**Q.4 study RESTful API**

One of the most popular types of [API](https://www.mulesoft.com/platform/api) is REST or, as they’re sometimes known, RESTful APIs. REST or RESTful APIs were designed to take advantage of existing protocols. While REST - or Representational State Transfer - can be used over nearly any protocol, when used for web APIs it typically takes advantage of HTTP. This means that developers have no need to install additional software or libraries when creating a REST API.

One of the key advantages of REST APIs is that they provide a great deal of flexibility. Data is not tied to resources or methods, so REST can handle multiple types of calls, return different data formats and even change structurally with the correct implementation of hypermedia. This flexibility allows developers to [build an API](https://www.mulesoft.com/lp/ebook/api/building-api-blueprint) that meets your needs while also meeting the needs of very diverse customers.

There are 6 key constraints to think about when considering whether a RESTful API is the right [type of API](https://www.mulesoft.com/resources/api/types-of-apis) for your needs:

* Client-Server: This constraint operates on the concept that the client and the server should be separate from each other and allowed to evolve individually.
* Stateless: REST APIs are stateless, meaning that calls can be made independently of one another, and each call contains all of the data necessary to complete itself successfully.
* Cache: Because a stateless API can increase request overhead by handling large loads of incoming and outbound calls, a REST API should be designed to encourage the storage of cacheable data.
* Uniform Interface: The key to the decoupling client from server is having a uniform interface that allows independent evolution of the application without having the application’s services, or models and actions, tightly coupled to the API layer itself.
* Layered System: REST APIs have different layers of their architecture working together to build a hierarchy that helps create a more scalable and modular application.
* Code on Demand: Code on Demand allows for code or applets to be transmitted via the API for use within the application.

Unlike SOAP, REST is not constrained to XML, but instead can return XML, JSON, YAML or any other format depending on what the client requests. And unlike RPC, users aren’t required to know procedure names or specific parameters in a specific order.

One of the disadvantages of RESTful APIs is that you can lose the ability to maintain state in REST, such as within sessions. It can also be more difficult for newer developers to use.

**Q 3 Think of services that should be offered by the software**

The general services provided by software development company are:

¬ Quality assurance

¬ Development of software and prototypes

¬ Test concepts development

¬ IT Projects coordination and management

¬ Re-design, revision and maintenance

¬ Vulnerability components analysis and identification

¬ Software, tuning and upgrading

Optimization level:

The SEO level is the highest level and it can be optimized into order. The selection of algorithms affects the efficiency which is available item of the set design.

Source code level:

In order to improve performance it is necessary to avoid quality coding. It can be done by avoiding common slowdowns.

Build level:

In this level between compile and source code level, build directives and flags may be used to tune optimal performances which are source code compiler and level.

Compile level:

It is an optimizing compiler which tends to make sure that executable program is optimized. The programs are optimized and the compiler can be able to predict.

Assembly level:

This is where language is used. Assembly designed purposely for certain hardware platform to produce efficient and also compact codes. The original program takes advantage of full repertoire machine instructions by s[oftware development company](http://www.htssolutions.org/)