HOMEWORK3
FATMA BETÜL UYAR

1 – SOAP vs Restful ?

- **SOAP** is a **protocol** that transports data. **REST** is an **architectural pattern**.
- SOAP only works with **XML** data formats, REST data formats can be **XML**, **JSON**,**HTML**,**TEXT**.
- If **data size** is important to your project, you can use Rest. Because JSON can transport small data type datas.
- REST faster than SOAP.
- In SOAP, It's **easier** to provide **security**.
- In REST we have to use **HTTP protocol**, whereas in SOAP we can use **other protocol**s such as TCP.
- We can find more documents or sources about SOAP.

2 - Difference between acceptance test and functional test ?

Acceptance testing is the type of testing which is used to check whether the software meets the customer requirements or not.
The purpose of **Functional tests** is to test each function of the software application, by providing appropriate input, verifying the output against the Functional requirements.
it is not concerned about the source code of the application.
For example; if we are testing that we go to the home page when we click the sign in button, we don't check if the user information is true or not.We are only testing this function.

3 - What is Mocking ?

Unit tests aim to it's at the unit level. I mean, the part we want to test has some dependencies like database connection. But we need to **test without** using these **dependencies**.
So we can use mocking. Mocking is creating objects that mimic the behavior of real objects. Through the mocking we don't have to wait for some processes that take a long time.We can focus only on the part we will test.
In Java, there are some libraries for mocking such as Mockito, JMock PowerMock.

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

Code coverage of 70-80% is a reasonable goal for system tests of most projects with most coverage metrics.
It isn't realistic to expect 100% code coverage through unit tests. The unit tests you create depend on business needs and the application or applications' complexity.

5 – HTTP/POST vs HTTP/PUT ?

- PUT method is idempotent. I mean, calling the same PUT requests multiple times will not change the result. But the result will change in the POST method.
- PUT is used to update results on the server.
- Server decide new resource path in POST, => POST/users
  Clients decide new resource path in PUT. => PUT/users/1
- POST ; I have data belonging to a resource I will create, take this data and create a new resource in this path for me.You decide the new resource path.
- PUT; I have data belonging to a resource I will create, data's path is … take this data and create a new resource in this path for me. If it already exists, update.

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

Methods that do **not change the server state** are considered safe. These methods are **GET**, **HEAD**, **OPTIONS**.

For example, requests that send for access to the city name from the database don't change the server state. So we can say that's a safe method.

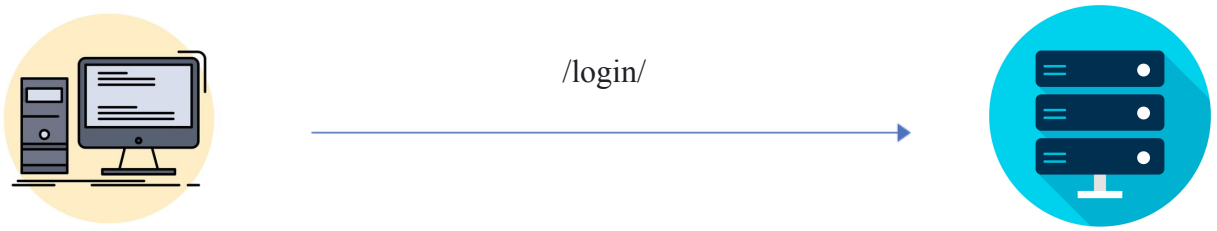
Idempotent means that multiple requests will have the same result.All safe methods are idempotent, but all idempotent methods are not safe.

7 - How does HTTP Basic Authentication work ?

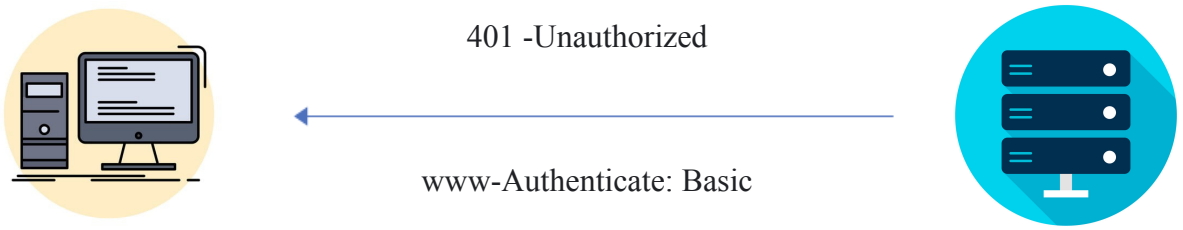
HTTP basic authentication is a simple challenge and response mechanism with which a server can request authentication information (a **user ID and password**) from a client.

Authentication steps are;

1-) Client sends a request to a page that uses basic authentication.



2-) Server sends HTTP 401 and www-Authenticate that it has Basic value with HTTP head.



Client Server

3-)Client's browser creates a field when it encounters this head.

Kullanıcı Adı: _____

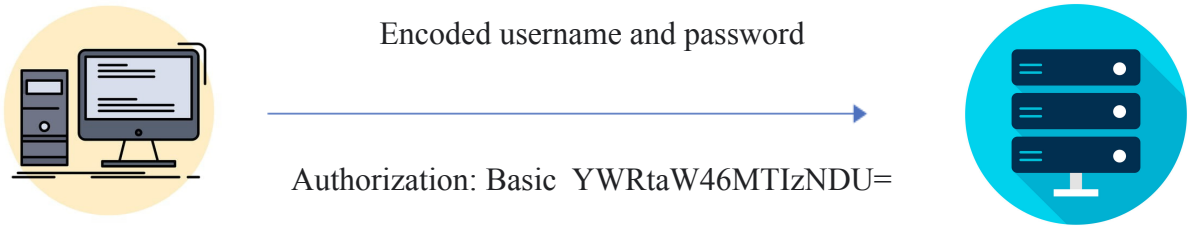
Şifre : _____

4-)Client fills the fields and sends them. Browser code uses base64 encoding.

Kodlama: Base64Encode('admin:12345');

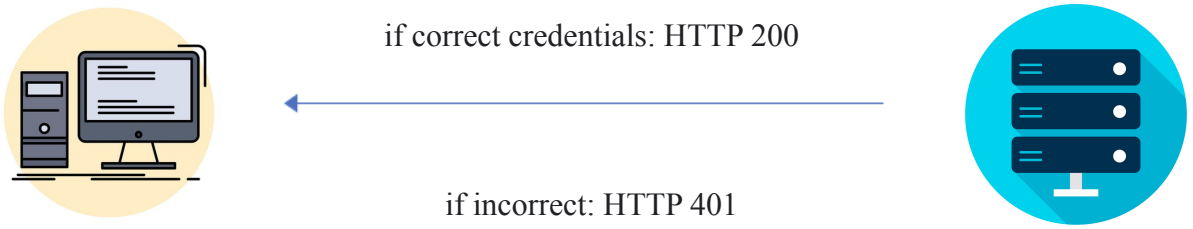
Çıktı : YWRtaW46MTIzNDU=

5-)Client browser sends it by adding Base64 value in Authorization head.



Client Server

6-)Server compares the stored value with the incoming value. If the two values are equal; It sends HTTP 200, but aren't equal sends HTTP 401 and back to step 2.



Client Server

8 - Define RestTemplate in Spring ?

RestTemplate is **a default class** that **manages** the **synchronous HTTP requests** on the client side.If we don't use the RestTemplate, for every request;

- configuring of http request
- exception handling
- creating a url object and connection

we have to do these. But we don't deal with them,when we use RestTemplate.

spring-boot-starter-web is a starter for building web applications. This dependency **contains** the class **RestTemplate**.

9 – What is idempotent and which HTTP methods are idempotent ?

Idempotent means that **multiple requests** will have the **same state**. **GET**,**HEAD**, **OPTIONS**, **DELETE** , **PUT** , **TRACE** are idempotent.

For example, we have some records in database like these;

‘ **id**: 1 , **name**: ‘Betul’, **surname**: ‘UYAR’,

‘ **id**: 2 , **name**: ‘Eslem’, **surname**: ‘UYAR’,

When we send **GET** request **/users/1 ,** => ‘ id: 1 , name: ‘Betul’, surname: ‘UYAR’,

Again we send **GET** request **/users/1 ,** => ‘ id: 1 , name: ‘Betul’, surname: ‘UYAR’,

Multiple requests will not change the state. So GET is idempotent.

But we send **POST** request **/users body{** name: ‘Fatma’, surname: ‘UYAR’**}** => ‘ id: 3 , name: ‘Fatma’, surname: ‘UYAR’, record that id=3 will return

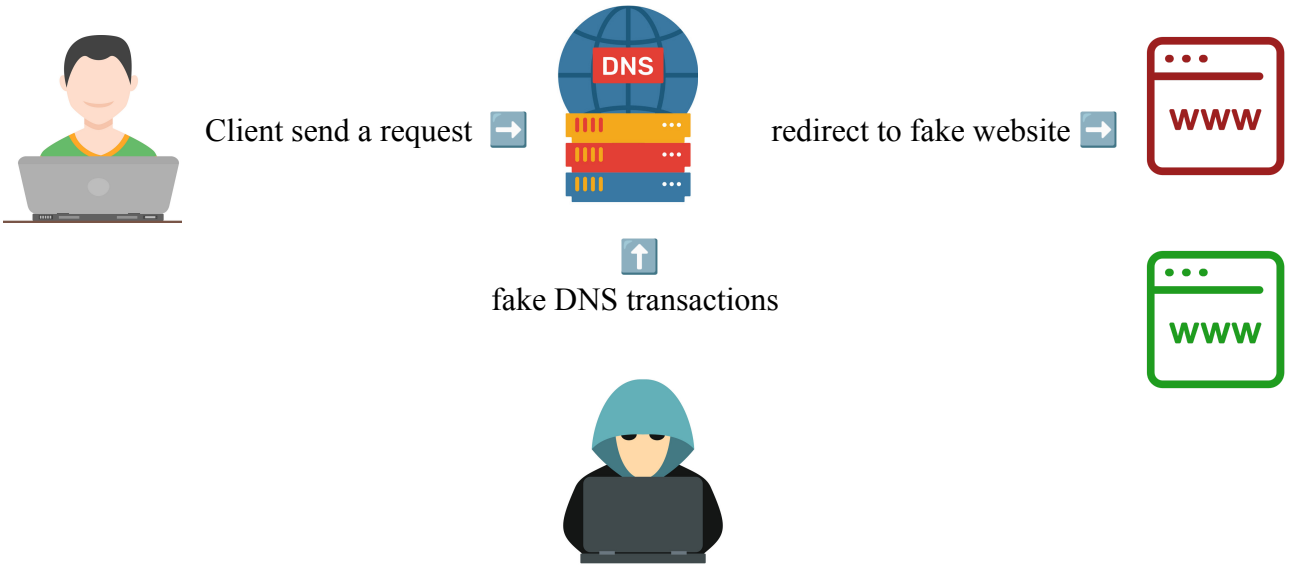
Again we send request **/users body{** name: Fatma, surname: ‘UYAR’**}** => ‘ id: 4 , name: ‘Fatma’, surname: ‘UYAR’, record that id=4 will return

Multiple requests will change the state. So POST is not idempotent.

10 – What is DNS Spoofing ? How to prevent ?

DNS converts IP addresses to human readable domain names.DNS cache is a temporary database that contains all records that send requests to other domain names. Every computer has a DNS cache.

Attackers who gain access to the DNS cache **replace real addresses** with **fake addresses**. So, users are redirected to the fake website.It is so difficult to understand that it is the fake website.Because fake websites look the same as the real website.



DNS servers **that use UDP** instead TCP and do **not authorize DNS information** causes DNS Spoofing. A hacker can send a message by UDP and imitate the real response.

To avoid;

-Tools built against DNS spoofing

- End to end encryption

- Don’t click unknown network
- Scan the computer for malware
- Use VPN

11 – What is content negotiation ?

Content negotiation is a deal that is made between client and server.It aims to serve different document type content with the same URI.
If a browser can display PNG instead of JPEG, It transmits to the server as **HTTP accepts Head** information.
For example;
Accept:image/PNG => It says that the user prefers PNG.
Accept: text/html, text/xml, image/jpeg, */*
***/* =>** different media types accept

12 – What is statelessness in RESTful Web Services ?

As per the REST architecture, a RESTful Web Service should not keep a client state on the server.It is the responsibility of the client to pass its context to the server and then the server can store this context to process the client's further request.

- Web services can treat each method request independently.
- Web services need not maintain the client's previous interactions. It simplifies the application design.
- As HTTP is itself a statelessness protocol, RESTful Web Services work seamlessly with the HTTP protocols.

13 - What is CSRF attack? How to prevent ?

Cross-site request forgery attacks are used to send malicious requests from an **authenticated user** to a web application.In a successful CSRF attack, the attacker causes the victim user to carry out an action unintentionally. For example, this might be to change the email address on their account to make a funds transfer. The attacker can’t see the responses to the forged requests, so CSRF attacks focus on state changes.
The most robust way to defend against CSRF attacks is to include a CSRF **token** within relevant requests.

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

HTTP Response;

1. **Status/Response Code** : For example, **404** means Page Not Found
2. **HTTP Version** : For example-HTTP **v1.1**.
3. **Response Header :** For example, Content-type
4. **Response Body** : It contains the data.

HTTP Request;

- **Method/Verb** : GET, PUT..
- **URI :** This part is used for uniquely identifying the resources on the server.
- **HTTP Version :** HTTP v1.1.
- **Request Header** − The content format supported, message format..
- **Request Body** − The actual message content to be sent to the server.