

Career Services Assignment 9 – API Flash Cards

Points possible: 50

Category	Criteria	% of Grade
Completeness	All requirements of the assignment are complete.	100

Instructions: Research common interview questions online revolving around REST, Spring, and building APIs and create 20 flash cards from the information you find. Study your flash cards regularly to better prepare for interviews. Fill out the table below with the information you put on each of your flash cards.

Front of Card	Back of Card
---------------	--------------

What does REST stand for?	REpresentational State Transfer
What is a resource?	A resource is how data is represented in the REST architecture. By exposing entities as the resource, it allows a client to read, write, modify, and create resources using HTTP methods, for example, GET , POST , PUT , DELETE, etc.
What are safe REST operations?	REST API uses HTTP methods to perform operations. Some of the HTTP operations, which don't modify the resource at the server, are known as safe operations, including GET and HEAD. On the other hand, PUT , POST, and DELETE are unsafe, because they modify the resource on the server.

<p>What are idempotent operations? Why is idempotency important?</p>	<p>There are some HTTP methods — like GET — that produce the same response no matter how many times you use them, sending multiple GET request to the same URI will result in the same response without any side-effect. Hence, this is known as idempotent.</p> <p>On the other hand, the POST is not idempotent, because if you send multiple POST requests, it will result in multiple resource creation on the server, but, again, PUT is idempotent, if you are using it to update the resource.</p> <p>Even multiple PUT requests can be used to update a resource on a server and will give the same end result. You can take an HTTP Fundamentals course by Pluralsight to learn more about idempotent methods of the HTTP protocol and HTTP in general.</p>
<p>Is REST scalable and/or interoperable ?</p>	<p>Yes, REST is scalable and interoperable. It doesn't mandate a specific choice of technology either at the client or server end. You can use Java, C++, Python, or JavaScript to create RESTful web services and consume them at the client end. I suggest you read a good book on REST API, like RESTful Web Services to learn more about REST.</p>

Which HTTP methods does REST use?	REST can use any HTTP methods, but the most popular ones are GET for retrieving a resource, POST for creating a resource, PUT for updating a resource , and DELETE for removing a resource from the server.
Is REST normally stateless?	<p>Yes, REST API should be stateless, because it is based on HTTP, which is also stateless. A request in REST API should contain all the details required to process it.</p> <p>It should not rely on previous or next requests or some data maintained at the server end, like sessions.</p> <p>The REST specification puts a constraint to make it stateless, and you should keep that in mind while designing your REST API.</p>

<p>What is the HTTP status return code for a successful DELETE statement?</p>	<p>There is no strict rule about what status code your REST API should return to after a successful DELETE. It can return 200 Ok or 204 No Content.</p> <p>In general, if the DELETE operation is successful, the response body is empty, return 204. If the DELETE request is successful and the response body is NOT empty, return 200.</p>
<p>What does CRUD mean?</p>	<p>CRUD is a short form of Create, Read, Update, and Delete. In REST API, the POST is used to create a resource, GET is used to read a resource, PUT is used to updated a resource, and DELETE is used to remove a resource from the server.</p>
<p>Is REST secure? What can you do to secure it?</p>	<p>This question is mostly asked by experienced Java programmers with about 2 to 5 years of experience with both REST and Spring. Security is a broad term; it could mean the security of a message, which is provided by encryption or access restriction that is provided using authentication and authorization.</p> <p>REST is normally not secure, but you can secure it by using Spring Security. At the very least, you can enable the HTTP basic authentication by using HTTP in your Spring Security configuration file. Similarly, you can expose your REST API using HTTPS, if the underlying server supports HTTPS.</p>

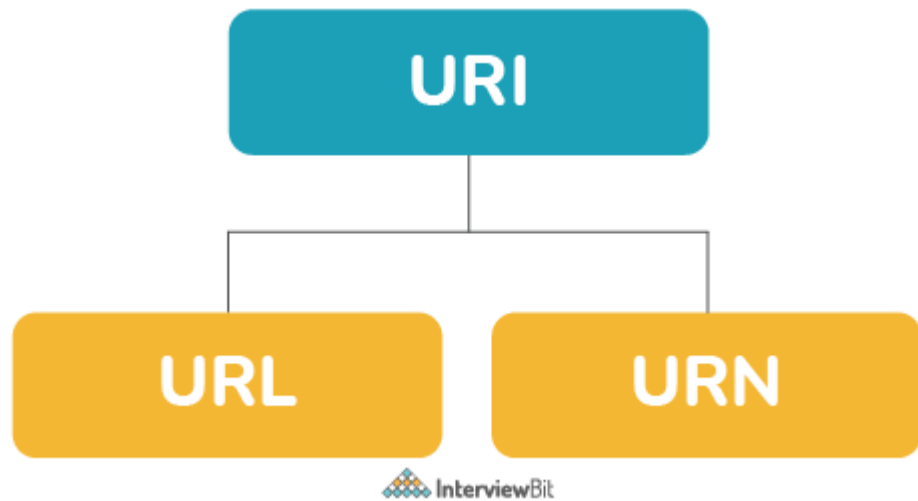
<p>Does REST work with transport layer security (TLS)?</p>	<p>Transport Layer Security (TLS) is used for secure communication between the client and the server. It is the successor of SSL (Secure Socket Layer). Since HTTPS can work with both SSL and TLS, REST can also work with TLS.</p> <p>Actually, in REST, it is up to the server to implement security protocols. The same RESTful web service can be accessed using HTTP and HTTPS if the server supports SSL.</p>
<p>What do you understand by RESTful Web Services?</p>	<p>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.</p>
<p>What is a REST Resource?</p>	<p>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.</p> <ul style="list-style-type: none"> • 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.

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:

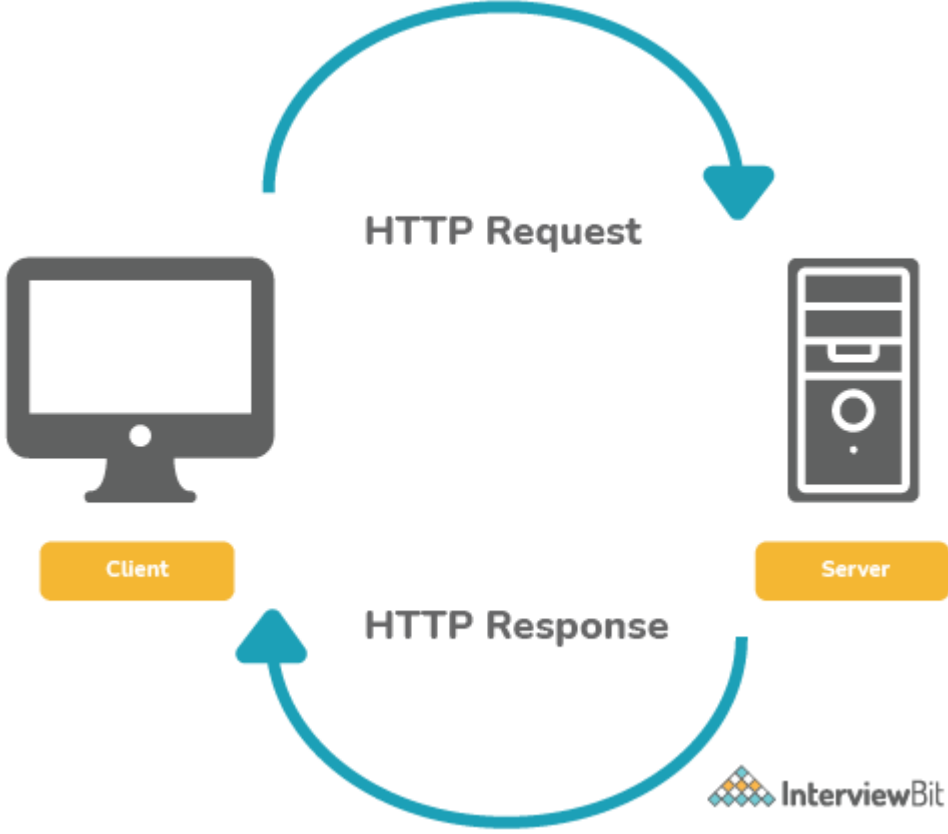
```
<protocol>://<service-name>/<ResourceType>/<ResourceID>
```

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:

	<ul style="list-style-type: none"> ▪ <code>http://abc.com/samplePage.html</code> ▪ <code>ftp://sampleServer.com/sampleFile.zip</code> ▪ <code>file:///home/interviewbit/sampleFile.txt</code> <ul style="list-style-type: none"> ○ 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.
What are the features of RESTful Web Services?	<p>Every RESTful web service has the following features:</p> <ul style="list-style-type: none"> • 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.
What is the concept of statelessness in REST?	<p>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.</p>

	 <p>The diagram illustrates the HTTP communication process. On the left is a computer icon labeled 'Client' in an orange box. On the right is a server rack icon labeled 'Server' in an orange box. A curved blue arrow points from the Client to the Server, labeled 'HTTP Request'. Another curved blue arrow points from the Server back to the Client, labeled 'HTTP Response'. The InterviewBit logo is in the bottom right corner.</p>
<p>What do you understand by JAX-RS?</p>	<p>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.</p>

<p>What are HTTP Status codes?</p>	<p>These are the standard codes that refer to the predefined status of the task at the server. Following are the status codes formats available:</p> <ul style="list-style-type: none">• 1xx - represents informational responses• 2xx - represents successful responses• 3xx - represents redirects• 4xx - represents client errors• 5xx - represents server errors <p>Most commonly used status codes are:</p> <ul style="list-style-type: none">• 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.
------------------------------------	---

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**.



Create



Read



Update



Delete

CRUD

	<p>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.</p>
<p>Can you tell the disadvantages of RESTful web services?</p>	<p>The disadvantages are:</p> <ul style="list-style-type: none"> • 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.