HW#3

1 - SOAP vs Restful?

SOAP (*Simple Object Access Protocol*) is a protocol. REST (*Representational State Transfer*) is an architectural pattern. SOAP was designed to ensure that programs built on different platforms and programming languages could be able to communicate and exchange data easily. REST is focused on accessing named resources through a single consistent interface and it is easier to use for the most part. REST seeks to fix the problems with SOAP and provide a simpler method of accessing web services.

SOAP was designed before REST. SOAP can't make use of REST since SOAP is a protocol and REST is an architectural pattern. But REST can make use of SOAP as the underlying protocol for web services.

2 - Difference between acceptance test and functional test?

A functional test verifies if the product works as we want and if it meets the business requirements. Functional tests are run by the software developers. Acceptance test verifies if the product actually solves the problem it was made to solve and if it handles test cases that cover the typical scenarios from real world. Acceptance tests are run by the users who requested the solution.

3 - What is Mocking?

Mocking is a unit test process. It is used for testing the code without touching the original objects, classes, functions, etc. In mocking, the dependencies are replaced by closely controlled replacements objects that simulate the behavior of the real ones. For example, when we are testing a business logic module which uses another module to make requests to an external API, we can mock the functions of the dependency for not to hit the API during our tests. We can run our tests, knowing what the mocked functions will return to our test subject. With mocking we can explicitly define the return value of methods without actually executing the steps of the method. So mocking is a useful testing process.

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

Code coverage is a metric which helps us to understand how much of our code is tested. It shows which areas of the code have been tested and which have not. Code coverage provides critical information to show teams where to focus their testing. It can provide insight and focus to help teams improve their testing.

The reasonable code coverage percentage depends on the expectations of people. It is not defined as a standard. Most people accept that the ideal reasonable code coverage percentage is 80. But it is better if the code coverage is closer to 100%. On the other hand, having "100% code-coverage" doesn't mean everything is tested completely. While it means every line of code is tested, it doesn't mean they are tested under every situation.

5 - HTTP/POST vs HTTP/PUT?

PUT and POST methods are used for data transmission between client to server. Generally, for to create data POST is a more ideal method. But for to update a data PUT is more ideal than post.

PUT method is idempotent but POST method is non-idempotent. PUT takes the data we gave and replaces it with the data at a specific url. If there is not a data at that url, PUT creates a new one with the data that we gave before. POST takes the data we gave and puts it under not a specific url but after putting the data it gets the exact url that stored the data we gave.

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

Safe methods of HTTP do not modify or delete resources. They lead to read-only operations. So using safe methods doesn’t create a danger for source data. On the other hand, unsafe methods of HTTP can make some changes or deletion on the source data. We should be careful when we use these methods.

Safe methods: GET, HEAD, OPTIONS, TRACE

Unsafe methods: POST, PUT, DELETE, CONNECT, PATCH

7 - How does HTTP Basic Authentication work?

HTTP basic authentication is a simple challenge and response mechanism with which a server can request authentication information (a user ID and password) from a client. The client sends HTTP requests with the Authorization header that contains the “*Basic*” word followed by a space and a base64-encoded string “*username:password*”. After that user ID and password information get verified if they are valid by comparing them against a database of authorized users.

8 - Define RestTemplate in Spring?

RestTemplate was designed for providing a simplified approach with default behaviors for performing complex tasks. RestTemplate is the central class within the Spring framework for executing synchronous HTTP requests on the client side. RestTemplate deals with a lot of action without bothering us. Generally it creates a URL object and opens the connection and configures the

HTTP request. Then it executes the HTTP request, interprets the response and converts it into a Java object. While dealing with these steps if it is necessary it deals with exception handling too. So we can say that RestTemplate makes our life easier.

But since Spring 5.0 RestTemplate is deprecated. It was replaced with WebClient class. Compared to RestTemplate, WebClient is more functional and fully reactive.

9 - What is idempotant and which HTTP methods are idempotant?

Idempotency means that every identical requests will have the same outcome without looking if a request is sent once or multiple times. So how many times a specific request is sent doesn't important for an idempotent method. If the method idempotent, outcome will be same for that request every time.

The idempotent method of HTTP are PUT, GET, HEAD, OPTIONS, TRACE and DELETE.

10 - What is DNS Spoofing? How to prevent?

DNS Spoofing is cyber-attack which is entering false information into a DNS cache. With that action DNS queries return an incorrect response which leads users to be directed to the wrong websites. It also known as "*DNS cache poisoning*". DNS Spoofing is a possible act because of DNS was originally designed without any verification structure. For to prevent DNS Spoofing it is suggested to use DNSSEC (Domain Name System Security Extensions). What DNSSEC does is verifying DNS data integrity and origin. Besides it is important to check for the padlock symbol next to the address bar when open a website.

11 - What is content negotiation?

REST resources can have multiple presentations. Content negotiation is a mechanism that makes it possible to serve different representation of a same resource (URI). There could be different clients expecting different representations. Content negotiation is the process of asking for a suitable presentation by a client. This means the client and server can negotiate about the representation type. With this process the ideal type of representation becomes determined.

12 - What is statelessness in RESTful Web Services?

Statelessness means that state of request is not maintained on server side. According to REST architecture, a RESTful Web Service should not keep a client state on the server. Every HTTP request send by a Statelessness Web Service happens in complete isolation. When the client makes an HTTP request, it includes all the information necessary for the server to fulfill that request. The server never relies on information from previous requests.

13 - What is CSRF attack? How to prevent?

Cross-Site Request Forgery (CSRF) is a cyber-attack that forces a user to execute unwanted actions on a web application in which they're authenticated. It tricks a user into sending unintended requests to modify data when only cookies are used for authentication. CSRF attack exploits a vulnerability in a Web application if it cannot differentiate between a request generated by an individual user and a request generated by a user without their consent. The most important thing to do for to prevent CSRF attack is to include a CSRF token within relevant requests. A CSRF secure application assigns a unique CSRF token for every user session. These tokens are inserted within hidden parameters of HTML forms related to critical server-side operations.

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

HTTP Request:

- *Start line:* Message sent by the client to initiate an action on the server. Contains three elements;
  - *HTTP method:* Methods such as GET, POST, PUT, etc.
  - *Request target:* Resource on the server
  - *HTTP version*: Defines the structure of the remaining message, indicates version
- *Request Headers:* Contains metadata, such as cache settings and client type, for the HTTP request message
- *Request Body:* Represents message content

HTTP Response:

- *Status line:* The start line of an HTTP response. Contains three elements:
  - *Protocol version:* Indicates the present version of HTTP
  - *Status code*: Indicating success or failure of the request
  - *Status text:* Textual description of the status code
- *Response Header:* Consists of metadata, like content length and server length, for the HTTP response message
- *Response Body:* Represents the response message content