**WHAT IS CORS?**

A request for a resource (like an image or a font) outside of the origin is known as a *cross-origin* request. CORS (cross-origin resource sharing) manages cross-origin requests.

Cross-origin requests, however, mean that servers must implement ways to handle requests from origins outside of their own. CORS allows servers to specify who (i.e., which origins) can access the assets on the server, among many other things.

**Example**:

You can think of these interactions as a building with a security entrance. For example, If you needed a particular tool, however, and you ordered it from an outside source like an online marketplace (i.e., cross-origin), the security at the entrance may request that the delivery person provide identification when your tool arrives.

For the above example Same Origin Works Like This:

If you need to borrow a ladder, you could ask a neighbour in the building who has one. The building’s security would likely not have a problem with this request (i.e., same-origin).

### WHY IS CORS NECESSARY?

The CORS standard is needed because it allows servers to specify not only who can access the assets, **but also how they can be accessed**.

Cross-origin requests are made using the standard HTTP request methods. Most servers will allow GET requests, meaning they will allow resources from external origins (say, a web page) to read their assets. HTTP requests methods like PATCH, PUT, or DELETE, however, may be denied to prevent malicious behavior. For many servers, this is intentional. For example, it is likely that server A does not want servers B, C, or D to edit or delete its assets.

With CORS, a server can specify who can access its assets and which HTTP request methods are allowed from external resources.

### HOW DOES CORS MANAGE REQUESTS FROM EXTERNAL RESOURCES?

An HTTP header is a piece of information associated with a request or a response. Headers are passed back and forth between your web browser (also referred to as a client) and a server when the web page you are on wants to use resources hosted on a different server. Headers are used to describe requests and responses. The CORS standard manages cross-origin requests by adding new HTTP headers to the standard list of headers. The following are the new HTTP headers added by the CORS standard:

* Access-Control-Allow-Origin
* Access-Control-Allow-Credentials
* Access-Control-Allow-Headers
* Access-Control-Allow-Methods
* Access-Control-Expose-Headers
* Access-Control-Max-Age
* Access-Control-Request-Headers
* Access-Control-Request-Method
* Origin

These are all important, but let’s focus on the following header:

* Access-Control-Allow-Origin

The Access-Control-Allow-Origin header allows servers to specify how their resources are shared with external domains. When a GET request is made to access a resource on Server A, Server A will respond with a value for the Access-Control-Allow-Origin header. Many times, this value will be \*, meaning that Server A will share the requested resources with any domain on the Internet. Other times, the value of this header may be set to a particular domain (or list of domains), meaning that Server A will share its resources with that specific domain (or list of domains). The Access-Control-Allow-Origin header is critical to resource security.

### PRE-FLIGHT REQUESTS

As mentioned before, most servers will allow GET requests but may block requests to modify resources on the server. Servers don’t just blindly block such requests though; they have a process in place that first checks and then communicates to the client (your web browser) which requests are allowed.

When a request is made using any of the following HTTP request methods, a standard preflight request will be made before the original request.

**WHY IS CORS NECESSARY?**

The CORS standard is needed because it allows servers to specify not only who can access the assets, but also *how* they can be accessed.

Cross-origin requests are made using the standard HTTP request methods. Most servers will allow GET requests, meaning they will allow resources from external origins (say, a web page) to read their assets. HTTP requests methods like PATCH, PUT, or DELETE, however, may be denied to prevent malicious behavior. For many servers, this is intentional. For example, it is likely that server A does not want servers B, C, or D to edit or delete its assets.

With CORS, a server can specify who can access its assets and which HTTP request methods are allowed from external resources.

**HOW DOES CORS MANAGE REQUESTS FROM EXTERNAL RESOURCES?**

An HTTP header is a piece of information associated with a request or a response. Headers are passed back and forth between your web browser (also referred to as a client) and a server when the web page you are on wants to use resources hosted on a different server. Headers are used to describe requests and responses. The CORS standard manages cross-origin requests by adding new HTTP headers to the standard list of headers. The following are the new HTTP headers added by the CORS standard:

* Access-Control-Allow-Origin
* Access-Control-Allow-Credentials
* Access-Control-Allow-Headers
* Access-Control-Allow-Methods
* Access-Control-Expose-Headers
* Access-Control-Max-Age
* Access-Control-Request-Headers
* Access-Control-Request-Method
* Origin

These are all important, but let’s focus on the following header:

* Access-Control-Allow-Origin

The Access-Control-Allow-Origin header allows servers to specify how their resources are shared with external domains. When a GET request is made to access a resource on Server A, Server A will respond with a value for the Access-Control-Allow-Origin header. Many times, this value will be \*, meaning that Server A will share the requested resources with *any* domain on the Internet. Other times, the value of this header may be set to a particular domain (or list of domains), meaning that Server A will share its resources with that specific domain (or list of domains). The Access-Control-Allow-Origin header is critical to resource security.

**PRE-FLIGHT REQUESTS**

As mentioned before, most servers will allow GET requests but may block requests to modify resources on the server. Servers don’t just blindly block such requests though; they have a process in place that first checks and then communicates to the client (your web browser) which requests are allowed.

When a request is made using any of the following HTTP request methods, a standard *preflight* request will be made before the original request.

# Some Info:

* CORS is meant to prevent the client browser from downloading content that should not be the part of the web page.
* CORS only works for browsers and not requests using other clients.

# Same Origin:

* Two URL’S have origins have same origins if they have identical Schemas (http/https), Host (sample.com, sample.net, unsample.com), port (5000, 500).
* Unless requests are from same origin the requests will be rejected.

# CORS:

* CORS allow us to access the all origins.
* We can selectively allow Cross Origin Requests by rejecting others.