputs on the server. Protect your server against SQL or NoSQL injection attacks.**Session based authentication** − Use session based authentication to authenticate a user whenever a request is made to a Web Service method.**No sensitive data in URL** − Never use username, password or session token in URL , these values should be passed to Web Service via POST method.**Restriction on Method execution** − Allow restricted use of methods like GET, POST, DELETE. GET method should not be able to delete data.**Validate Malformed XML/JSON** − Check for well formed input passed to a web service method.**Throw generic Error Messages** − A web service method should use HTTP error messages like 403 to show access forbidden etc.

31. **What should be the purpose of HEAD method of RESTful web services**

It should return only HTTP Header, no Body and should be read only.

32**. Which header of HTTP response, provides the date and time of the resource when it was created**

Date header provides the date and time of the resource when it was created.

33. **Which header of HTTP response, provides the date and time of the resource when it was last modified**

Last Modified header provides the date and time of the resource when it was last modified.

34. **Which header of HTTP response provides control over caching**

Cache-Control is the primary header to control caching.

35. **Which header of HTTP response sets expiration date and time of caching**

Expires header sets expiration date and time of caching.

36.**Which directive of Cache Control Header of HTTP response indicates that resource is cachable by any component**

Public directive indicates that resource is cachable by any component.

37. **Which directive of Cache Control Header of HTTP response indicates that resource is cachable by only client and server, no intermediary can cache the resource**

Private directive indicates that resource is cachable by only client and server, no intermediary can cache the resource.

38. **Which directive of Cache Control Header of HTTP response indicates that resource is not cachable**

no-cache/no-store directive indicates that resource is not cachable.

39. **Which directive of Cache Control Header of HTTP response can set the time limit of caching**

max-age directive indicates that the caching is valid up to max-age in seconds. After this, client has to make another request.

40. **Which directive of Cache Control Header of HTTP response provides indication to server to revalidate resource if max-age has passed**

**must-revalidate** directive provides indication to server to revalidate resource if max-age has passed.

41.**What are the best practices for caching**

Always keep static contents like images, css, JavaScript cacheable, with expiration date of 2 to 3 days. Never keep expiry date too high.

Dynamic contents should be cached for few hours only.

42. **What is the purpose of HTTP Status Code**

HTTP Status code are standard codes and refers to predefined status of task done at server.

43.**What is messaging in RESTFUL web services**

A client sends a message in form of a HTTP Request and server responds in form of a HTTP Response. This technique is termed as Messaging. These messages contain message data and metadata i.e. information about message itself.

44. **What is the purpose of HTTP Verb in REST based web services**

VERB identifies the operation to be performed on the resource.

45. **What are the best practices to create a standard URI for a web service**

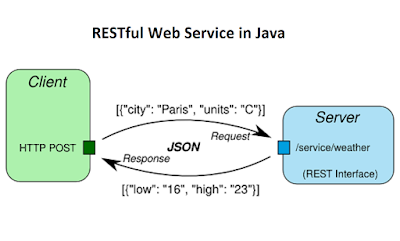
Following are important points to be considered while designing a URI −

**Use Plural Noun** − Use plural noun to define resources. For example, we’ve used users to identify users as a resource.**Avoid using spaces** − Use underscore(\_) or hyphen(-) when using a long resource name, for example, use authorized\_users instead of authorized%20users.**Use lowercase letters** − Although URI is case-insensitive, it is good practice to keep url in lower case letters only.**Maintain Backward Compatibility** − As Web Service is a public service, a URI once made public should always be available. In case, URI gets updated, redirect the older URI to new URI using HTTP Status code, 300.**Use HTTP Verb** − Always use HTTP Verb like GET, PUT, and DELETE to do the operations on the resource. It is not good to use operations names in URI.

# Java67: 20Q

### Question 1: What is REST?

Answer: REST is an architectural style of developing web services that take advantage of the ubiquity of HTTP protocol and leverages the HTTP method to define actions. REST stands for *REpresntational State Transfer*.  
  
  
**Question 2: What is RESTFul Web Service?**  
Answer: There is two popular way to develop web services, using SOAP (Simple Object Access Protocol) which is XML based way to expose web services and second REST-based web services which use the HTTP protocol. Web services developed using REST-style is also known as RESTful Web Services. You can see the [**REST API Design and Development**](https://click.linksynergy.com/fs-bin/click?id=JVFxdTr9V80&subid=0&offerid=323058.1&type=10&tmpid=14538&RD_PARM1=https%3A%2F%2Fwww.udemy.com%2Frest-api%2F)course on Udemy to learn more about them.

[](https://click.linksynergy.com/fs-bin/click?id=JVFxdTr9V80&subid=0&offerid=323058.1&type=10&tmpid=14538&RD_PARM1=https%3A%2F%2Fwww.udemy.com%2Frest-api%2F)

**Question 3: What is HTTP Basic Authentication and how it works?**

### Question 4: Can you tell me which API can be used to develop the RESTFul web service in Java?

Answer: There are many framework and libraries out there, which helps to develop RESTful web services in Java, including JAX-RS, which is a standard way to build REST web services. Jersey is one of the popular implementations of JAX-RS, which also offers more than specification recommendations. Then you also have [RESTEasy](https://javarevisited.blogspot.com/2017/02/difference-between-jax-rs-restlet-jersey-apache-cfx-RESTEasy.html" \t "_blank), [Restlet](https://javarevisited.blogspot.com/2016/10/restlet-helloworld-example-in-java-and-Eclipse.html" \l "axzz5j9AEsxuT" \t "_blank), [Jersey](https://javarevisited.blogspot.com/2017/06/jersey-web-service-hello-world-example.html), and Apache CFX. If you like Scala, then you can also use the Play framework to develop RESTful web services.  
  
  
Question 5: How do you configure the RESTFul web service?  
  
Question 6: How you apply security in RESTFul web services?  
  
Question 7: Have you used securing RESTful APIs with HTTP Basic Authentication  
  
Question 8: How you maintain sessions in RESTful services?

### Question 9: Have you used Jersey API to develop RESTful services in Java?

Answer: Jersey is one of the most popular frameworks and API to develop REST-based web services in Java. Since many organization uses Jersey, they check if the candidate has used it before or not. It's simple to answer, say Yes if you have really used and No if you have not.  
  
In the case of No, you should also mention which framework you have applied for developing RESTful web services, like Jersey, Apache CFX, Play, or Restlet. And, if you want to learn Jersey for building REST APIs in Java, I suggest you take a look at the **[RESTFul Services in Java using Jersey](https://pluralsight.pxf.io/c/1193463/424552/7490?u=https%3A%2F%2Fwww.pluralsight.com%2Fcourses%2Frestful-services-java-using-jersey" \t "_blank)** by Bryan Hansen course on Pluralsight, one of the in-depth course to learn Jersey for RESTful application.

[](https://pluralsight.pxf.io/c/1193463/424552/7490?u=https%3A%2F%2Fwww.pluralsight.com%2Fcourses%2Frestful-services-java-using-jersey)

**Question 10: What is WADL in RESTFul?**

### Question 11: What do you understand by the payload in RESTFul?

Answer: Payload means data that passed inside the request body; also, the payload is not request parameters. So only you can do payload in POST  and not in GET and DELETE method  
  
  
**Question 12: Can you do payload in the GET method?**  
Answer: No, the payload can only be passed using the POST method.

**Question 13: Can you do payload in HTTP DELETE?**  
Answer: This is again, similar to the previous REST interview question, the answer is No. You can only pass payload using the HTTP POST method.

### Question 14: How do you test RESTful web services?

You can test RESTful web service using tools like Postman and Swagger. I love Postman they provide a lot of functionalities to test RESTful web services like you can send a request to any end-points, you can see the response, you can convert them to JSON and XML and you can even inspect request and response parameters, headers, and query parameters.  
  
If you are testing REST API then I strongly suggest learning Postman and if you need a resource this **[Postman Crash Course](https://click.linksynergy.com/deeplink?id=JVFxdTr9V80&mid=39197&murl=https%3A%2F%2Fwww.udemy.com%2Fpostman-crash-course-for-beginners-learn-rest-api-testing%2F" \t "_blank)** on Udemy is a great place to start with. It's also a free course and tutorial, which means you don't need to pay anything, you just need an Udemy account to enroll in the course.

[](https://click.linksynergy.com/deeplink?id=JVFxdTr9V80&mid=39197&murl=https%3A%2F%2Fwww.udemy.com%2Fpostman-crash-course-for-beginners-learn-rest-api-testing%2F)

### Question 15: How much maximum payload you could do in the POST method?

Answer: If you remember the [difference between the GET and POST request](http://java67.blogspot.sg/2014/08/difference-between-post-and-get-request.html" \t "_blank), then you know that unlike GET, which passes data on URL and thus limited by maximum URL length, POST has no such limit. So, theoretically, you can pass unlimited data as payload to the POST method. Still, you need to take practical things into account, like sending a POST with a large payload will consume more bandwidth, take more time and present performance challenge to your server.

### Question 16: What is the difference between SOAP and RESTFul web services?

Answer: There is much difference between these two styles of web services, e.g. SOAP takes more bandwidth because of heavyweight XML based protocol, but REST takes less bandwidth because of widespread use of JSON as message protocol and leveraging HTTP method to define action. This also means that REST is faster than SOAP-based web services.  
  
You can derive many differences between SOAP and RESTful with the fact that it's HTTP based, like REST URLs, can be cached or bookmarked. Here are a few more differences between them :

[](https://pluralsight.pxf.io/c/1193463/424552/7490?u=https%3A%2F%2Fwww.pluralsight.com%2Fcourses%2Frestful-services-java-using-jersey)

### Question 17: If you have to develop web services, which one you will choose SOAP OR RESTful, and why?

Answer: You can answer this question based upon your experience, but the key here is if you know the difference between them, then you can answer this question in more detail. For example, it's easy to develop RESTful web services than SOAP-based web services but later comes with some in-built security features.

### Question 18: What framework you had used to develop RESTFul services?

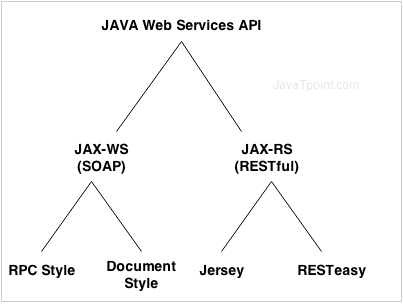
Answer: This is a real experience-based question. If you have used Jersey to develop RESTFul web services, then answer as Jersey but expect some follow-up question on Jersey. Similarly, if you have used Apache CFX or Restlet, then respond to them accordingly.  
  
Read more: [https://www.java67.com/2015/09/top-10-restful-web-service-interview-questions-answers.html#ixzz7NtSDtI8G](https://www.java67.com/2015/09/top-10-restful-web-service-interview-questions-answers.html" \l "ixzz7NtSDtI8G)

# JavaTPoint: 42Q

### 1) What is Web Service?

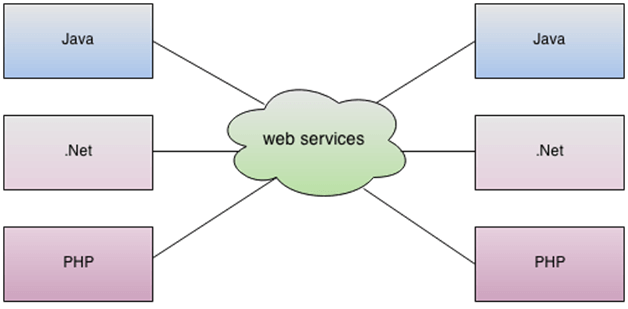
The Web Service is a standard software system used for communication between two devices (client and server) over the network. Web services provide a common platform for various applications written in different languages to communicate with each other over the network.

#### Java Web Services API

  
[More details..](https://www.javatpoint.com/what-is-web-service)

### 2) How does a web service work?

A web service is used to communicate among various applications by using open standards such as HTML, XML, WSDL, and SOAP. You can build a Java-based web service on Solaris that is accessible from your Visual Basic program that runs on Windows. You can also use C# to develop new web services on Windows invokes from your web application that is based on Java Server Pages (JSP) and runs on Linux.



### 3) What are the advantages of web services?

These are some of the important advantages of web services:

338.4K

Online hate speech in UK and US rose by 20 percent during pandemic

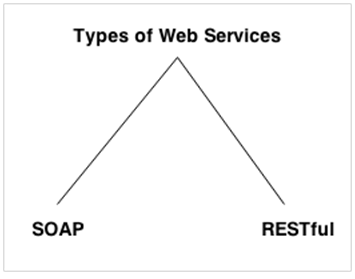
* **Interoperability**: With the help of web services, an application can communicate with other application developed in any language.
* **Reusability**: We can expose the web service so that other applications can use it.
* **Modularity**: With the help of web service, we can create a service for a specific task such as tax calculation.
* **A Standard protocol for every application program**: Web services use standard protocol so that all the client applications written in different languages can understand it. This Standard protocol helps in achieving cross-platform.
* **Cheaper cost for communication**: Web services uses SOAP over HTTP so that anybody can use existing internet for using web services.

[More details..](https://www.javatpoint.com/what-is-web-service)

### 4) What are the different types of web services?

There are two types of web services:

* **SOAP** - It is an XML-based protocol for accessing web services.
* **RESTful** - It is an architectural style, not a protocol.



### 5) What are the main features of web services?

Following is a list of main features of web services:

* It is available over the Internet or private (intranet) networks.
* It uses a standardized XML messaging system.
* It is not tied to any one operating system or programming language.
* It is self-describing via a common XML grammar.
* It is discoverable via a simple find mechanism.

### 6) What is SOAP?

The SOAP stands for Simple Object Access Protocol. It is an XML-based protocol for accessing web services. It is platform independent and language independent. By using SOAP, you can interact with other programming language applications. [More details..](https://www.javatpoint.com/soap-web-services)

### 7) What are the advantages of SOAP web services?

These are some of the important advantages of SOAP web services:

* **WS Security** - SOAP defines its security known as WS Security.
* **Language Independent** - Its web services can be written in any programming language
* **Platform Independent** - Its web services can be executed on any platform.

[More details..](https://www.javatpoint.com/soap-web-services)

### 8) What are the disadvantages of SOAP web services?

These are some of the important disadvantages of SOAP web services:

* **Slow** - It uses XML format that must be parsed to be read and defines many standards that must be followed while developing the SOAP applications. So it is slow and consumes more bandwidth and resource.
* **WSDL Dependent** - It uses WSDL and doesn't have any other mechanism to discover the service.

[More details..](https://www.javatpoint.com/soap-web-services)

### 9) What are the main features of SOAP?

The following list specifies the features of SOAP:

* SOAP is a communication protocol.
* SOAP communicates between applications.
* SOAP is a format for sending messages.
* SOAP is designed to communicate via Internet.
* SOAP is platform independent.
* SOAP is language independent.
* SOAP is simple and extensible.
* SOAP allows you to get around firewalls.
* SOAP developed as a W3C standard.

### 10) What is WSDL?

The WSDL stands for Web Services Description Language. It is an XML document containing information about web services such as method name, method parameter. The Client needs a data dictionary which contains information about all the web services with methods names and parameters list to invoke them for the web services. The Web Service Description Language bridge up this gap, by providing all necessary information to the client.

Some Important elements used in Web Services Description language are as follows:

* **<message>**: The message element in WSDL is used to define all different data elements for each operation performed by the web service.
* **<portType>**: The port type element is used to determine the operation which can be performed by the web service. This operation can have two messages one is input and the second one is the output message.
* **<binding>**: This element contains the used protocol.

[More details..](https://www.javatpoint.com/web-service-components)

### 11) What is UDDI?

The UDDI stands for Universal Description, Discovery and Integration. It is a XML based framework for describing, discovering and integrating web services. It contains a list of available web services. WSDL is the part of UDDI. [More details..](https://www.javatpoint.com/web-service-components)

### 12) What is RESTful web services?

The REST stands for Representational State Transfer. It is an architectural style. It is not a protocol like SOAP. [More details..](https://www.javatpoint.com/restful-web-services)

### 13) What are the advantages of RESTful web services?

These are some of the important advantages of RESTful web services:

* Fast - The Web Services are fast because there is no strict specification of SOAP. It consumes less bandwidth and resource.
* Language Independent - The web services can be written in any programming language.
* Platform Independent - The web services can be executed on any platform.
* Can use SOAP - The web services can use SOAP web services as the implementation.
* Allows different data format - The web service permits different data format such as Plain Text, HTML, XML, and JSON.

[More details..](https://www.javatpoint.com/restful-web-services)

### 14) What is the difference between SOAP and REST web services?

|  |  |  |
| --- | --- | --- |
| **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) | SOAP **defines standards**to be strictly followed. | REST does not define too much standards like SOAP. |
| 6) | SOAP **permits XML** data format only. | REST **permits different** data format such as Plain text, HTML, XML, JSON. |

[More details..](https://www.javatpoint.com/soap-vs-rest-web-services)

### 15) What is SOA?

SOA stands for Service Oriented Architecture. It is a design pattern to provide services to other application through protocol.

  
[More details..](https://www.javatpoint.com/service-oriented-architecture)

### 16) What tools are used to test web services?

The tools used to test web services are:

[Learn more](https://aax-eu.amazon-adsystem.com/x/c/QsCTmbl36a4w2NT-GWxIn9YAAAF_nTX7TwMAAAalBM1yDU0/https:/www.amazon.de/gp/product/B09S8FKYFP?tag=ms-de-21&ref=aap_588963350312460649" \t "_blank)

* **SoapUI tool** for testing SOAP and RESTful web services
* **Poster** for firefox browser
* **Postman** extension for Chrome

### 17) What is the advantage of XML in web service?

In Web service, an XML is used to tag the data, format the data.

### 18) What is the usage of WSDL in a web service?

WSDL is used in web service to describe the availability of service.

### 19) What is Interoperability in Web services?

The Web services facilitate various applications to communicate with each other and share data and services among themselves. Other applications can also use the web services. For example, a VB or .NET application can communicate with a Java web services and vice versa. Web services are used to make the application platform and technology independent.

### 20) Explain the loosely coupled architecture of web services.

A consumer of a web service is not tied to that web service directly. The web service interface can change over time without compromising the client's ability to interact with the service. A tightly coupled system implies that the client and server logic are closely tied to one another, implying that if one interface changes, the other must be updated. Adopting a loosely coupled architecture tends to make software systems more manageable and facilitates simpler integration between different systems.

### 21) What are the advantages of having XML based Web services?

Using XML eliminates any networking, operating system, or platform binding. So Web Services based applications are highly interoperable application at their core level.

### 22) What do you mean by synchronicity?

Synchronicity is used to bind the client to the execution of the service. In synchronous invocations, the client blocks and waits for the service to complete its operation before continuing. On the other hand, synchronous operations facilitate a client to invoke a service and then execute different functions.

### 23) What is the usage of Service Transport Layer in Web service protocol stack?

The Service Transport Layer is used to transport messages between applications.

This layer includes Hypertext Transport Protocol (HTTP), Simple Mail Transfer Protocol (SMTP), File Transfer Protocol (FTP), and newer protocols like Blocks Extensible Exchange Protocol (BEEP).

### 24) What is the usage of Service Description layer in Web Service Protocol Stack?

The Service Description layer is used to describe the public interface to a specific web service. Currently, service description is handled via the Web Service Description Language (WSDL).

### 25) What is the usage of Service Discovery layer in Web Service Protocol Stack?

The Service Discovery layer is used for centralizing services into a universal registry and providing easy publish/find functionality.

Currently, service discovery is handled via Universal Description, Discovery, and Integration (UDDI).

### 26) What is a remote procedure call (RPC)?

The Remote procedure calls refer to the calls made to the methods which are hosted by related web service.

### 27) What is meant by SOAP message?

The SOAP message refers to the data sent to the application from web services. SOAP message is an XML document which is sent through web services to provide data to the client application written in any programming language.

SOAP message sends via using hypertext transfer protocol.

### 28) What is the need of <Envelope> element in the SOAP document?

The <Envelope> element is used as the root element of every SOAP message.

The Root element is known as the first element in the XML Document.

The envelope, in turn, separated into two parts. One is the header part and second is the body part. The header contains the routing data which stores the source and destination address of the client. So the body includes the actual data.

### 29) Explain web service protocol stack and its layers?

The web services consist of four layers, as mentioned below:

**Service transport:**

This layer is the first layer in the web services protocol stack used in transporting XML files between various clients applications. Protocols used in the layer is as follows:

* HTTP (Hypertext transfer protocol)
* SMTP (Simple Mail Transfer Protocol)
* FTP (File Transfer Protocol)
* BEEP (Block Extensible Exchange Protocol)

**XML Messaging:**

This layer is the second layer in the web services protocol stack based on XML model where messages are encoded in common XML format which can be understandable to other client applications. This layer includes the following protocols:

* XML - RPC
* SOAP (Simple Object Access Protocol)

**Service Description:**

This layer provides the service description to the public interface like the location of web service, Available functions, And the data types for XML messaging. This layer only includes one language:

* WSDL: WSDL stands for Web Service Description Language.

**Service Discovery:**

This layer in the Web Services protocol stack is used to publish or finding web services over the web. This layer includes:

UDDI (Universal Description, Discovery, and integration).

### 30) Explain web service architecture?

The web service framework includes three different layers.

The roles of these layers are:

* **Service Provider**: Role of Service provider is to make the web service which makes it accessible to the client applications over the Web.
* **Service Requestor**: Service requestor refers to any consumer of web service like any client application. Client applications are written in any language contact web service for any functionality by sending XML request over the available network connection.
* **Service Registry**: Service Registry is the centralized directory System which helps to locate the web services for client applications. Used to find the existing web services, as well as developers, can also create the brand new one web service also.

The Service Provider uses the interface named as ?Publish? interface of Service Registry to make the existing web services available to client applications. With all the information provided by the service registry, service requestor able to find or invoke services.

### 31) What is XML-RPC?

The RPC is Remote Procedure Call. It is the method used for calling a procedure or function available on any remote computer on the web.

XML-RPC refers to a simple protocol used to perform RPCs by using XML messaging. It is an excellent tool for connecting different environments and also establishing connections between wide varieties of computers.

### 32) Explain BEEP?

The BEEP stands for Blocks Extensible Exchange Protocol. BEEP is an alternative to HTTP and FTP. BEEP is determined as building new protocols for the variety of applications such as instant messaging, network management, file transfer. It is termed as new Internet Engineering Task Force (IETF) which is layered directly over TCP.

Some of the Build-in features of BEEP protocol are listed below:

* Authentication
* Security
* Error handling
* Initial Handshake Protocol

### 33) What are the requirements to access a Web Service?

The requirement for accessing web services from any application is that should support XML-based request and response. Hence there is no need to install any app for accessing web services.

### 34) Which language does UDDI use?

The UDDI uses the language known as WSDL (Web Service Description Language).

### 35) Explain different HTTP methods supported by RESTful web services?

Enlisted below are some common HTTP methods along with their functions that are supported by RESTful web services.

* **GET**: Read-only access to the resource.
* **PUT**: Creation of new resource.
* **DELETE**: Removal of a resource.
* **POST**: Update of an existing resource.
* **OPTIONS**: Get supported operations on the resource.
* **HEAD**: Returns HTTP header only, nobody.

### 36) What are the steps involved in accessing a web service?

These are the steps involved in accessing a web service:

[Learn more](https://aax-eu.amazon-adsystem.com/x/c/QgtUiGA9TEiB3h_QKStKeCEAAAF_nTX7TAMAAAalBKj-YWQ/https:/www.amazon.de/gp/product/B09S8FKYFP?tag=ms-de-21&ref=aap_588963350312460649" \t "_blank)

volume is gedempt

1. Client application bundled the information and into a SOAP message.
2. SOAP message sends to the server as a body of Hyper-Text markup language using POST method.
3. Web service unpacks the SOAP message and converts it into a command understandable by the application.
4. Application processes the information and in turn bundled the info and send it back to the client as a SOAP message.
5. A Client then unpacks the SOAP message to obtain the results.

### 37) How many Communication protocols can be used to implement a SOAP message? Is SOAP messages are tied to any protocol?

Communication protocol refers to the protocols which were used to transmit information over the web. By using Transport protocols, applications from the different background can quickly communicate with each other without knowing the inside functioning of the various systems. HTTP (Hyper-Text Transfer Protocol) can be used to implement a SOAP message whereas FTP (File Transfer Protocol) can be used as the reliable transport mechanism. SMTP and BEEP can also be used for transport mechanism.

SOAP message is not tied to any protocol. It can use any of the open Transport protocol.

### 38) How are the terms "Platform independent" and "Diverse Application" are related to each other in the context of XML-RPC?

The terms "Platform independent" and "Diverse Application" were related to each other because XML-RPC uses HTTP for transporting SOAP messages over the web. The HTTP is a universal standard protocol for exchanging information on the Web. Hence, it leads to Cross Platform support/ Platform independent. So because it is Platform independent, it leads to the diverse application capable of accessing the web services.

### 39) Explain the role of web service provider/ Publisher.

The role of a Web Service provider is to implement web service and make it available to the web service requestor/ consumer.

### 40) Explain the role of web service requestor/ consumer.

The role of Web Service Requestor / Consumer is to utilize the pre-existing web service provided by the Web Service Provider/ Publisher. Web Service Requestor/ Consumer request the Web Service provider for the information by sending a SOAP message to the Web Service provider. Then in-Turn Web Service Publisher sends the requested information back to the requestor in the form of a SOAP message.

### 41) Write an example to demonstrate the working of the Web Service Provider.

Here is the example of a Web Service Provider:

1. using System;
2. using System.Web.Services;
3. using System.Xml.Serialization;
5. [WebService(Namespace="http://localhost/MyWebServices/")]
6. public class FirstService : WebService{
7. [WebMethod]
8. public int Add(int a, int b) {
9. return a + b;
10. }
12. [WebMethod]
13. public String SayHello() {
14. return "Hello World";
15. }
16. }

### 42) Explain the difference between Web Service Provider/Publisher and Web Service requestor/Consumer.

As the name suggests Web Service Provider provides the web services to the various application irrespective of their background, and Web Service Requestor/ Consumer as the name suggests is the requestor for the web services.

# Katalon: 61Q

## Definition & Functions of an API (Common Web API Testing interview questions)

### 1. What is an API?

An API (Application Programming Interface) is a software intermediary that enables two applications to communicate with each other. It comprises a number of subroutine definitions, logs, and tools for creating application software.

In an API testing interview, you could be asked to give some API examples, here are the well-known ones: Google Maps API, Amazon Advertising API, Twitter API, YouTube API, etc.

### 2. What are main differences between API and Web Service?

* All Web services are APIs but not all APIs are Web services.
* Web services might not contain all the specifications and cannot perform all the tasks that APIs would perform.
* A Web service uses only three styles of use: SOAP, REST and XML-RPC for communication whereas API may be exposed to in multiple ways.
* A Web service always needs a network to operate while APIs don’t need a network for operation.

### 3. What are the Limits of API Usage?

Many APIs have a certain limit set up by the provider. Thus, try to estimate your usage and understand how that will impact the overall cost of the offering. Whether this will be a problem depends in large part on how data is leveraged. Getting caught by a quota and effectively cut-off because of budget limitations will render the service (and any system or process depending on it) virtually useless.

## Creating an API (Common Web API Testing interview questions)

### 4. What are some architectural styles for creating a Web API?

This is one of the fundamental Web API interview questions. Bellows are four common Web API architectural styles:

* HTTP for client-server communication
* XML/JSON as formatting language
* Simple URI as the address for the services
* Stateless communication

### 5. Who can use a Web API?

Web API can be consumed by any clients which support HTTP verbs such as GET, PUT, DELETE, POST. Since Web API services do not require configuration, they can be easily used by any client. In fact, even portable devices such as mobile devices can easily use Web API, which is undoubtedly the biggest advantage of this technology.

## Testing an API – Top Web API Testing interview questions & answers

### 6. What is API Testing?

[API testing](https://www.katalon.com/api-testing/) is a kind of software testing that determines if the developed APIs meet expectations regarding the functionality, reliability, performance, and security of the application.

### 7. What are the advantages of API Testing?

In an API interview, they are likely to ask about the advantages of API testing. So be prepared with the significant ones such as:

* **Test for Core Functionality:**API testing provides access to the application without a user interface. The core and code-level of functionalities of the application will be tested and evaluated early before the GUI tests. This will help detect the minor issues which can become bigger during the GUI testing.
* ***Time Effective:*** API testing usually is less time consuming than functional GUI testing. The web elements in GUI testing must be polled, which makes the testing process slower. Particularly, API test automation requires less code so it can provide better and faster test coverage compared to GUI test automation. These will result in the cost saving for the testing project.
* **Language-Independent:** In API testing, data is exchanged using XML or JSON. These transfer modes are completely language-independent, allowing users to select any code language when adopting automation testing services for the project.
* ***Easy Integration with GUI:*** API tests enable highly integrable tests, which is particularly useful if you want to perform functional GUI tests after API testing. For instance, simple integration would allow new user accounts to be created within the application before a GUI test started.

### 8. Some common protocols used in API testing?

Many protocols are now available to be used in API testing, such as JMS, REST, HTTP, UDDI and SOAP.

### 9. What is the test environment of API?

Setting up the API’s test environment is not an easy task, so you should have a ready answer if your API testing interview is coming. The test environment of API is a bit complete and requires the configuration of the database and server, depending on the software requirements. No GUI (Graphical User Interface) is available in this test form.

When the installation process is complete, API is verified for the proper operation. Throughout the process, the API called from the original environment is set up with different parameters to study the test results.

### 10. What are principles of an API test design?

The five most important principles of an API test design are:

* Setup: Create objects, start services, initialize data, etc
* Execution: Steps to apply API or the scenario, including logging
* Verification: Oracles to evaluate the result of the execution
* Reporting: Pass, failed or blocked
* Clean up: Pre-test state

### 11. What are the common API testing types?

While there are certainly specialty tests, and no list can be asked to be comprehensive in this realm, most tests fit broadly into these following nine categories that you should remember before attending in an API testing interview.

1. Validation Testing
2. Functional Testing
3. UI testing
4. Load testing
5. Runtime/ Error Detection
6. Security testing
7. Penetration testing
8. Fuzz testing
9. Interoperability and WS Compliance testing

### 12. What is the procedure to perform API testing?

1. Choose the suite to add the API test case
2. Choose the test development mode
3. Demand the development of test cases for the required API methods
4. Configure the control parameters of the application and then test conditions
5. Configure method validation
6. Execute the API test
7. Check test reports and filter API test cases
8. Arrange all API test cases

### 13. What must be checked when performing API testing?

During the API testing process, a request is raised to the API with the known data. This way you can analyze the validation response. While testing an API, you should consider:

* Accuracy of data
* Schema validation
* HTTP status codes
* Data type, validations, order and completeness
* Authorization checks
* Implementation of response timeout
* Error codes in case API returns, and
* Non-functional testing like performance and security testing

### 14. What is the best approach method to perform API testing?

The following factors should be considered when performing API testing:

* Defining the correct input parameters
* Verifying the calls of the mixture of two or more added value parameters
* Defining the basic functionality and scope of the API program
* Writing appropriate API test cases and making use of testing techniques such as equivalence class, boundary value, etc. to check the operability
* Testing case execution
* Comparing the test result with the expected result
* Verifying the API behavior under conditions such as connection to files and so on.

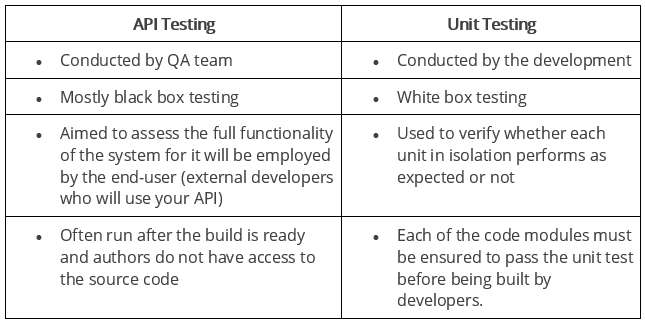
### 15. What are tools could be used for API testing?

There are myriad of different [API testing tools](https://www.katalon.com/resources-center/blog/top-5-free-api-testing-tools/) available. A few of common tools are Katalon Studio, Postman, SoapUi Pro, Apigee, etc.  While doing Unit and API testing, both targets source code. If an API method uses code based in .NET then another supporting tool must have .NET.

Learn more: [SoapUI vs Postman, Katalon Studio: A Review of Top 3 API Tools](https://www.katalon.com/resources-center/blog/soapui-vs-postman-katalon-api-tools/)

[](https://www.katalon.com/)

### 16. What are differences between API Testing and Unit Testing?



### 17. What are differences between API Testing and UI Testing?

* API enables communication between two separate software systems. A software system implementing an API contains functions or subroutines that can be executed by another software system.
* On the other hand, UI ( User Interface) testing refers to testing graphical interface such as how users interact with the applications, testing application elements like fonts, images, layouts etc. UI testing basically focuses on look and feel of an application.

### 18. What are major challenges faced in API testing?

If you can overcome the challenges in API Testing, you can be confident in the API testing interview too. They are:

* Parameter Selection
* Parameter Combination
* Call sequencing
* Output verification and validation
* Another important challenge is providing input values, which is very difficult as GUI is not available in this case.

### 19. What are the testing methods that come under API testing?

One of the most common Web API testing interview questions is about the testing methods. They are:

* Unit testing and Functional testing
* Load testing to test the performance under load
* Discovery testing to list, create and delete the number of calls documented in API
* Usability and Reliability testing to get consistent results
* Security and Penetration testing to validate all types of authentication
* Automation testing to create and run scripts that require regular API calls
* End to end Integration and Web UI testing
* API documentation testing to determine its efficiency and effectiveness

### 20. Why is API testing considered as the most suitable form for Automation testing?

API testing is now preferred over GUI testing and is considered as most suitable because:

* It verifies all the functional paths of the system under test very effectively.
* It provides the most stable interface.
* It is easier to maintain and provides fast feedback.

### 21. What are common API errors that often founded?

Not only API fundamental questions, the interviewer also determine your knowledge and experience by asking about the API errors in a Web API testing interview. So the most common ones are:

* Missing module errors
* Documentation errors
* Parameter validation errors
* And some standard error expectations as if the result is not so predicted then the occurrence of errors can be seen and for the same warnings are specified in the form of a message. There can be one or more warnings within an individual module.

### 22. What kinds of bugs that API testing would often find?

* Missing or duplicate functionality
* Fails to handle error conditions gracefully
* Stress
* Reliability
* Security
* Unused flags
* Not implemented errors
* Inconsistent error handling
* Performance
* Multi-threading issues
* Improper errors

## Documenting the API (Common Web API Testing interview questions)

### 23. What is API documentation?

The API documentation is a complete, accurate technical writing giving instructions on how to effectively use and integrate with an API. It is a compact reference manual that has all the information needed to work with the API, and helps you answer all the API testing questions with details on functions, classes, return types, arguments, and also examples and tutorials.

### 24. What are API documentation templates that are commonly used?

There are several available API documentation templates help to make the entire process simple and straightforward, which could be  answered in your API testing interview, such as:

* Swagger
* Miredot
* Slate
* FlatDoc
* API blueprint
* RestDoc
* Web service API specification

### 25. When writing API document, what must be considered?

* Source of the content
* Document plan or sketch
* Delivery layout
* Information needed for every function in the document
* Automatic document creation programs

### 26. How often are the APIs changed and, more importantly, deprecated?

APIs, especially modern RESTful APIs, are a nice creation that can certainly simplify and accelerate integration efforts, which makes it more likely you will benefit from them. But APIs can and do change for various reasons, sometimes abruptly, and hence REST APIs do not differ from traditional integration methods in this respect. If an API call is obsolete and disappears, your procedure will interrupt and it is important to understand how often the APIs you depend on change or are deprecated.

## REST (Common Web API Testing interview questions)

### 27. What is REST?

REST (Representational State Transfer) is an architectural style for developing web services which exploit the ubiquity of HTTP protocol and uses HTTP method to define actions. It revolves around resource where every component being a resource that can be accessed through a shared interface using standard HTTP methods.  
  
In REST architecture, a REST Server provides access to resources and REST client accesses and makes these resources available. Here, each resource is identified by URIs or global IDs, and REST uses multiple ways to represent a resource, such as text, JSON, and XML. XML and JSON are nowadays the most popular representations of resources.

### 28. What is a RESTFul Web Services?

Mostly, there are two kinds of Web Services which should be remembered in your next API testing interview:

1. SOAP (Simple Object Access Protocol) – an XML-based method to expose web services.
2. Web services developed in the REST style are referred to as RESTful web services. These web services use HTTP methods to implement the concept of REST architecture. A RESTful web service usually defines a URI, Uniform Resource Identifier a service, provides resource representation like JSON and a set of HTTP methods.

### 29. What is a “Resource” in REST?

REST architecture treats any content as a resource, which can be either text files, HTML pages, images, videos or dynamic business information.  
REST Server gives access to resources and modifies them, where each resource is identified by URIs/ global IDs.

### 30. What is the most popular way to represent a resource in REST?

REST uses different representations to define a resource like text, JSON, and XML.  
XML and JSON are the most popular representations of resources.

### 31. Which protocol is used by RESTful Web services?

RESTful web services use the HTTP protocol as a medium of communication between the client and the server.

### 32. What are some key characteristics of REST?

Key characteristics of REST are likely asked in a Web API Testing interview. So please get the answer ready in your mind with these 2 ones:

* REST is stateless, therefore the SERVER has no status (or session data)  
  With a well-applied REST API, the server could be restarted between two calls, since all data is transferred to the server
* Web service uses POST method primarily to perform operations, while REST uses GET for accessing resources.

### 33. What is messaging in RESTful Web services?

RESTful web services use the HTTP protocol as a communication tool between the client and the server. The technique that when the client sends a message in the form of an HTTP Request, the server sends back the HTTP reply is called Messaging. These messages comprise message data and metadata, that is, information on the message itself.

### 34. What are the core components of an HTTP request?

An HTTP request contains five key elements:

1. An action showing HTTP methods like GET, PUT, POST, DELETE.
2. Uniform Resource Identifier (URI), which is the identifier for the resource on the server.
3. HTTP Version, which indicates HTTP version, for example-HTTP v1.1.
4. Request Header, which carries metadata (as key-value pairs) for the HTTP Request message. Metadata could be a client (or browser) type, format supported by the client, format of a message body format, cache settings, and so on.
5. Request Body, which indicates the message content or resource representation.

### 35. What are the most commonly used HTTP methods supported by REST?

* GET is only used to request data from a specified resource. Get requests can be cached and bookmarked. It remains in the browser history and haS length restrictions. GET requests should never be used when dealing with sensitive data.
* POST is used to send data to a server to create/update a resource. POST requests are never cached and bookmarked and do not remain in the browser history.
* PUT replaces all current representations of the target resource with the request payload.
* DELETE removes the specified resource.
* OPTIONS is used to describe the communication options for the target resource.
* HEAD asks for a response identical to that of a GET request, but without the response body.

### 36. Can GET request to be used instead of PUT to create a resource?

The PUT or POST method should be used to create a resource. GET is only used to request data from a specified resource.

### 37. Is there any difference between PUT and POST operations?

PUT and POST operation are quite similar, except the terms of the result generated by them.

PUT operation is idempotent, so you can cache the response while the responses to POST operation are not cacheable, and if you retry the request N times, you will end up having N resources with N different URIs created on server.

In a Web API Testing interview, you should give a specific example for PUT and POST operations to make crystal clear to the interviewer. Below is an example:

Scenario: Let’s say we are designing a network application. Let’s list down few URIs and their purpose to get to know when to use POST and when to use PUT operations.  
  
GET /device-management/devices : Get all devices  
POST /device-management/devices : Create a new device  
  
GET /device-management/devices/{id} : Get the device information identified by “id”  
PUT /device-management/devices/{id} : Update the device information identified by “id”  
DELETE /device-management/devices/{id} : Delete device by “id”

### 38. Which purpose does the OPTIONS method serve for the RESTful Web services?

The OPTIONS Method lists down all the operations of a web service supports. It creates read-only requests to the server.

### 39. What is URI? What is the main purpose of REST-based web services and what is its format?

URI stands for Uniform Resource Identifier. It is a string of characters designed for unambiguous identification of resources and extensibility via the URI scheme.

The purpose of a URI is to locate a resource(s) on the server hosting of the web service.

A URI’s format is <protocol>://<service-name>/<ResourceType>/<ResourceID>.

### 40. What is payload in RESTFul Web services?

The “payload” is the data you are interested in transporting. This is differentiated from the things that wrap the data for transport like the HTTP/S Request/Response headers, authentication, etc.

### 41. What is the upper limit for a payload to pass in the POST method?

<GET> appends data to the service URL. But, its size shouldn’t exceed the maximum URL length. However, <POST> doesn’t have any such limit.

So, theoretically, a user can pass unlimited data as the payload to POST method. But, if we consider a real use case, then sending POST with large payload will consume more bandwidth. It’ll take more time and present performance challenges to your server. Hence, a user should take action accordingly.

### 42. What is the caching mechanism?

Caching is just the practice of storing data in temporarily and retrieving data from a high-performance store (usually memory) either explicitly or implicitly.

When a caching mechanism is in place, it helps improve delivery speed by storing a copy of the asset you requested and later accessing the cached copy instead of the original.

## SOAP (Common Web API Testing interview questions)

### ****43. What are SOAP Web services?****

This is one of the fundamental Web services testing questions that you must know the answer. The SOAP (Simple Object Access Protocol) is defined as an XML-based protocol. It is known for designing and developing web services as well as enabling communication between applications developed on different platforms using various programming languages over the Internet. It is both platform and language independent.

### ****44. How does SOAP work?****

SOAP is used to provide a user interface that can be accessed by the client object, and the request that it sends goes to the server, which can be accessed using the server object. The user interface creates some files or methods consisting of server object and the name of the interface to the server object. It also contains other information such as the name of the interface and methods. It uses HTTP to send the XML to the server using the POST method, which analyzes the method and sends the result to the client. The server creates more XML consisting of responses to the request of user interface using HTTP. The client can use any approach to send the XML, like the SMTP server or POP3 protocol to pass the messages or reply to queries.

### ****45. When to use SOAP API?****

Use the SOAP API to create, retrieve, update or delete records, like accounts, leads, and user-defined objects. With more than 20 different calls, you can also use the SOAP API to manage passwords, perform searches, etc. by using the SOAP API in any language that supports web services.

### ****46. How users utilize the facilities provided by SOAP?****

* PutAddress(): It is used to enter an address in the webpage and has an address instance on the SOAP call.
* PutListing(): It is used to allow the insertion of a complete XML document into the web page. It receives the XML file as an argument and transports the XML file to XML parser liaison, which reads it and inserts it into the SOAP call as a parameter.
* GetAddress(): It is used to get a query name and gets the result that best matches a query. The name is sent to the SOAP call in the form of text character string.
* GetAllListing(): It is used to return the full list in an XML format.

### ****47. What is the major obstacle users faced when using SOAP?****

When using SOAP, users often see the firewall security mechanism as the biggest obstacle. This block all the ports leaving few like HTTP port 80 and the HTTP port used by SOAP that bypasses the firewall. The technical complaint against SOAP is that it mixes the specification for message transport with the specification for message structure.

### ****48. What are the various approaches available for developing****SOAP based****web services?****

There are two different methods available for developing SOAP-based web services, which are explained below:

* Contract-first approach: the contract is first defined by XML and WSDL, and then Java classes are derived from the contract.
* Contract-last approach: Java classes are first defined, and then the contract is generated, which is normally the WSDL file from the Java class.

“Contract-first” method is the most popular approach.

### ****49. What are the elements of a SOAP message structure?****

It is a common XML document that contains the elements as a SOAP message

Envelope: It is an obligatory root element that translates the XML document and defines the beginning and end of the message.

Header: It is an optional item which contains information about the message being sent.

Body: It contains the XML data comprising the message being sent.

Fault: It provides the information on errors that occurred while during message processing.

### ****50. What are the syntax rules for a SOAP message?****

* Must use encoded XML
* Envelope namespace must be used
* Encoding namespace must be used
* Must not consist of a DTD reference
* Must not have XML processing instruction

### ****51. What is the transport method in SOAP?****

Application layer and transport layers of a network are used by SOAP; HTTP and SMTP are the valid protocol of the application layer used as the transport for SOAP. HTTP is more preferable, since it works well with the current Internet infrastructure, in particular with firewalls.  
The SOAP requests can be sent using an HTTP GET method while the specification only contains details about HTTP POST.

### ****52. What are some important characteristics of a SOAP envelope element?****

* SOAP message has a root Envelope element
* Envelope is an obligatory part of the SOAP message.
* If an envelope includes a header element, it should not contain more than one.
* Envelope version will change if the SOAP version changes.
* The SOAP envelope is indicated by the prefix ENV and the envelope element.
* The optional SOAP encoding is also specified using a namespace and the optional encoding style element.

### ****53. What are the major functionalities provided by the SOAP protocol class?****

The SOAP protocol is used to provide simple access methods for all the applications available on the Internet, providing the following functionalities:

* **Call**: A class which provides the main functionality for a remote method for which a call is needed. It is used to create the call() and to specify the encoding style of the registry that will be assigned when if necessary. This call() function is used by the RPC call, which represents the options of the call object.
* **Deployment Descriptor**: A class used to provide the information about the SOAP services. It enables easy deployment without the need for other approaches.
* ***DOM2 Writer***: A class that serializes and uses DOM node as XML string to provide more functionalities.
* ***RPC Message***: A class used as the base class that calls and replies to the request submitted to the server.
* **Service Manager**: A class that provides, lists and then outputs all SOAP services.

### ****54. What are the web relation functionalities provided by SOAP protocol?****

* ***HTTPUtils***: This provides the functionality of the POST method to safely meet the requirements.
* ***Parameter***: It is an argument for an RPC call used by both the client and the server.
* ***Response***: It is an object that represents an RPC reply from both client and server, but the result will not be displayed until after the method call.
* ***TCPTunnel***: It is an object that provides the ability to listen on a specific port and to forward all the host and port names.
* ***TypeConverter***: It helps to convert an object of one type into another type and this is called using the class in the form object.

### ****55. How does the message security model allow the creation of SOAP more secure to use?****

The security model includes the given security tokens. These tokens comprise digital signatures for protection and authentication of SOAP messages. Security tokens can be used to provide the bond between authentication secrets or keys and security identities. Security token uses the authentication protocols and an X.509 certificate to define the relationship between the public key and identity key. The signatures are used to verify the messages and their origin, generate knowledge to confirm the security tokens to bind the identity of a person to the identity of the originator. Security model prevents different attacks and can be used to protect the SOAP architecture.

### ****56. What is the difference between top down & bottom up approach in SOAP Web services?****

* Top down SOAP Web services include creating WSDL document to create a contract between the web service and the client, with a required code as an option. This is also known as Contract-first approach. The top-down approach is difficult to implement because classes must be written to confirm the contract defined in WSDL. One of the benefits of this method is that both client and server code can be written in parallel.
* Bottom up SOAP web services require the code to be written first and then WSDL is generated. It is also known as Contract-last approach. Since WSDL is created based on the code, bottom-up approach is easy to implement and client codes must wait for WSDL from the server side to start working.

### ****57. What are advantages of SOAP?****

* SOAP is both platform and language independent.
* SOAP separates the encoding and communications protocol from the runtime environment.
* Web service can retrieve or receive a SOAP user data from a remote service, and the source’s platform information is completely independent of each other.
* Everything can generate XML, from Perl scripts through C++ code to J2EE app servers.
* It uses XML to send and receive messages.
* It uses standard internet HTTP protocol.
* SOAP runs over HTTP; it eliminates firewall problems. When protocol HTTP is used as the protocol binding, an RPC call will be automatically assigned to an HTTP request, and the RPC response will be assigned to an HTTP reply.
* Compared to RMI, CORBA and DCOM, SOAP is very easy to use.
* SOAP acts as a protocol to move information in a distributed and decentralized environment.
* SOAP is independent of the transport protocol and can be used to coordinate different protocols.

### ****58. What are disadvantages of SOAP?****

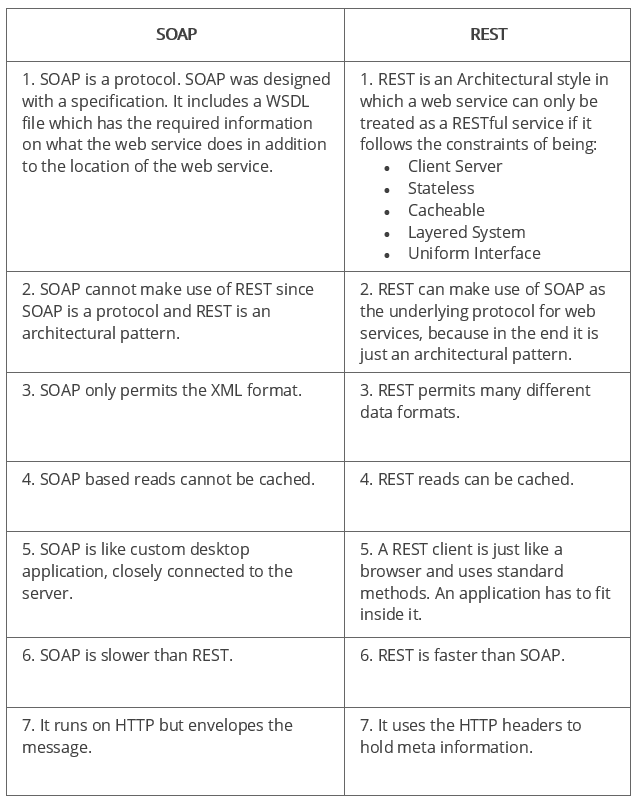
SOAP is typically significantly slower than other types of [middleware](https://searchmicroservices.techtarget.com/definition/middleware) standards, including CORBA, because SOAP uses a detailed XML format. A complete understanding of the performance limitations before building applications around SOAP is hence required.

SOAP is usually limited to pooling and not to event notifications when HTTP is used for the transport. In addition, only one client can use the services of one server in typical situations.

If HTTP is used as the transport protocol, firewall latency usually occurs since the firewall analyzes the HTTP transport. This is because HTTP is also leveraged for Web browsing, and so many firewalls do not understand the difference between using HTTP within a web browser and using HTTP within SOAP.

SOAP has different support levels, depending on the supported programming language. For instance, SOAP supported in [Python](https://whatis.techtarget.com/definition/Python) and PHP is not as powerful as it is in Java and .NET

### 59. What are the differences between SOAP and REST?



### 60. SOAP or Rest APIs, which method to use?

SOAP is the heavyweight choice for Web service access. It provides the following advantages when compared to REST:

* SOAP is not very easy to implement and requires more bandwidth and resources.
* SOAP message request is processed slower as compared to REST and it does not use web caching mechanism.
* WS-Security: While SOAP supports SSL (just like REST) it also supports WS-Security which adds some enterprise security features.
* WS-AtomicTransaction: Need ACID Transactions over a service, you’re going to need SOAP.
* WS-ReliableMessaging: If your application needs Asynchronous processing and a guaranteed level of reliability and security. Rest doesn’t have a standard messaging system and expects clients to deal with communication failures by retrying.
* If the security is a major concern and the resources are not limited then we should use SOAP web services. Like if we are creating a web service for payment gateways, financial and telecommunication related work, then we should go with SOAP as here high security is needed.

REST is easier to use for the most part and is more flexible. It has the following advantages when compared to SOAP:

* Since REST uses standard HTTP, it is much simpler.
* REST is easier to implement, requires less bandwidth and resources.
* REST permits many different data formats whereas SOAP only permits XML.
* REST allows better support for browser clients due to its support for JSON.
* REST has better performance and scalability. REST reads can be cached, SOAP based reads cannot be cached.
* If security is not a major concern and we have limited resources. Or we want to create an API that will be easily used by other developers publicly then we should go with REST.
* If we need Stateless CRUD operations then go with REST.
* REST is commonly used in social media, web chat, mobile services and Public APIs like Google Maps.
* RESTful service returns various MediaTypes for the same resource, depending on the request header parameter “Accept” as application/xml or application/json for POST and /user/1234.json or GET /user/1234.xml for GET.
* REST services are meant to be called by the client-side application and not the end user directly.
* ST in REST comes from State Transfer. You transfer the state around instead of having the server store it, this makes REST services scalable.

### 61. What are the factors that help to decide which style of Web services – SOAP or REST – to use?

Generally, REST is preferred due to its simplicity, performance, scalability, and support for multiple data formats.

However, SOAP is favorable to use where service requires an advanced level of security and transactional reliability.

But you can read the following facts before opting for any of the styles.

* **Does the service expose data or business logic?** REST is commonly used for exposing data while SOAP for logic.
* **The requirement from clients or providers for a formal contract**. SOAP can provide contract via WSDL.
* **Support multiple data formats**.
* **Support for AJAX calls.** REST can apply the XMLHttpRequest.
* **Synchronous and asynchronous calls.**SOAP enables both synchronous/ asynchronous operations whereas REST has built-in support for synchronous.
* **Stateless or Stateful calls.** REST is suited for stateless operations.
* **Security.** SOAP provides a high level of security.
* **Transaction support.** SOAP is good at transaction management.
* **Limited bandwidth**. SOAP has a lot of overhead when sending/receiving packets since it’s XML based, requires a SOAP header. However, REST requires less bandwidth to send requests to the server. Its messages are mostly built using JSON.
* **Ease of use**. REST based application is easy to implement, test, and maintain.

# 10.ToolsQA: 16Q

### *****1. What is REST?*****

REST is an architectural style for developing web services which exploit the ubiquity of HTTP protocol and uses the HTTP method to define actions. It revolves around resource where every component being a resource that can be accessed through a shared interface using standard HTTP methods.

In REST architecture, a REST Server provides access to resources and REST client accesses and makes these resources available. Here, each resource is identified by URIs or global IDs, and REST uses multiple ways to represent a resource, such as text, JSON, and XML. XML and JSON are nowadays the most popular representations of resources.

### *****2. What is a RESTFul Web Services?*****

Mostly, there are two kinds of Web Services which should be remembered in your next API testing interview:

1. ***SOAP*** (Simple Object Access Protocol): An XML-based method to expose web services.
2. ***REST*** (Representational State Transfer): Web services developed in the REST style are referred to as RESTful web services. These web services use HTTP methods to implement the concept of REST architecture. A RESTful web service usually defines a URI, Uniform Resource Identifier a service, provides resource representation like JSON and a set of HTTP methods.

### *****3. What is a “Resource” in REST?*****

REST architecture treats any content as a resource, which can be either text files, HTML pages, images, videos or dynamic business information. REST Server gives access to resources and modifies them, where each resource is identified by URIs/ global IDs.

### *****4. What is the most popular way to represent a resource in REST?*****

REST uses different representations to define a resource like text, JSON, and XML. XML and JSON are the most popular representations of resources.

### *****5. Which protocol is used by RESTful Web services?*****

RESTful web services use the HTTP protocol as a medium of communication between the client and the server.

### *****6. What are some key characteristics of REST?*****

Key characteristics of REST are likely asked in a Web API Testing interview. So please get the answer ready in your mind with these 2 ones:

* REST is stateless, therefore the SERVER has no status (or session data) With a well-applied REST API, the server could be restarted between two calls, since all data is transferred to the server
* Web service uses POST method primarily to perform operations, while REST uses GET for accessing resources.

### *****7. What is messaging in RESTful Web services?*****

RESTful web services use the HTTP protocol as a communication tool between the client and the server. The technique that when the client sends a message in the form of an HTTP Request, the server sends back the HTTP reply is called Messaging. These messages comprise message data and metadata, that is, information on the message itself.

### *****8. What are the core components of an HTTP request?*****

An HTTP request contains five key elements:

1. An action showing HTTP methods like GET, PUT, POST, DELETE.
2. Uniform Resource Identifier (URI), which is the identifier for the resource on the server.
3. HTTP Version, which indicates HTTP version, for example-HTTP v1.1.
4. Request Header, which carries metadata (as key-value pairs) for the HTTP Request message. Metadata could be a client (or browser) type, format supported by the client, format of a message body format, cache settings, and so on.
5. Request Body, which indicates the message content or resource representation.

### *****9. What are the most commonly used HTTP methods supported by REST?*****

* GET is only used to request data from a specified resource. Get requests can be cached and bookmarked. It remains in the browser history and haS length restrictions. GET requests should never be used when dealing with sensitive data.
* POST is used to send data to a server to create/update a resource. POST requests are never cached and bookmarked and do not remain in the browser history.
* PUT replaces all current representations of the target resource with the request payload.
* DELETE removes the specified resource.
* OPTIONS is used to describe the communication options for the target resource.
* HEAD asks for a response identical to that of a GET request, but without the response body.

### *****10. Can GET request to be used instead of PUT to create a resource?*****

The PUT or POST method should not be used to create a resource. You can use the GET operation which has view-only rights.

### *****11. Is there any difference between PUT and POST operations?*****

PUT and POST operation are quite similar, except the terms of the result generated by them. PUT operation is idempotent, so you can cache the response while the responses to POST operation are not cacheable, and if you retry the request N times, you will end up having N resources with N different URIs created on server.

In a Web API Testing interview, you should give a specific example for PUT and POST operations to make crystal clear to the interviewer. Below is an example:

Scenario: Let’s say we are designing a network application. Let’s list down few URIs and their purpose to get to know when to use POST and when to use PUT operations.

GET /device-management/devices : Get all devices POST /device-management/devices : Create a new device

GET /device-management/devices/{id} : Get the device information identified by “***id***” PUT /device-management/devices/{id} : Update the device information identified by “***id***” DELETE /device-management/devices/{id} : Delete device by “***id***”

### *****12. Which purpose does the OPTIONS method serve for the RESTful Web services?*****

The OPTIONS Method lists down all the operations of a web service supports. It creates read-only requests to the server.

### *****13. What is URI? What is the main purpose of REST-based web services and what is its format?*****

URI stands for ***Uniform Resource Identifier.*** It is a string of characters designed for unambiguous identification of resources and extensibility via the URI scheme. The purpose of a URI is to locate a resource(s) on the server hosting of the web service.

A URI’s format is *<protocol>*://*<service-name>*/*<ResourceType>*/*<ResourceID>.*

### *****14. What is payload in RESTFul Web services?*****

The “***payload***” is the data you are interested in transporting. This is differentiated from the things that wrap the data for transport like the HTTP/S Request/Response headers, authentication, etc.

### *****15. What is the upper limit for a payload to pass in the POST method?*****

<GET> appends data to the service URL. But, its size shouldn’t exceed the maximum URL length. However, <POST> doesn’t have any such limit.

So, theoretically, a user can pass unlimited data as the payload to POST method. But, if we consider a real use case, then sending POST with large payload will consume more bandwidth. It’ll take more time and present performance challenges to your server. Hence, a user should take action accordingly.

### *****16. What is the caching mechanism?*****

Caching is just the practice of storing data in temporarily and retrieving data from a high-performance store (usually memory) either explicitly or implicitly.

When a caching mechanism is in place, it helps improve delivery speed by storing a copy of the asset you requested and later accessing the cached copy instead of the original.

# 11.ErrorSea: 25Q

### What is a web service?

Web services are standard software systems that provide a common platform for various applications (written in different programming languages) to communicate over the network. They help in establishing client-server communication.

### What does REST stand for?

REST stands for Representational State Transfer.

### Explain REST and RESTFUL.

A software architectural style that defines a set of rules that are to be used for creating web services is known as REST. REST uses the HTTP Protocol. It, itself, is not a protocol or a standard.

Web services that follow the REST architectural style are known as RESTful web services. A RESTful system consists of:

* A client who requests a particular resource
* A server that has all the resources

**Read Also:** [How to call API in PHP using CURL](https://errorsea.com/how-to-call-api-in-php-using-curl/)

### Define the architectural style for creating a web API.

The architectural style for creating a web API are:

* HTTP for client-server communication
* XML/JSON as the formatting language
* Simple URI as the address for the services
* Stateless Communication

### What are the HTTP methods supported by REST?

The HTTP methods supported by REST are:

* GET:  The GET method retrieves specific information from the server according to the specifications made by the URI.
* POST: The POST method modifies data on that particular server from which the request was sent.
* PUT: The PUT method is used to request that the message body that has been returned is stored under the location mentioned in the HTTP message.
* DELETE: The DELETE method is used to delete the specified resources.
* HEAD: The HEAD method is quite similar to the GET method. The only difference is that the HEAD method returns only the meta information and not the message body.

### Name the tools that are used to test a web API.

The SOAPUI tool is used for SOAP WS and the Firefox ‘poster’ plugin for RESTFUL services. These are the two most popular testing tools.

### What are some of the key characteristics of REST?

* The REST API is stateless. It does not store data like user credentials and needs to be reminded of it with each new session. This feature helps in increasing scalability.
* It supports both JSON and XML, therefore, catering to the varying expectations of the developer community. Giants like Microsoft have implemented this API in their architecture.
* The REST architecture is simpler to implement as compared to SOAP.
* REST can be operated under limited resources and low network bandwidth.

### What is meant by addressing in RESTful web services?

Addressing in RESTful web services refers to locating a resource or multiple resources present on the server. It is similar to locating the postal address of a person.

### What is the use of JAXB in RESTful web API?

JAXB stands for Java Architecture for XML Binding. It is an XML-to-Java binding technology that enables the easy transformation of schema to Java objects and vice versa.

### List the different application integration styles in real-time systems.

The different integration styles are:

* Directly accessing or sharing the database
* Batch file transfer or sharing
* Invoking remote procedure (RPC)
* Swapping asynchronous messages over a message-oriented middleware (MOM).

### What is API Testing?

API testing is quite similar to software testing. It is used to determine if the developed APIs meet the expectations of that particular application’sREST On functionality, reliability, performance, and security of that particular application.

**Read Also:** [Top 25 HR Interview Questions and Answers](https://errorsea.com/top-25-hr-interview-questions-and-answers/)

### Mention some protocols used in API Testing.

Some commonly used protocols in API Testing are JMS, REST, HTTP, UDDI, and SOAP.

### What are the common types of API Testing?

There is no fixed list as there is speciality testing as it depends upon the developer’s expectations and requirements. However, most tests can be broadly categorized into 9 types:

* Validation Testing
* Functional Testing
* UI Testing
* Load Testing
* Runtime/ Error Detection
* Security Testing
* Penetration Testing
* Fuzz Testing
* Interoperability and WS Compliance testing

### Which are the parameters that must be checked when performing API testing?

In the process of API testing, a request is raised to the API with the known data. It helps in analyzing the response. So, while testing the API, the following parameters must be kept in mind:

* Data Accuracy
* Schema Validation
* HTTP Status Code
* Data type, validations, order, and completeness
* Authorization checks
* Implementation of response timeout
* Error codes in case API returns, and
* Non-functional testing like performance and security testing

### List a few testing tools for web services for REST API.

* Spring REST web service using MVC
* Jersey API
* CXF
* Axis
* Restlet

### Differentiate between SOAP and REST.

|  |  |
| --- | --- |
| SOAP | REST |
| SOAP (Simple Object Access Protocol) is a protocol through which two systems communicate by sharing XML documents. | REST (Representational State Transfer) is a service architecture and design for network-based software architectures. |
| SOAP only supports XML. | REST, on the other hand, supports many different formats of data. |
| The reads on SOAP cannot be cached. | The REST reads are cachable. |
| SOAP runs on HTTP but envelopes the message. | REST uses the HTTP headers to hold meta information. |
| SOAP cannot use REST since it itself is a protocol while REST is an architectural pattern. | On the other hand, REST can make use of SOAP as the underlying protocol for web services. |
| SOAP is slower than REST | REST is faster than SOAP |

### Specify the format of a URI in REST architecture.

The format of a URI is as follows:

📋

<protocol>://<service-name>/<ResourceType>/<ResourceID>

### Which web services method is available on read-only mode?

The operations under the GET method are read-only and are safe.

### What is a “Resource” in REST?

REST considers all types of content as a resource; they can be text files, HTML pages, images, videos, or even dynamic business information. The REST server gives access to the resources and modifies them according to the different HTTP methods specified. Each resource is identified by URIs/ global IDs.

### Which is the most popular way of representing a resource in REST?

Different representations are used by the REST architecture to define a particular resource. XML(Extensible Markup Language) and JSON(JavaScript Object Notation) are the most popular representations of resources.

### Explain XML and JSON in brief.

XML:

* The Extensible Markup Language is quite similar to HTML. It is designed to store and transport data. One of its biggest advantages over HTML is that it allows users to create their own custom tags. XML is generally used to represent structured information like documents, data, configuration, etc.

JSON:

* JavaScript Object Notation is a lightweight format that has been designed to store and transport data. It represents structured data based on JavaScript object syntax. It is fast and easy to use and understand.

### Which are the safe REST operations?

REST works on the HTTP protocol and hence, uses the HTTP methods to carry out its operations. The HTTP methods that do not change or modify the resources at the server are known as safe operations. The GET method returns the resource only in the read-only method, and the HEAD method returns the metadata of the resource and is, therefore, safe. The methods PUT, POST, and DELETE, on the other hand, modify the data at the server and are, therefore, considered unsafe.

### Name the method that validates all controls on a page.

The Page.Validate() method validates all controls on a page.

### Which library is used for JSON serialization in Web APIs?

The JSON.NET library is used for JSON serialization.

### Who are the consumers of Web API?

Web APIs cater to a broad range of clients including browsers, and mobile devices. It is also used along with native applications that require web services but don’t support SOAP. Clients that support the HTTP methods like GET, DELETE, POST, can also use Web APIs.

# 12. WisdomJobs: 50Q

1. **Question 1. What Rest Stands For?**

**Answer :**

REST stands for REpresentational State Transfer.

1. **Question 2. What Is Rest?**

**Answer :**

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. REST was first introduced by Roy Fielding in 2000.

In REST architecture, a REST Server simply provides access to resources and REST client accesses and presents the resources. Here each resource is identified by URIs/ global IDs. REST uses various representations to represent a resource like text, JSON and XML. Now a days JSON is the most popular format being used in web services.

[Web Service Testing Interview Questions](https://www.wisdomjobs.com/e-university/web-service-testing-interview-questions.html)

1. **Question 3. Name Some Of The Commonly Used Http Methods Used In Rest Based Architecture?**

**Answer :**

**Following well known HTTP methods are commonly used in REST based architecture:−**

* + **GET −**Provides a read only access to a resource.
  + **PUT** − Used to create a new resource.
  + **DELETE** − Ued to remove a resource.
  + **POST** − Used to update a existing resource or create a new resource.
  + **OPTIONS** − Used to get the supported operations on a resource.

**Question 4. What Are Web Services?**

**Answer :**

A web service is a collection of open protocols and standards used for exchanging data between 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.

[Web Service Testing Tutorial](https://www.wisdomjobs.com/e-university/web-service-testing-tutorial-178.html" \o "Web Service Testing Tutorial)

1. **Question 5. What Are Restful Web Services?**

**Answer :**

Web services based on REST Architecture are known as RESTful web services. These web services use HTTP methods to implement the concept of REST architecture. A RESTful web service usually defines a URI, Uniform Resource Identifier a service, provides resource representation such as JSON and set of HTTP Methods.

[Web Services Interview Questions](https://www.wisdomjobs.com/e-university/web-services-interview-questions.html" \o "Web Services Interview Questions)

1. **Question 6. What Is A Resource In Rest?**

**Answer :**

REST architecture treats every content as a resource. These resources can be text files, html pages, images, videos or dynamic business data. REST Server simply provides access to resources and REST client accesses and modifies the resources. Here each resource is identified by URIs/ global IDs.

1. **Question 7. How To Represent A Resource In Rest?**

**Answer :**

REST uses various representations to represent a resource where text, JSON, XML. XML and JSON are the most popular representations of resources.

[Web Services Tutorial](https://www.wisdomjobs.com/e-university/web-services-tutorial-179.html" \o "Web Services Tutorial) [Core Java Interview Questions](https://www.wisdomjobs.com/e-university/core-java-interview-questions.html" \o "Core Java Interview Questions)

1. **Question 8. What Are The Best Practices To Design A Resource Representation?**

**Answer :**

**Following are important points to be considered while designing a representation format of a resource in a RESTful web services:**

* + **Understandability** − Both Server and Client should be able to understand and utilize the representation format of the resource.
  + **Completeness** − Format should be able to represent a resource completely. For example, a resource can contain another resource. Format should be able to represent simple as well as complex structures of resources.
  + **Linkablity** − A resource can have a linkage to another resource, a format should be able to handles such situations.

1. **Question 9. Which Protocol Is Used By Restful Web Services?**

**Answer :**

RESTful web services make use of HTTP protocol as a medium of communication between client and server.

[Java-Springs Interview Questions](https://www.wisdomjobs.com/e-university/java-springs-interview-questions.html" \o "Java-Springs Interview Questions)

1. **Question 10. What Is Messaging In Restful Web Services?**

**Answer :**

A client sends a message in form of a HTTP Request and server responds in form of a HTTP Response. This technique is termed as Messaging. These messages contain message data and metadata i.e. information about message itself.

[Core Java Tutorial](https://www.wisdomjobs.com/e-university/core-java-tutorial-231.html" \o "Core Java Tutorial)

1. **Question 11. What Are The Core Components Of A Http Request?**

**Answer :**

**A HTTP Request has five major parts :**

* + **Verb −** Indicate HTTP methods such as GET, POST, DELETE, PUT etc.
  + **URI −** Uniform Resource Identifier (URI) to identify the resource on server.
  + **HTTP Version −** Indicate HTTP version, for example HTTP v1.1 .
  + **Request Header** − Contains metadata for the HTTP Request message as key-value pairs. For example, client ( or browser) type, format supported by client, format of message body, cache settings etc.
  + **Request Body −** Message content or Resource representation.

[CSS Interview Questions](https://www.wisdomjobs.com/e-university/css-interview-questions.html" \o "CSS Interview Questions)

1. **Question 12. What Are The Core Components Of A Http Response?**

**Answer :**

**A HTTP Response has four major parts:-**

* + **Status/Response Code** − Indicate Server status for the requested resource. For example 404 means resource not found and 200 means response is ok.
  + **HTTP Version** − Indicate HTTP version, for example HTTP v1.1 .
  + **Response Heade**r − Contains metadata for the HTTP Response message as key-value pairs. For example, content length, content type, response date, server type etc.
  + **Response Body** − Response message content or Resource representation.

[Web Service Testing Interview Questions](https://www.wisdomjobs.com/e-university/web-services-practice-tests-179-327625" \o "Web Service Testing Interview Questions)

1. **Question 13. What Is Addressing In Restful Web Services?**

**Answer :**

Addressing refers to locating a resource or multiple resources lying on the server. It is analogous to locate a postal address of a person.

[Java-Springs Tutorial](https://www.wisdomjobs.com/e-university/java-springs-tutorial-287.html" \o "Java-Springs Tutorial)

1. **Question 14. What Is Uri?**

**Answer :**

URI stands for Uniform Resource Identifier. Each resource in REST architecture is identified by its URI.

1. **Question 15. What Is Purpose Of A Uri In Rest Based Web Services?**

**Answer :**

Purpose of an URI is to locate a resource(s) on the server hosting the web service.

[Java Abstraction Interview Questions](https://www.wisdomjobs.com/e-university/java-abstraction-interview-questions.html" \o "Java Abstraction Interview Questions)

1. **Question 16. What Is Format Of A Uri In Rest Architecture?**

**Answer :**

**A URI is of following format:−**

<protocol>://<service-name>/<ResourceType>/<ResourceID>

[CSS Tutorial](https://www.wisdomjobs.com/e-university/css-tutorial-1198.html" \o "CSS Tutorial)

1. **Question 17. What Is The Purpose Of Http Verb In Rest Based Web Services?**

**Answer :**

VERB identifies the operation to be performed on the resource.

[Amazon Web Services (AWS) Interview Questions](https://www.wisdomjobs.com/e-university/amazon-web-services-aws-interview-questions-answers.html" \o "Amazon Web Services (AWS) Interview Questions)

1. **Question 18. What Are The Best Practices To Create A Standard Uri For A Web Service?**

**Answer :**

**Following are important points to be considered while designing a URI** −

* + **Use Plural Noun** − Use plural noun to define resources. For example, we've used users to identify users as a resource.
  + **Avoid using spaces** − Use underscore(\_) or hyphen(-) when using a long resource name, for example, use authorized\_users instead of authorized%20users.
  + **Use lowercase letters** − Although URI is case-insensitive, it is good practice to keep url in lower case letters only.
  + **Maintain Backward Compatibility** − As Web Service is a public service, a URI once made public should always be available. In case, URI gets updated, redirect the older URI to new URI using HTTP Status code, 300.
  + **Use HTTP Verb** − Always use HTTP Verb like GET, PUT, and DELETE to do the operations on the resource. It is not good to use operations names in URI.

[Web Services Interview Questions](https://www.wisdomjobs.com/e-university/web-services-interview-questions.html" \o "Web Services Interview Questions)

1. **Question 19. What Is Statelessness In Restful Web Services?**

**Answer :**

As per REST architecture, a RESTful web service should not keep a client state on server. This restriction is called statelessness. It is responsibility of the client to pass its context to server and then server can store this context to process client's further request. For example, session maintained by server is identified by session identifier passed by the client.

[Restful web service Tutorial](https://www.wisdomjobs.com/e-university/restful-web-service-tutorial-1255.html" \o "Restful web service Tutorial)

1. **Question 20. What Are The Advantages Of Statelessness In Restful Web Services?**

**Answer :**

**Following are the benefits of statelessness in RESTful web services:**

* + Web services can treat each method request independently.
  + Web services need not to maintain client's previous interactions. It simplifies application design.
  + As HTTP is itself a statelessness protocol, RESTful Web services work seamlessly with HTTP protocol.

[RESTful JAVA Web Services Interview Questions](https://www.wisdomjobs.com/e-university/restful-java-web-services-interview-questions.html" \o "RESTful JAVA Web Services Interview Questions)

1. **Question 21. What Are The Disadvantages Of Statelessness In Restful Web Services?**

**Answer :**

**Following is the disadvantage of statelessness in RESTful web services:-**

Web services need to get extra information in each request and then interpret to get the client's state in case client interactions are to be taken care of.

1. **Question 22. What Do You Mean By Idempotent Operation?**

**Answer :**

Idempotent operations means their result will always same no matter how many times these operations are invoked.

[Spring MVC Framework Tutorial](https://www.wisdomjobs.com/e-university/spring-mvc-framework-tutorial-1806.html" \o "Spring MVC Framework Tutorial)

1. **Question 23. Which Type Of Web Services Methods Are To Be Idempotent?**

**Answer :**

PUT and DELETE operations are idempotent.

[Spring MVC Framework Interview Questions](https://www.wisdomjobs.com/e-university/spring-mvc-framework-interview-questions.html" \o "Spring MVC Framework Interview Questions)

1. **Question 24. Which Type Of Web Services Methods Are To Be Read Only?**

**Answer :**

GET operations are read only and are safe.

[Core Java Interview Questions](https://www.wisdomjobs.com/e-university/core-java-interview-questions.html" \o "Core Java Interview Questions)

1. **Question 25. What Is The Difference Between Put And Post Operations?**

**Answer :**

PUT and POST operation are nearly same with the difference lying only in the result where PUT operation is idempotent and POST operation can cause different result.

1. **Question 26. What Should Be The Purpose Of Options Method Of Restful Web Services?**

**Answer :**

It should list down the supported operations in a web service and should be read only.

[Soap Web Services Interview Questions](https://www.wisdomjobs.com/e-university/soap-web-services-interview-questions.html" \o "Soap Web Services Interview Questions)

1. **Question 27. What Should Be The Purpose Of Head Method Of Restful Web Services?**

**Answer :**

It should return only HTTP Header, no Body and should be read only.

[Java-Springs Interview Questions](https://www.wisdomjobs.com/e-university/java-springs-interview-questions.html" \o "Java-Springs Interview Questions)

1. **Question 28. What Is Caching?**

**Answer :**

Caching refers to storing server response in client itself so that a client needs not to make server request for same resource again and again. A server response should have information about how a caching is to be done so that a client caches response for a period of time or never caches the server response.

1. **Question 29. Which Header Of Http Response, Provides The Date And Time Of The Resource When It Was Created?**

**Answer :**

Date header provides the date and time of the resource when it was created.

1. **Question 30. Which Header Of Http Response, Provides The Date And Time Of The Resource When It Was Last Modified?**

**Answer :**

Last Modified header provides the date and time of the resource when it was last modified.

1. **Question 31. Which Header Of Http Response Provides Control Over Caching?**

**Answer :**

Cache-Control is the primary header to control caching.

1. **Question 32. Which Header Of Http Response Sets Expiration Date And Time Of Caching?**

**Answer :**

Expires header sets expiration date and time of caching.

1. **Question 33. Which Directive Of Cache Control Header Of Http Response Indicates That Resource Is Cacheable By Any Component?**

**Answer :**

Public directive indicates that resource is cacheable by any component.

[CSS Interview Questions](https://www.wisdomjobs.com/e-university/css-interview-questions.html" \o "CSS Interview Questions)

1. **Question 34. Which Directive Of Cache Control Header Of Http Response Indicates That Resource Is Catchable By Only Client And Server, No Intermediary Can Cache The Resource?**

**Answer :**

Private directive indicates that resource is cachable by only client and server, no intermediary can cache the resource.

1. **Question 35. Which Directive Of Cache Control Header Of Http Response Indicates That Resource Is Not Cacheable?**

**Answer :**

no-cache/no-store directive indicates that resource is not cacheable.

1. **Question 36. Which Directive Of Cache Control Header Of Http Response Can Set The Time Limit Of Caching?**

**Answer :**

max-age directive indicates that the caching is valid up to max-age in seconds. After this, client has to make another request.

[Java Abstraction Interview Questions](https://www.wisdomjobs.com/e-university/java-abstraction-interview-questions.html" \o "Java Abstraction Interview Questions)

1. **Question 37. Which Directive Of Cache Control Header Of Http Response Provides Indication To Server To Revalidate Resource If Max-age Has Passed?**

**Answer :**

must-revalidate directive provides indication to server to revalidate resource if max-age has passed.

1. **Question 38. What Are The Best Practices For Caching?**

**Answer :**

Always keep static contents like images, css, JavaScript cacheable, with expiration date of 2 to 3 days. Never keep expiry date too high.  
Dynamic contents should be cached for few hours only.

1. **Question 39. What Are The Best Practices To Be Followed While Designing A Secure Restful Web Service?**

**Answer :**

As RESTful web services work with HTTP URLs Paths so it is very important to safeguard a RESTful web service in the same manner as a website is be secured. Following are the best practices to be followed while designing a RESTful web service:

* + **Validation** − Validate all inputs on the server. Protect your server against SQL or NoSQL injection attacks.
  + **Session based authentication** − Use session based authentication to authenticate a user whenever a request is made to a Web Service method.
  + **No sensitive data in URL** − Never use username, password or session token in URL , these values should be passed to Web Service via POST method.
  + **Restriction on Method execution** − Allow restricted use of methods like GET, POST, DELETE. GET method should not be able to delete data.
  + **Validate Malformed XML/JSON** − Check for well formed input passed to a web service method.
  + **Throw generic Error Messages** − A web service method should use HTTP error messages like 403 to show access forbidden etc.

1. **Question 40. What Is The Purpose Of Http Status Code?**

**Answer :**

HTTP Status code are standard codes and refers to predefined status of task done at server. For example, HTTP Status 404 states that requested resource is not present on server.

[Amazon Web Services (AWS) Interview Questions](https://www.wisdomjobs.com/e-university/amazon-web-services-aws-interview-questions-answers.html" \o "Amazon Web Services (AWS) Interview Questions)

1. **Question 41. What Http Status Code 200 States?**

**Answer :**

It means, OK, shows success.

1. **Question 42. What Http Status Code 201 States?**

**Answer :**

It means, CREATED, when a resource is successful created using POST or PUT request. Return link to newly created resource using location header.

[RESTful JAVA Web Services Interview Questions](https://www.wisdomjobs.com/e-university/restful-java-web-services-interview-questions.html" \o "RESTful JAVA Web Services Interview Questions)

1. **Question 43. What Http Status Code 204 States?**

**Answer :**

It means, NO CONTENT, when response body is empty for example, a DELETE request.

1. **Question 44. What Http Status Code 304 States?**

**Answer :**

It means, NOT MODIFIED, used to reduce network bandwidth usage in case of conditional GET requests. Response body should be empty. Headers should have date, location etc.

1. **Question 45. What Http Status Code 400 States?**

**Answer :**

It means, BAD REQUEST, states that invalid input is provided e.g. validation error, missing data.

1. **Question 46. What Http Status Code 401 States?**

**Answer :**

It means, FORBIDDEN, states that user is not having access to method being used for example, delete access without admin rights.

1. **Question 47. What Http Status Code 404 States?**

**Answer :**

It means, NOT FOUND, states that method is not available.

1. **Question 48. What Http Status Code 409 States?**

**Answer :**

It means, CONFLICT, states conflict situation while executing the method for example, adding duplicate entry.

1. **Question 49. What Http Status Code 500 States?**

**Answer :**

It means, INTERNAL SERVER ERROR, states that server has thrown some exception while executing the method.

1. **Question 50. What Is Jax-rs?**

**Answer :**

JAX-RS stands for JAVA API for RESTful Web Services. JAX-RS is a JAVA based programming language API and specification to provide support for created RESTful Webservices. Its 2.0 version was released in 24 May 2013. JAX-RS makes heavy use of annotations available from Java SE 5 to simplify development of JAVA based web services creation and deployment. It also provides supports for creating clients for RESTful web services.