# HOMEWORK- 3

## 1–) SOAP vs Restful ?

REST (Representational State Transfer) is a service structure that enables fast and easy communication between client and server. REST is a data transfer method used in software built on service-oriented architecture.

**SOAP** *(Simple Object Access Protocol)* in its most basic sense, is a protocol for transferring small amounts of information or messages over the Internet.

- While we can work with REST with JSON, XML and even TEXT, we should use XML with SOAP. REST can be more useful in this way. With JSON, you can perform operations with smaller data.
- If you want your application to run faster, it will be beneficial to use REST.

## 2-) Difference between acceptance test and functional test ?

**Functional Testing:** Application of test data derived from the specified functional requirements without regard to the final program structure. Also known as black-box testing.

**Acceptance Testing:** Formal testing conducted to determine whether or not a system satisfies its acceptance criteria—enables an end user to determine whether or not to accept the system.

## 3-) What is Mocking ?

Mocking is primarily used in unit testing. An object under test may have dependencies on other (complex) objects. To isolate the behavior of the object you want to replace the other objects by mocks that simulate the behavior of the real objects. This is useful if the real objects are impractical to incorporate into the unit test. In short, mocking is creating objects that simulate the behavior of real objects.

## 4-) What is a reasonable code coverage % for unit tests (and why) ?

Code coverage of 70-80% is a reasonable goal for system test of most projects with most coverage metrics. Use a higher goal for projects specifically organized for high testability or that have high failure costs. Minimum code coverage for unit testing can be 10-20% higher than for system testing.

## 5–) HTTP/POST vs HTTP/PUT ?

PUT puts a file or resource at a specific URI, and exactly at that URI. If there's already a file or resource at that URI, PUT replaces that file or resource. If there is no file or resource there, PUT creates one.

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

The fundamental difference between the POST and PUT requests is reflected in the different meaning of the Request-URI. The URI in a POST request identifies the resource that will handle the enclosed entity. That resource might be a data-accepting process, a gateway to some other protocol, or a separate entity that accepts annotations.

## 6-) What are the Safe and Unsafe methods of HTTP ?

**Safe method** means that if a method is logically read-only and the requests we make with this method do not cause any damage or change to the server. While all safe methods are also idempotent, not all idempotent methods are safe. For example, PUT and DELETE are both idempotent but unsafe.

**Likewise Unsafe method** means that if the requests we make with some method cause any changes on the server.We are adding information to or removing information from our database with unsafe methods.

## 7-) How does HTTP Basic Authentication work ?

In the context of a HTTP transaction, basic access authentication is a method for an HTTP user agent to provide a user name and password when making a request.

HTTP Basic authentication implementation is the simplest technique for enforcing access controls to web resources because it doesn't require cookies, session identifier and login pages. Rather, HTTP Basic authentication uses static, standard HTTP headers which means that no handshakes have to be done in anticipation.

When the user agent wants to send the server authentication credentials it may use the Authorization header. The Authorization header is constructed as follows:

1) Username and password are combined into a string "username:password"
2) The resulting string is then encoded using Base64 encoding
3) The authorization method and a space i.e. "Basic " is then put before the encoded string.

For example, if the user agent uses 'Aladdin' as the username and 'open sesame' as the password then the header is formed as follows:

```
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
```

## 8-) Define RestTemplate in Spring ?

RestTemplate is a synchronous client to perform HTTP requests. It uses a simple, template method API over underlying HTTP client libraries such as the JDK HttpURLConnection, Apache HttpComponents, and others.

Since Spring 5.0, a new client WebClient is available that can be use do create both synchronous and asynchronous requests. In the future releases, RestTemplate will be deprecated in favour of WebClient.

## 9–) What is idempotant and which HTTP methods are idempotant ?

An HTTP method is idempotent if an identical request can be made once or several times in a row with the same effect while leaving the server in the same state. In other words, an idempotent method should not have any side-effects (except for keeping statistics). Implemented correctly, the GET, HEAD, PUT, and DELETE methods are idempotent, but not the POST method. All safe methods are also idempotent. To be idempotent, only the actual back-end state of the server is considered, the status code

returned by each request may differ: the first call of a DELETE will likely return a 200, while successive ones will likely return a 404.

## 10–) What is DNS Spoofing ? How to prevent ?

The “spoofing” term in the attack means that the threat actor is using a malicious site that resembles the official website a user knows. Since DNS is a critical part of Internet communication, poisoning entries give an attacker the perfect phishing scenario to collect sensitive data. The threat actor can collect passwords, banking information, credit card numbers, contact information, and geographic data.

## 11–) What is content negotiation?

Content Negotiation is a content agreement between client and server. Its purpose is to be able to serve content in different document types with the same URI. In other words, the form of reference is determined by the users. Browsers forward their requests to the server as HTTP Accept Header information.

## 12 – What is statelessness in RESTful Web Services?

As per the rule followed by REST architecture, RESTful web services do not maintain a client’s state on a server. This restriction is known as **Statelessness.** Therefore the client shoulders the responsibility of passing its context directly to the server. The server, then stores the stores this context for processing client’s requests. For example, if a session is maintained by a server it will be identified by session identifier as it has been passed by the client.
RESTful web service is bound to follow this restriction. In this context you must be remembering that in the chapter **RESTful – web services- Method,** we have already studied about the special feature of web service methods which do not store a single information which is derived from a client.

## 13-) What is CSRF attack? How to prevent ?

Cross-site request forgery (also known as CSRF) is a web security vulnerability that allows an attacker to induce users to perform actions that they do not intend to perform. It allows an attacker to partly circumvent the same origin policy, which is designed to prevent different websites from interfering with each other.

The most robust way to defend against CSRF attacks is to include a CSRF token within relevant requests. The token should be:

- Unpredictable with high entropy, as for session tokens in general.
- Tied to the user's session.
- Strictly validated in every case before the relevant action is executed.

## 14-) What are the core components of the HTTP request and HTTP response ?

HTTP Response;

**Status/Response Code:** These are response codes issued by a server to a client's request. For example, 404 means Page Not Found, and 200 means Response is OK.

**HTTP Version**: describes HTTP version, for example-HTTP v1.1.

**Response Header:** Includes information for the HTTP response message. For example, Content-type, Content-length, date, status and server type.

**Response Body**: It contains the data that was requested by a client to server.

HTTP Request;

**HTTP methods** : Set of request methods to perform desired action for a given resource (GET, PUT, POST, DELETE)

**Uniform Resource Identifier (URI)**: Describes the resource

**HTTP Version**: HTTP v1.1

**Request Headers:** The content format supported , message format.

**Payload** : It is basically a Request Body which includes message content.