1. Explain REST and RESTFUL?

REST (REpresentational State Transfer) is an architectural style for developing web services. REST is popular due to its simplicity and the fact that it builds upon existing systems and features of the internet's Hypertext Transfer Protocol (HTTP) in order to achieve its objectives, as opposed to creating new standards, frameworks and technologies.

A RESTful API is an application program interface (API) that uses HTTP requests to GET, PUT, POST and DELETE data. A RESTful API -- also referred to as a RESTful web service or REST API -- is based on representational state transfer (REST), an architectural style and approach to communications often used in web services development.

2. Mention what are the HTTP methods supported by REST?

The following subset of HTTP methods are supported for the REST:

* GET: - The GET method retrieves specific information from the server as identified by the request URI.
* PUT: - The PUT method requests that the message body sent with the request be stored under the location provided in the HTTP message.
* DELETE: - The DELETE method deletes the specified resources.
* POST: - The POST method modifies data on the server from which a request was sent.
* HEAD: - The HEAD method is similar to the GET method except the message body is not returned in the response. The response only includes metainformation, such as a response code or corresponding headers.

3. Explain the architectural style for creating web API.

The Web API architecture style is an hybrid style derived from both the REST and the RPC styles.

RESTful API Style: - REST (Representational State Transfer) is an architectural style for services, and as such it defines a set of architectural constraints and agreements. A service, which complies with the REST constraints, is said to be RESTful. REST is designed to make optimal use of an HTTP-based infrastructure and due to the success of the web, HTTP-based infrastructures, such as servers, caches, and proxies, are widely available. The web, which is based on HTTP, provides some proof for an architecture that not only scales extremely well but also has longevity. The basic idea of REST is to transfer the ideas that worked well for the web and apply them to web services.

RPC Style: - RPC is an abbreviation for Remote Procedure Call. RPC is an architectural style for distributed systems. It has been around since the 1980s. Today the most widely used RPC styles are JSON-RPC and XML-RPC. Even SOAP can be considered to follow an RPC architectural style. The central concept in RPC is the procedure. The procedures do not need to run on the local machine, but they can run on a remote machine within the distributed system. When using an RPC framework, calling a remote procedure should be as simple as calling a local procedure.

4. Explain the RESTFul Web Service?

RESTful web services are built to work best on the Web. Representational State Transfer (REST) is an architectural style that specifies constraints, such as the uniform interface, that if applied to a web service induce desirable properties, such as performance, scalability, and modifiability, that enable services to work best on the Web. In the REST architectural style, data and functionality are considered resources and are accessed using Uniform Resource Identifiers (URIs), typically links on the Web. The resources are acted upon by using a set of simple, well-defined operations. The REST architectural style constrains an architecture to a client/server architecture and is designed to use a stateless communication protocol, typically HTTP. In the REST architecture style, clients and servers exchange representations of resources by using a standardized interface and protocol.

5. Explain what is a “Resource” in REST?

The fundamental concept in any RESTful API is the resource. A resource is an object with a type, associated data, relationships to other resources, and a set of methods that operate on it. It is similar to an object instance in an object-oriented programming language, with the important difference that only a few standard methods are defined for the resource (corresponding to the standard HTTP GET, POST, PUT and DELETE methods), while an object instance typically has many methods.

Resources can be grouped into collections. Each collection is homogeneous so that it contains only one type of resource, and unordered. Resources can also exist outside any collection. In this case, we refer to these resources as singleton resources. Collections are themselves resources as well.

Collections can exist globally, at the top level of an API, but can also be contained inside a single resource. In the latter case, we refer to these collections as sub-collections. Sub-collections are usually used to express some kind of “contained in” relationship.

6. Which protocol is used by RESTful web services?

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.

7. What is messaging in RESTful web services?

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.

8. State the core components of an HTTP Request?

The core components of an 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.

9. State the core components of an HTTP response?

The core components of an HTTP response are: -

* Request Code: This contains various codes which 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.

10. What do you understand about payload in RESTFul web service?

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.

11. Explain the caching mechanism?

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.

12. List the main differences between SOAP and REST?

The differences between SOAP and REST are: -

* SOAP is a protocol. REST is an architectural style. An API is designed to expose certain aspects of an application’s business logic on a server, and SOAP uses a service interface to do this while REST uses URIs.
* REST APIs access a resource for data (a URI); SOAP APIs perform an operation. REST is an architecture that’s more data-driven; SOAP is a standardized protocol for transferring structured information that’s more function-driven.
* REST permits many different data formats including plain text, HTML, XML, and JSON, which is a great fit for data and yields more browser compatibility; SOAP only uses XML.
* Security is handled differently. SOAP supports WS-Security, which is great at the transport level and a bit more comprehensive than SSL, and more ideal for integration with enterprise-level security tools. Both support SSL for end-to-end security, and REST can use the secure version of the HTTP protocol, HTTPS.
* SOAP requires more bandwidth; REST requires fewer resources (depending on the API). There’s a little more overhead with SOAP out of the gate, on account of the envelope-style of payload transport. Because REST is used primarily for web services, its being lightweight is an advantage in those scenarios.
* REST calls can be cached, SOAP-based calls cannot be cached. Data can be marked as cacheable, which means it can be reused by the browser later without having to initiate another request back to the server. This saves time and resources.
* An API is built to handle your app’s payload, and REST and SOAP do this differently. A payload is data sent over the internet, and when a payload is “heavy,” it requires more resources. REST tends to use HTTP and JSON, which lighten the payload; SOAP relies more on XML.
* SOAP is tightly coupled with the server; REST is coupled to a lesser degree. In programming, the more layers of abstraction between two pieces of technology, the less control you have over their interaction, but there’s also less complexity and it’s easier to make updates to one or the other without blowing up the whole relationship. The same goes for APIs and how closely they interact with a server. This is a key difference between SOAP and REST to consider. SOAP is very closely coupled with the server, having a strict communication contract with it that makes it more difficult to make changes or updates. A client interacting with a REST API needs no knowledge of the API, but a client interacting with a SOAP API needs knowledge about everything it will be using before it can even initiate an interaction.

13. Enlist advantages and disadvantages of ‘Statelessness’.

Advantages of 'Statelessness'

* 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 of 'Statelessness'

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