What is a Web Service?
A Web service is a web application that can communicate with other web based applications over network. Web service implementation allows two web applications developed in different languages to interact with each other using a standardized medium like XML, SOAP, HTTP etc

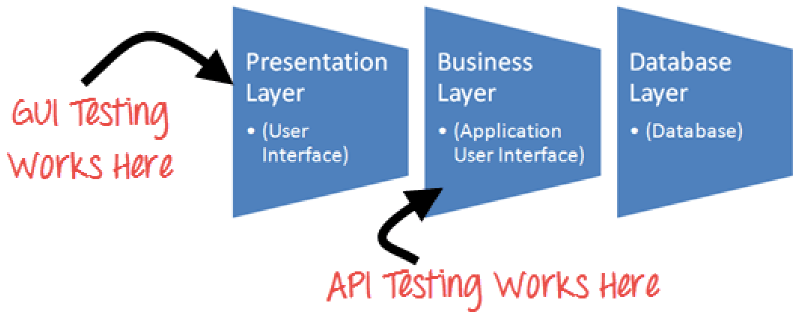
Features of web services-

- As web services are based on open standards like XML, HTTP so these are **operating system independent**
- Likewise web services are **programming language independent**, a java application can consume a PHP web service
- Web services can be **published over internet** to be consumed by other web applications
- The consumer of web service is **loosely coupled with the web service**, so the web service can update or change their underlying logic without affecting the consumer

What is an API?
API is an acronym for **A**pplication **P**rogramming **I**nterface.
It enables communication and data exchange between two separate software systems. A software system implementing an API contains functions/sub-routines which can be executed by another software

API Testing is entirely different from GUI Testing and mainly concentrates on the business logic layer of the software architecture. This **testing won't concentrate on the look and feel of an application.**
Instead of using standard user inputs(keyboard) and outputs, in API Testing, you use software to send calls to the API, get output, and note down the system's response.

API Testing requires an application to interact with API. In order to test an API, you will need to
Use Testing Tool to drive the API
Write your own code to test the API



API vs Web Service

- API and Web service serve as a **means of communication**.
- The only difference is that a Web service facilitates interaction between two machines over a network.
- An API acts as an interface between two different applications so that they can communicate with each other. An API is a method by which the third-party vendors can write programs that interface easily with other programs.
- A Web service is designed to have an interface that is depicted in a machine-processable format usually specified in Web Service Description Language (WSDL).
- Typically, “HTTP” is the most commonly used protocol for communication.
- Web service also uses SOAP, REST, and XML-RPC as a means of communication.
- API may use any means of communication to initiate interaction between applications. For example, the system calls are invoked using interrupts by the Linux kernel API.

“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.

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.

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.

WEB SERVICE	**API**
All web services are APIs.	All APIs are not web services.
It is not open source but can be used by any client that understands XML.	It is open source and it can be used by any client that understands JSON or XML.
It requires a SOAP protocol to receive and send data over the network, so it is not a light-weight architecture.	It is light-weight architectured and good for devices which have limited bandwidth, like mobile devices.
A Web service uses only three styles of use: SOAP, REST and XML-RPC for communication.	API may use any style of communication.
It only supports the HTTP protocol.	It supports the HTTP protocol: URL, Request/Response Headers, caching, versioning, content formats.

SOAP Web Services: SOAP (Simple Object Access Protocol) is an XML-based protocol for accessing web services. Its interface is described in a machine-processable format called WSDL (Web Service Definition Language) document. A web service is described by using a standard, formal XML notion that provides all necessary details like message format, transport protocols, and location to interact with the web service.

SOAP stands for Simple Object Access Protocol, it is a standardized protocol for message exchange between web applications. The message format supported by SOAP is XML. A web service that is based on SOAP protocol is called SOAP web service.

REST Web Services: REST (Representational State Transfer) is a style of software architecture. The data format is described by using JSON schema notation, and it requires the use of the HTTP transport protocol.

SOAP	REST
SOAP is a protocol.	REST is an architectural style.
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.
SOAP only permits XML.	REST permits many different data formats including plain text, HTML, XML, and JSON…
SOAP requires more bandwidth and more resources.	REST requires less bandwidth and less resources.
SOAP supports both SMTP and HTTP protocols.	REST requires the use of HTTP only.
SOAP is more reliable than REST.	REST is less secure than SOAP.
In most cases, SOAP is faster than REST.	REST is slower than SOAP.
SOAP defines its own security.	RESTful web services inherit security measures from the underlying transport.

The six REST architecture constraints are-

1.Client-Server - Client and server are separated by a uniform interface and are not concerned with each other's internal logic
2.Stateless - Each client request is independent and contains all the necessary information required to get executed. No client data is saved at the server.
3.Cacheable - Client should have the ability to cache the responses
4.Layered System - A layered system having multiple layers wherein each layer communicates with adjacent layer only
5.Uniform Interface - A uniform interface design requires each component within the service to share a single and uniform architecture
6.Code on Demand - This constraint is optional. It extends client side execution of code transfer of executable scripts like javascript from server.

RESTful APIs implements the following types of HTTP methods-

•GET - HTTP GET method is used to retrieve some information
•POST - HTTP POST method submits and creates new resources
•PUT - HTTP PUT is used to update an already existing resource
•DELETE - HTTP DELETE is used to delete a resource

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>.

Example
We have address **https://www.google.com/folder/page.html** where,
URI(Uniform Resource Identifier) => https://www.google.com/folder/page.html
URL(Uniform Resource Locator) => https://www.google.com/
URN(Uniform Resource Name) => /folder/page.html
URI => (URL + URN) or URL only or URN only

A Client and a Server establishes a connection using HTTP protocol. Once the connection is established, Client sends across the request to the Server in the form of XML or JSON which both entities (Client and Server) understand. After understanding the request Server responds with appropriate data by sending back a Response.

What is HTTP Request?
HTTP Request is a packet of Information that one computer sends to another computer to communicate something. To its core, *HTTP Request* is a packet of binary data sent by the Client to server. An *HTTP Request* contains following parts
Request Line
Headers, 0 or more Headers in the request
An optional Body of the Request

Resource URL: *http://restapi.demoqa.com/utilities/weatherfull/*
Parameter: *city/cityName=city/<NameFoTheCity>*
City Name is provided in the URL via a parameter query at the end of the URL. Let's say that we want to see the weather details of Hyderabad. In that case the URL will become *http://restapi.demoqa.com/utilities/weatherfull/city/hyderabad*

What is HTTP Response?
HTTP Response is the packet of information sent by *Server* to the *Client* in response to an earlier *Request* made by *Client*. *HTTP Response* contains the information requested by the Client.

Header - Content-Type / Status line / value = application/json; charset=utf-8.

***Body*- JSON body**

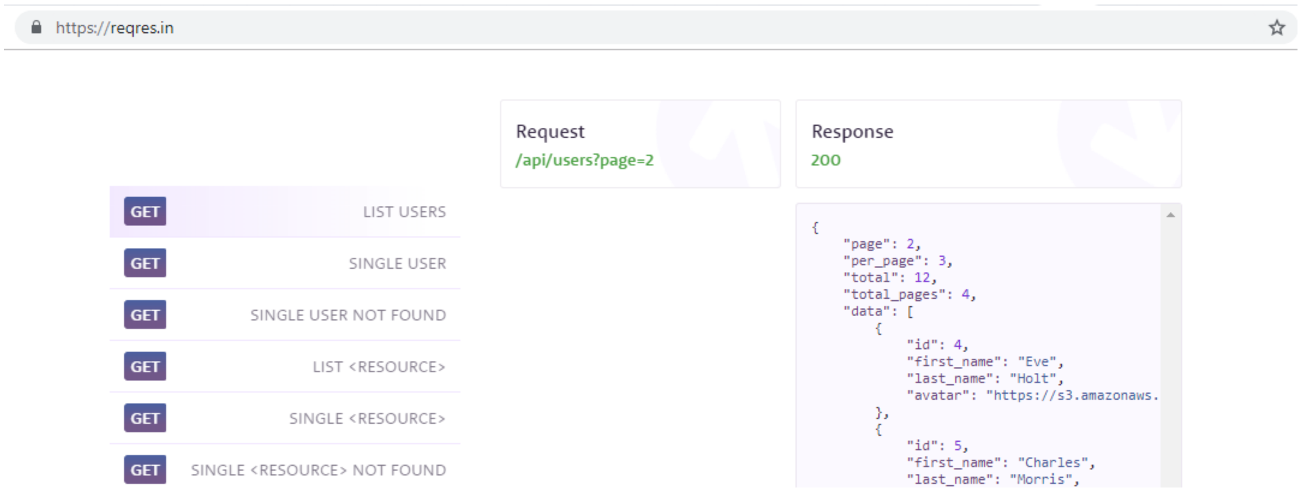
What do you mean by PKI?
It means Public-Key Infrastructure.

Differentiate between a SOA and a Web service?
SOA is a design and architecture to implement other services. SOA can be easily implemented using various protocols such as HTTP, HTTPS, JMS, SMTP, RMI, IIOP, RPC etc. While Web service, itself is an implemented technology. In fact one can implement SOA using the web service.

Can you name different kinds of web services?
There are two types of web services in total i.e. SOAP based web service and RESTful web service.

WSDL: It stands for web service description language (WSDL). It is used to describe web services. The description includes URL of web services, properties and methods supported by web services, data type it supports and protocol detail it supports

GET - HTTP GET method is used to retrieve some information
Status code - 200



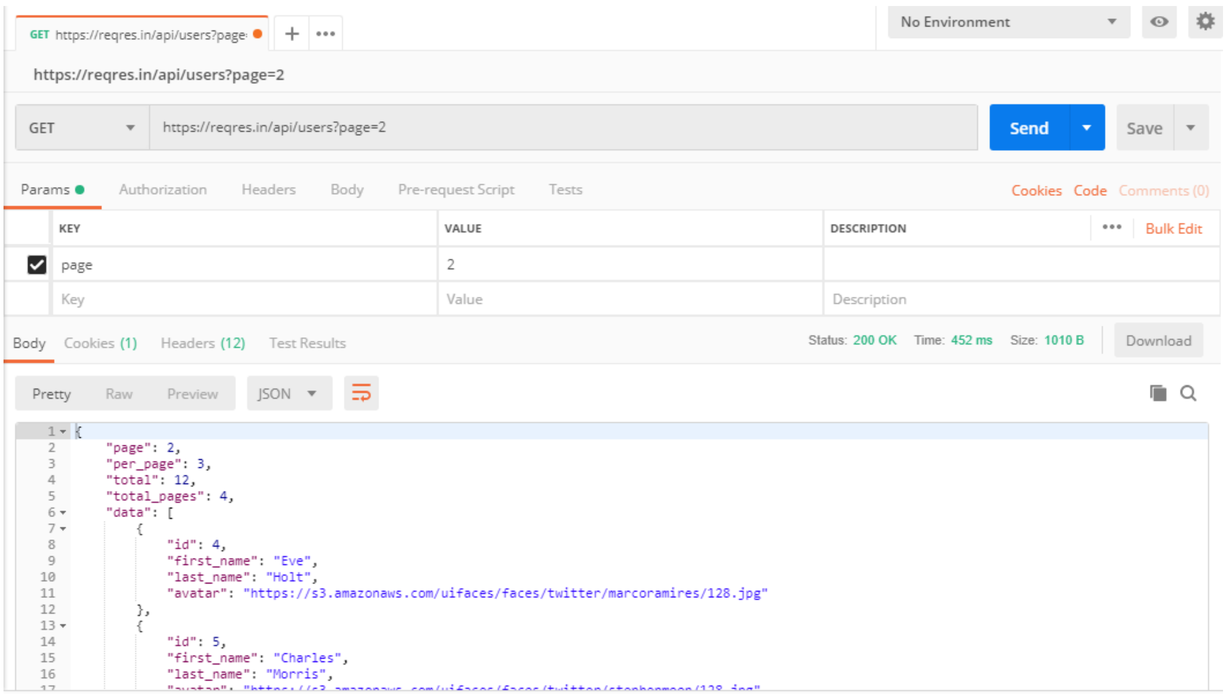
https://reqres.in

GET	LIST USERS
GET	SINGLE USER
GET	SINGLE USER NOT FOUND
GET	LIST <RESOURCE>
GET	SINGLE <RESOURCE>
GET	SINGLE <RESOURCE> NOT FOUND

Request
/api/users?page=2

Response
200

```
{
    "page": 2,
    "per_page": 3,
    "total": 12,
    "total_pages": 4,
    "data": [
        {
            "id": 4,
            "first_name": "Eve",
            "last_name": "Holt",
            "avatar": "https://s3.amazonaws.
        },
        {
            "id": 5,
            "first_name": "Charles",
            "last_name": "Morris",
```



GET https://reqres.in/api/users?page=2

GET https://reqres.in/api/users?page=2

KEY	VALUE	DESCRIPTION
page	2	
Key	Value	Description

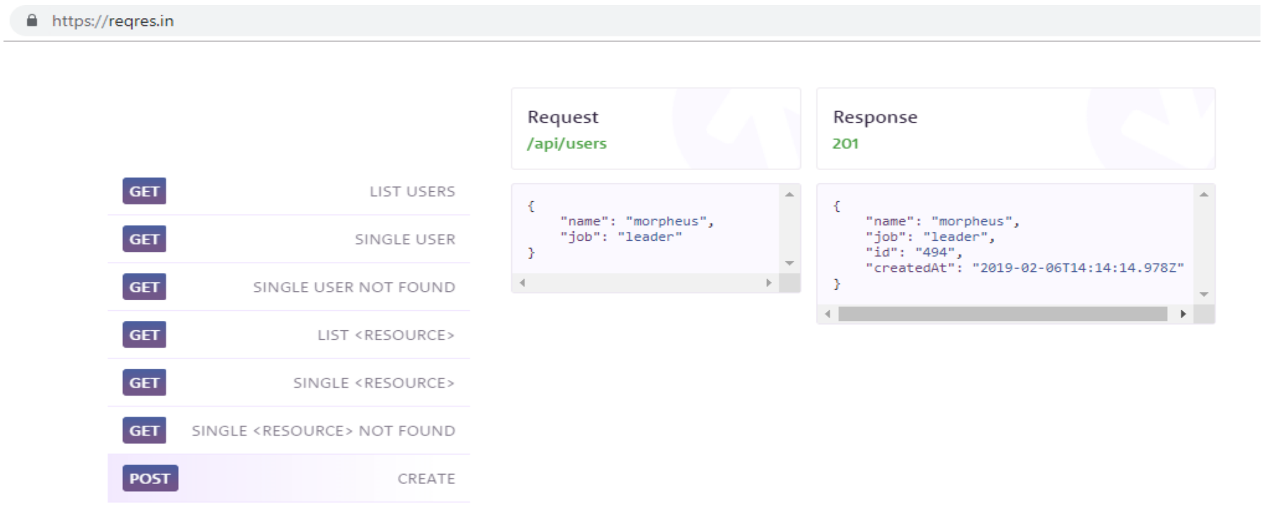
Status: 200 OK Time: 452 ms Size: 1010 B

```
{
    "page": 2,
    "per_page": 3,
    "total": 12,
    "total_pages": 4,
    "data": [
        {
            "id": 4,
            "first_name": "Eve",
            "last_name": "Holt",
            "avatar": "https://s3.amazonaws.com/uifaces/faces/twitter/marcoramires/128.jpg"
        },
        {
            "id": 5,
            "first_name": "Charles",
            "last_name": "Morris",
```

```java
@Test
public void getMethod() throws JSONException {
    Response response = get("https://reqres.in/api/users?page=2");
    ResponseBody body = response.getBody();
    int statusCode = response.getStatusCode();
    System.out.println(body.asString());
    System.out.println(statusCode);
}
```

```
200
{"page":2,"per_page":3,"total":12,"total_pages":4,"data":[{"id":4,"first_name":"Eve","last_name":"Holt","avatar":
```

•POST - HTTP POST method submits and creates new resources
•Staus Code - 201


https://reqres.in

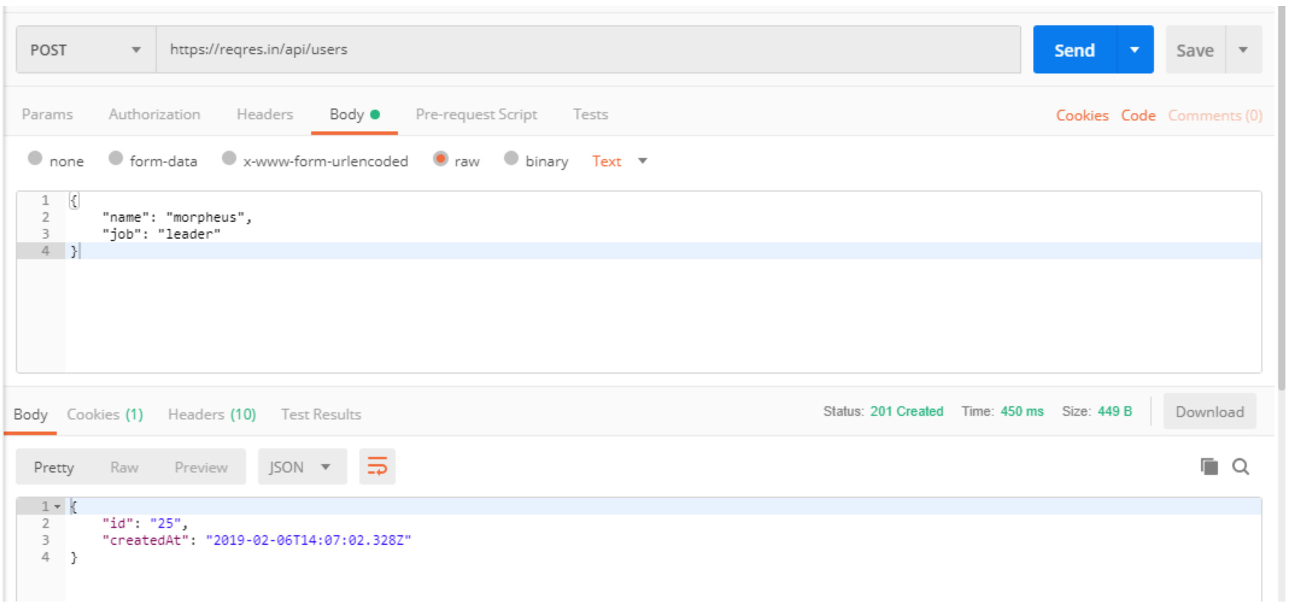
GET	LIST USERS
GET	SINGLE USER
GET	SINGLE USER NOT FOUND
GET	LIST <RESOURCE>
GET	SINGLE <RESOURCE>
GET	SINGLE <RESOURCE> NOT FOUND
POST	CREATE

Request
/api/users

```
{
    "name": "morpheus",
    "job": "leader"
}
```

Response
201

```
{
    "name": "morpheus",
    "job": "leader",
    "id": "494",
    "createdAt": "2019-02-06T14:14:14.978Z"
}
```


POST https://reqres.in/api/users

none form-data x-www-form-urlencoded [✓ raw] binary Text

```
{
    "name": "morpheus",
    "job": "leader"
}
```

Status: 201 Created Time: 450 ms Size: 449 B

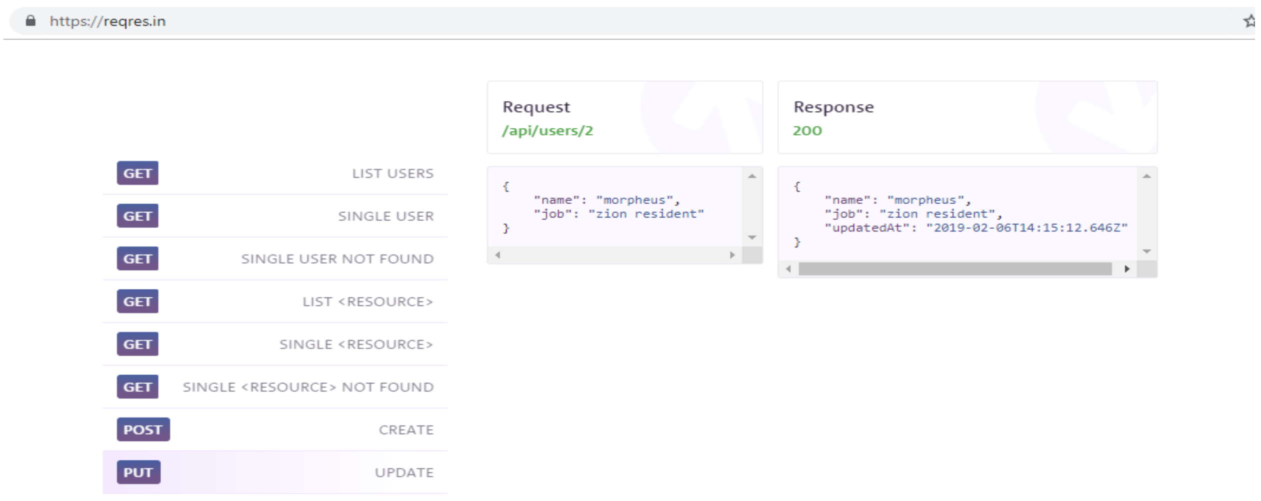
```
{
    "id": "25",
    "createdAt": "2019-02-06T14:07:02.328Z"
}
```

```
@Test
public void postMethod() {
    String post = "{\"name\": \"morpheus\",\"job\": \"leader\"}";
    Response response = given().body(post).when().contentType (ContentType.JSON).post("https://reqres.in/api/users");
    ResponseBody body = response.getBody();
    int statusCode = response.getStatusCode();
    System.out.println(body.asString());
    System.out.println(statusCode);

}
```

```
{"name":"morpheus","job":"leader","id":"873","createdAt":"2019-02-06T14:04:14.557Z"}
201
```

•PUT - HTTP PUT is used to update an already existing resource
•Status Code : 200

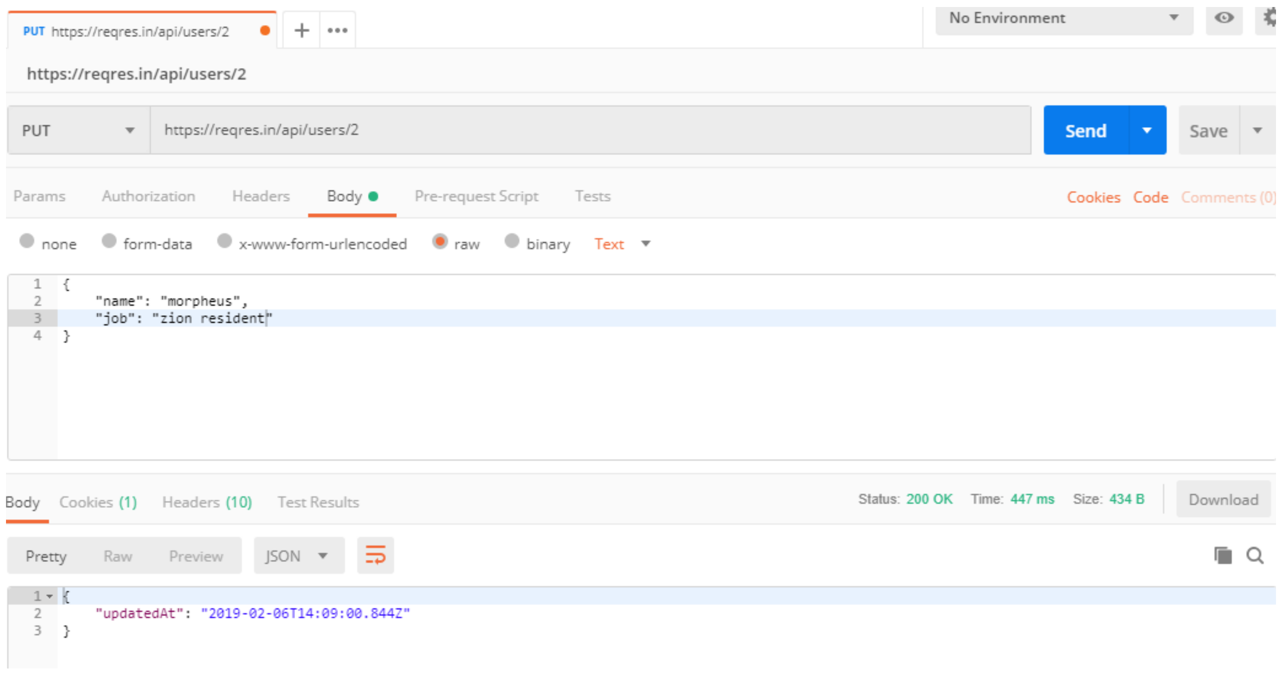


Request /api/users/2

```
{
    "name": "morpheus",
    "job": "zion resident"
}
```

Response 200

```
{
    "name": "morpheus",
    "job": "zion resident",
    "updatedAt": "2019-02-06T14:15:12.646Z"
}
```



PUT https://reqres.in/api/users/2

Body (raw, Text):

```
{
    "name": "morpheus",
    "job": "zion resident"
}
```

Status: 200 OK Time: 447 ms Size: 434 B

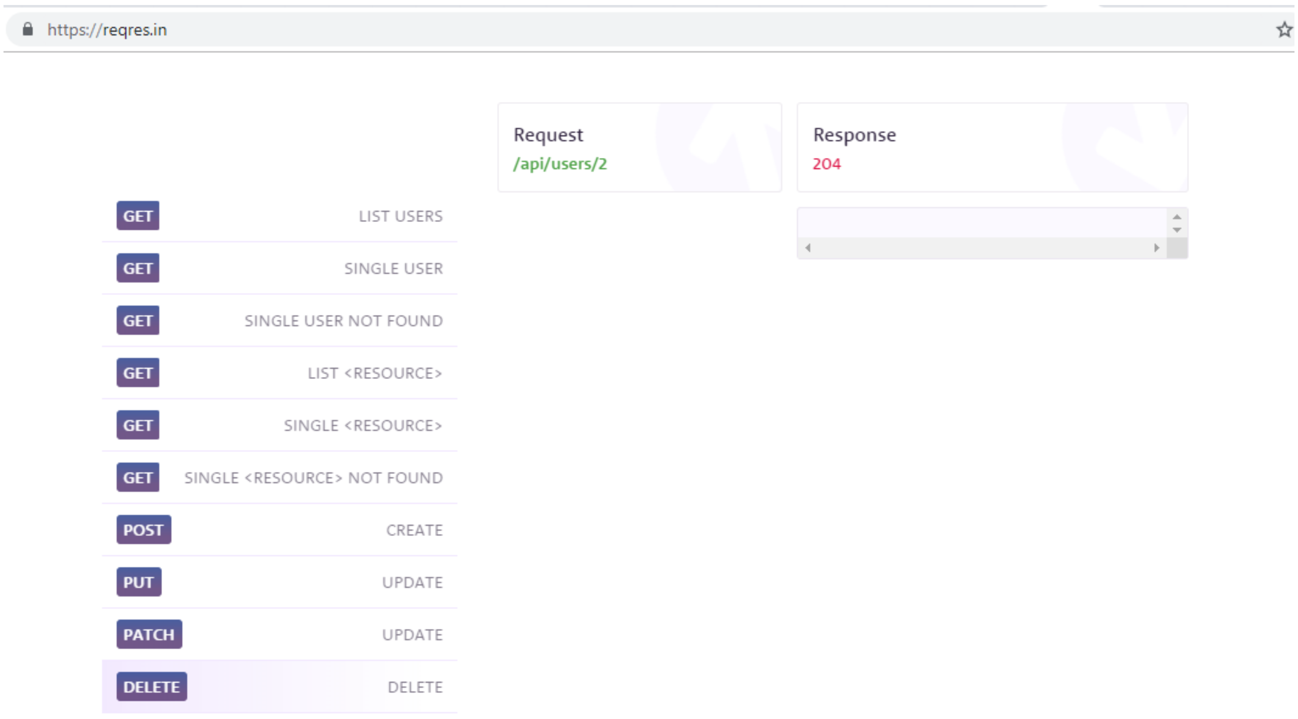
```
{
    "updatedAt": "2019-02-06T14:09:00.844Z"
}
```

```
@Test
public void putMethod() {

    String put = "{\"name\": \"morpheus\",\"job\": \"zion resident\"}";
    Response response = given().body(put).when().contentType(ContentType.JSON).put("https://reqres.in/api/users/2");
    ResponseBody body = response.getBody();
    int statusCode = response.getStatusCode();
    System.out.println(body.asString());
    System.out.println(statusCode);
}
```

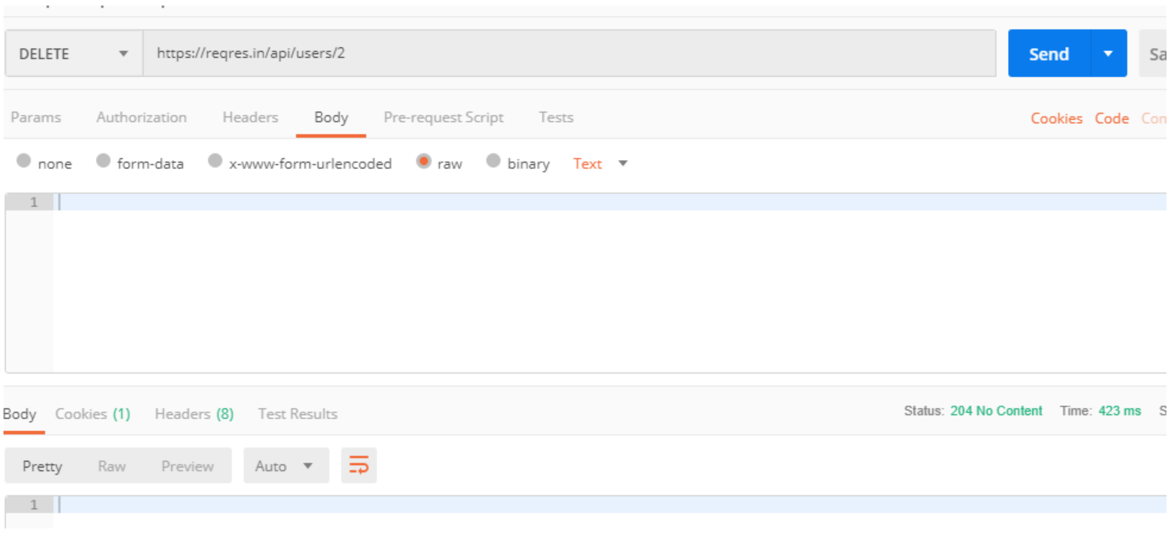
```
{"name":"morpheus","job":"zion resident","updatedAt":"2019-02-06T14:04:13.126Z"}
200
```

•DELETE - HTTP DELETE is used to delete a resource
•Status code : 204


https://reqres.in

Request
/api/users/2

Response
204


DELETE https://reqres.in/api/users/2

Status: 204 No Content Time: 423 ms

```java
@Test
public void deleteMethod() throws JSONException {
    Response response = given().when().contentType(ContentType.JSON).delete("https://reqres.in/api/users/2");
    System.out.println(response.getStatusCode());

}
```

```
204
```