# CORS

* It stands for Cross Origin Resource Sharing.
* It is an HTTP-Header based mechanism that allows a server to indicate any other origins.
* It basically checks if the HTTP request header comes from a trusted source.
  + That depends on you as a developer who you trust to use your rest api
  + You could be a paid service and only allow your clients to use your api
  + Of course, for security reasons also

# Same Origin Policy

* Browser security prevents a web page from making request from a different domain than the one that served the web page.

## What is an Origin?

* It is the url you use to navigate in a webpage.

## What counts as a same Origin?

* It needs to have the same protocol, the same host, and (optionally) the same port
  + Ex: <https://google.com> - <https://google.com/search>?... Counts as the same origin

https://google.com – <http://google.com>.. Is not the same origin

* So the main idea is you don’t want a completely different website running some sort of Javascript in their website to check your cookies and obtain crucial information (like your login creds) and store it in their database without you knowing. The cookies will only be accessed if it came from the same origin the cookies were obtained.

# CSRF

* It stands for Cross-Site Request Forgery, and it is a web security vulnerability that bypasses Same Origin Policy.
* It basically uses you as the user to do that request without you really knowing.
* So, let’s say you are in a bad banking website that gives a basic form that asks the user the account number and some money to send to that account and a submit button to send that HTTP request. You left this bank website all authenticated and went to a different website that uses a comment functionality that has a submit button. You harmless posted a comment for a blog, video, photo, etc. but in their backend is an invisible form that emulates the same form as your bank website. It has prefilled values and it will send that http request and make seem like you, the user, wants to process this request.
* Tokens are the way to combat this security vulnerability.

# Difference between Rest and Soap

## Rest

* It only does HTTP and HTTPS only
* Any format can be used to send information like JSON
* Very easy to setup and get it running because HTTP is most used
* You can choose to not deal with XML at all

## SOAP

* Can use any protocols.
* Just uses XML to send information or WSDL.
* Well documented.