CORS

Cross-Origin Resource Sharing

Kenneth Fossen

bouvet

WHAT IS THIS?

```
<img src="http://bank.com/transfer.do?acct=MARIA&amount=100000" width="0" height="0" border="0">
```

IS THIS ALSO XSS?

```
<script>
function put() {
    var x = new XMLHttpRequest();
    x.open("PUT","http://bank.com/transfer.do",true);
    x.setRequestHeader("Content-Type", "application/json");
    x.send(JSON.stringify({"acct":"BOB", "amount":100}));
}
</script>

<body onload="put()">
```

https://owasp.org/www-community/attacks/csrf

COMMON FACTORS FOR THESE ATTACKS

- They both use Javascript in the browser
- These attacks are used to
  - Collect user cookies/secrets
  - Act on the user's behalf (without them knowing)
- CSRF is also known as the Confused Deputy

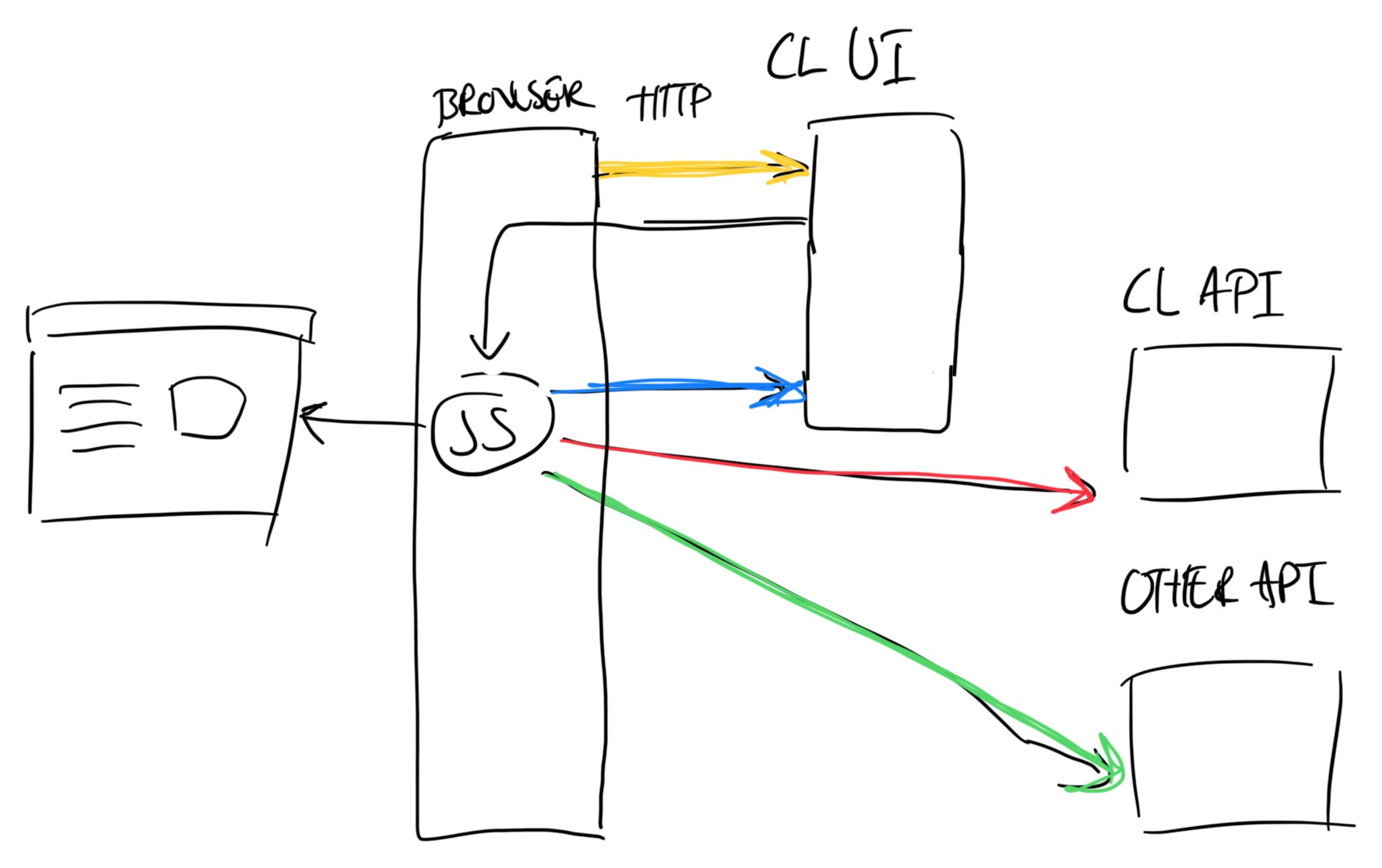
Confused deputy problem. (2022, August 5). In Wikipedia. https://en.wikipedia.org/wiki/Confused_deputy_problem

WELCOME CORS

- The modern browser now:
  - Restricts access to Cookies for JS
  - Content Security Policy to enable/disable APIs Javascript has access to
  - Uses CORS to restrict where we can download resources
- CORS helps unless you do stuff like this
  - `Access-Control-Allow-Origin: *`

TODAY'S PROBLEM?

Where is CORS in this?



AGENDA

- Repetition: Origin
- CORS Deep Dive
- Azure App Service CORS
- ASP.NET WebAPI CORS
- Backend-For-Frontend (BFF)

REPETITION: ORIGIN

What is Origin in HTTP world?

It is a tuple consisting of `<protocol>://<host>:<port>`

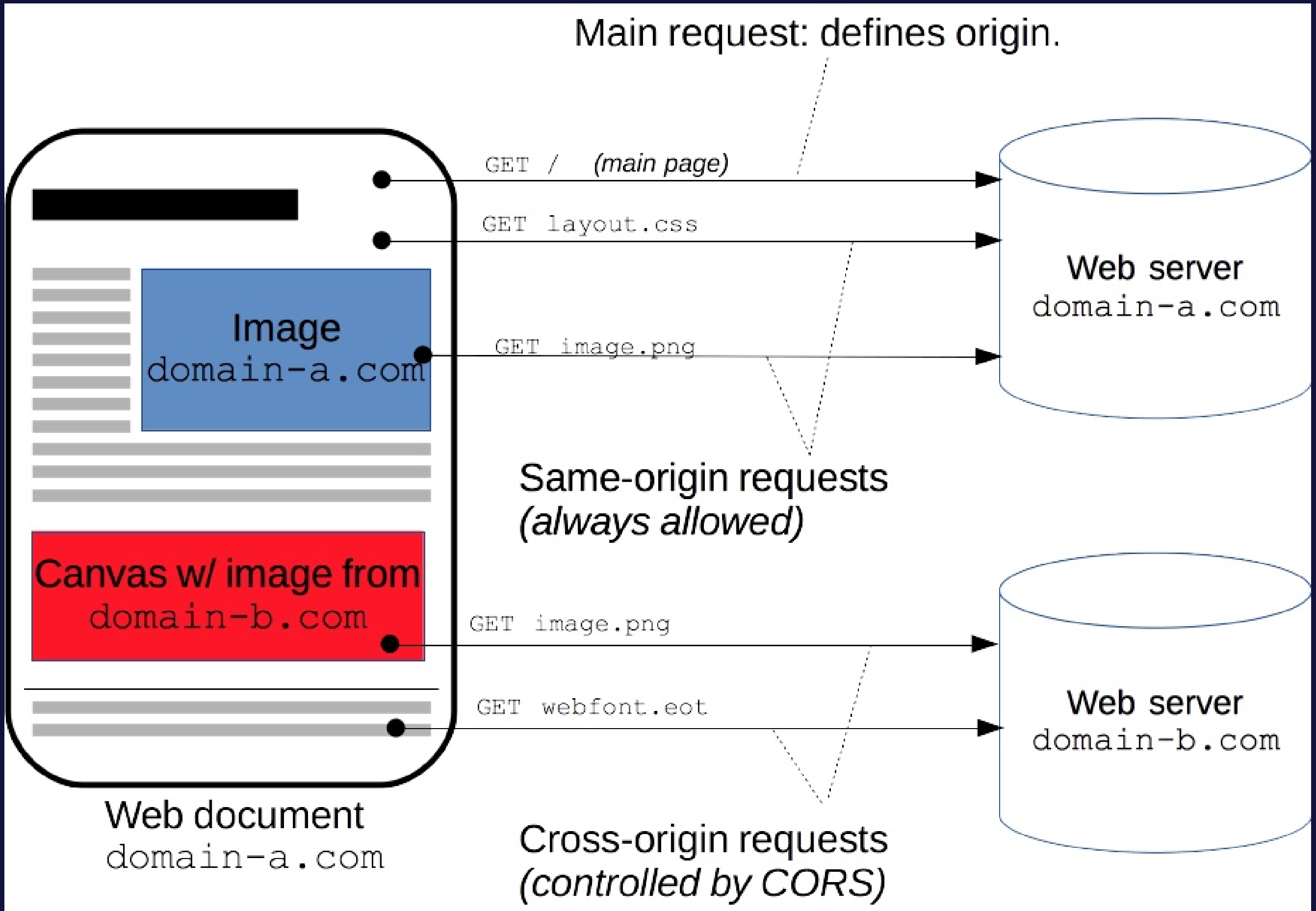
```
https://www.kefo.no/
http://www.kefo.no:443/
https://kefo.no/boisys/blog/today_i_hiked_photos
http://blog.kefo.no/
http://blog.kefo.no:81
```

CORS DEEP DIVE

*"An HTTP-Header mechanism that allows a **server** to indicate any origins other than its **own** from which a browser should be permitted to be loading resources."*

OVERVIEW

CORS



A SIMPLE REQUEST IS

- A `GET`, `HEAD`, or `POST` request

That only allows the following headers:

- `Accept`, `Accept-Language`, `Content-Language`, `Range`, and `Content-Type`

`Content-Type` can only hold these values:

- `application/x-www-form-urlencoded`
- `multipart/form-data`
- `text/plain`

https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS

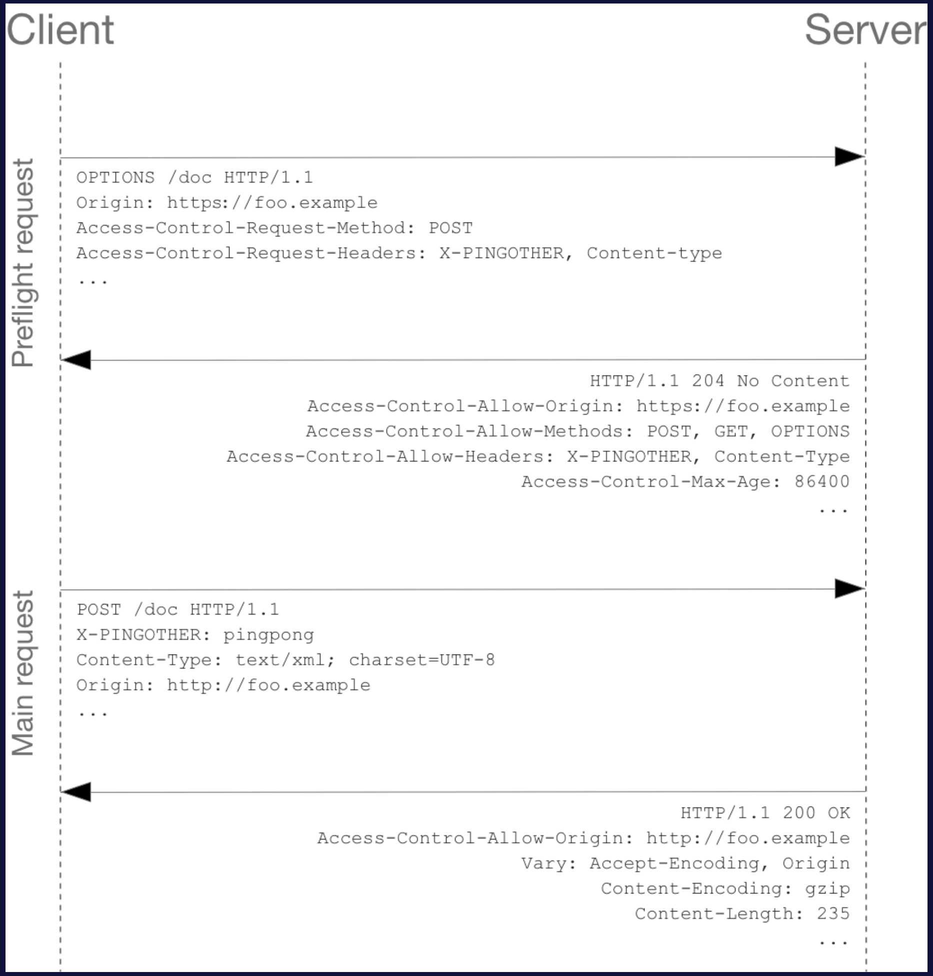
THAT WAS EASY

Yeah BUT, Safari, Firefox, and Chrome do implement things differently! 🤯 🤦🏻‍♂️

- Firefox does not implement the `Range`.
- Safari/WebKit is using stricter versions of `Accept`, `Accept-Language`, and `Content-Language`.

CORS: PREFLIGHT REQUEST

- The browser sends an OPTIONS request
- If Response is:
  - HTTP 204 or 200 - Everything OK
  - HTTP 405 - Not allowed BAD



```
OPTIONS /doc HTTP/1.1
Origin: https://foo.example
Access-Control-Request-Method: POST
Access-Control-Request-Headers: X-PINGOTHER, Content-type
...
```

```
HTTP/1.1 204 No Content
Access-Control-Allow-Origin: https://foo.example
Access-Control-Allow-Methods: POST, GET, OPTIONS
Access-Control-Allow-Headers: X-PINGOTHER, Content-Type
Access-Control-Max-Age: 86400
...
```

```
POST /doc HTTP/1.1
X-PINGOTHER: pingpong
Content-Type: text/xml; charset=UTF-8
Origin: http://foo.example
...
```

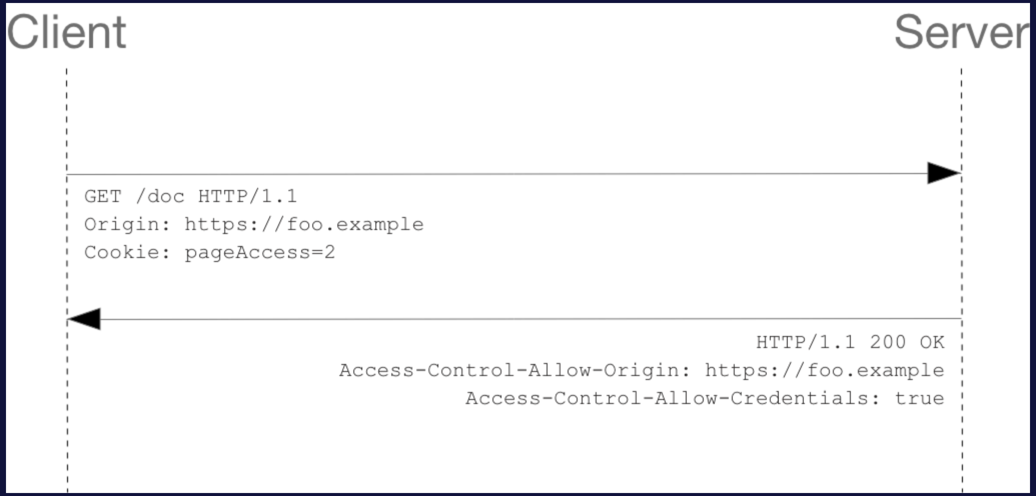
```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: http://foo.example
Vary: Accept-Encoding, Origin
Content-Encoding: gzip
Content-Length: 235
...
```

```
❯ curl -X OPTIONS --location "http://localhost:5220/Bar" \
    -H "Origin: "https://local.kefo.no"" \
    -H "Access-Control-Request-Method: POST" -v
*   Trying 127.0.0.1:5220...
* Connected to localhost (127.0.0.1) port 5220 (#0)
> OPTIONS /Bar HTTP/1.1
> Host: localhost:5220
> User-Agent: curl/8.0.1
> Accept: */*
> Origin: https://local.kefo.no
> Access-Control-Request-Method: POST
>
< HTTP/1.1 204 No Content
< Date: Tue, 21 Mar 2023 21:23:03 GMT
< Server: Kestrel
< Access-Control-Allow-Methods: POST
< Access-Control-Allow-Origin: https://local.kefo.no
```

CREDENTIAL REQUEST

are requests that contain:

- HTTP Cookie
- HTTP Authentication information.



```
GET /doc HTTP/1.1
Origin: https://foo.example
Cookie: pageAccess=2
```

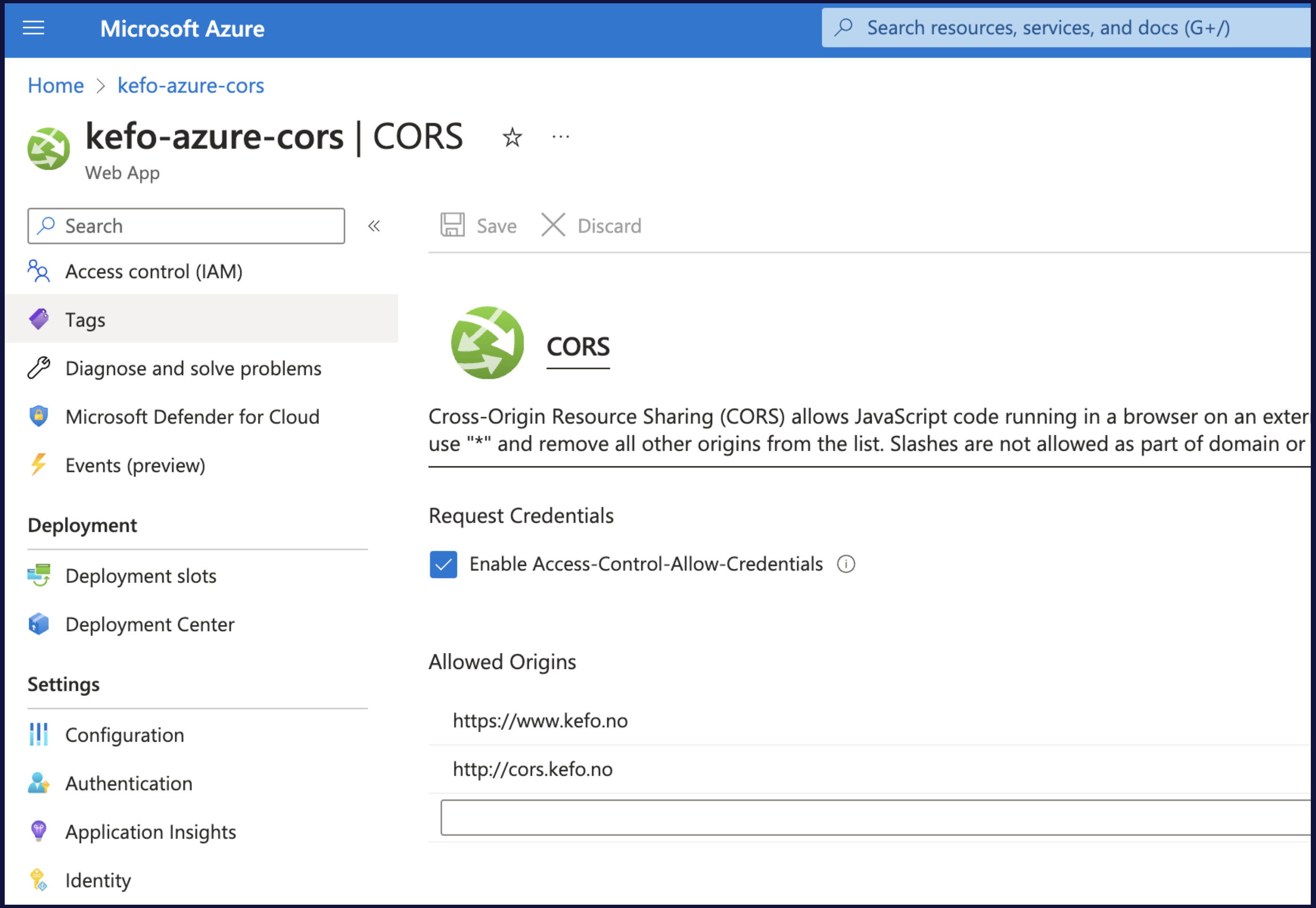
```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: https://foo.example
Access-Control-Allow-Credentials: true
```

CREDENTIAL REQUEST CONT.

- Not all are pre-flighted (e.g. a `GET` some requests)
- The server must specify: `Access-Control-Allow-Origin` without `*`
- The client (browser) will reject the *RESPONSE* if `Access-Control-Allow-Credentials: true` is missing.

AZURE APP SERVICE CORS

- Enabled or disabled?
- What does this checkbox mean?



kefo-azure-cors | CORS ☆ ···

Web App

CORS

Cross-Origin Resource Sharing (CORS) allows JavaScript code running in a browser on an exter
use "*" and remove all other origins from the list. Slashes are not allowed as part of domain or

Request Credentials

☑ Enable Access-Control-Allow-Credentials ⓘ

Allowed Origins

https://www.kefo.no

http://cors.kefo.no

AZURE DEFAULT POLICY

- Takes Precedence

For the list of Allowed Origins this is the policy:

- `AllowAllHeaders`
- `AllowAnyMethods`
- `IncludeCredentials` (selectable)

AZURE DEFAULTS

Returns

- 200 OK for ok requests (204)
- 400 Bad Request for non-compliant requests (405)

How to remove

- WebPage: unclick and remove all origins
- Cli: `az webapp cors remove --allow-origins -g off-cors -n kefo-azure-cors-settings`

ASP.NET WEBAPI CORS

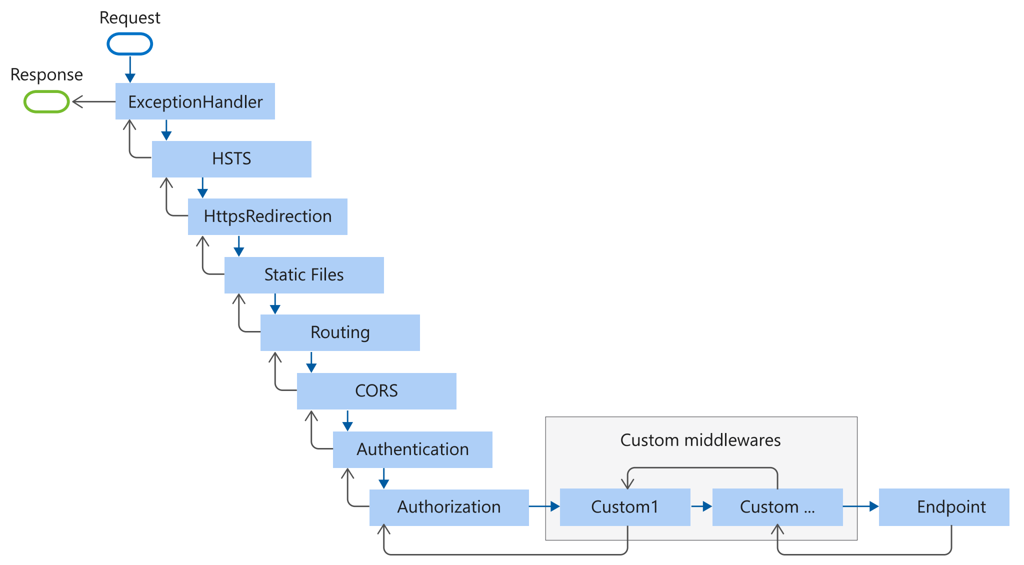
Is defined through:

- Middleware `Program.cs`
- Controller `Attributes`

CONFIGURATIONS

Only use:

- Attributes `[EnableCors("policyname")]`
- Middleware `app.UseCors()`



EXAMPLE: MIDDLEWARE ORDER

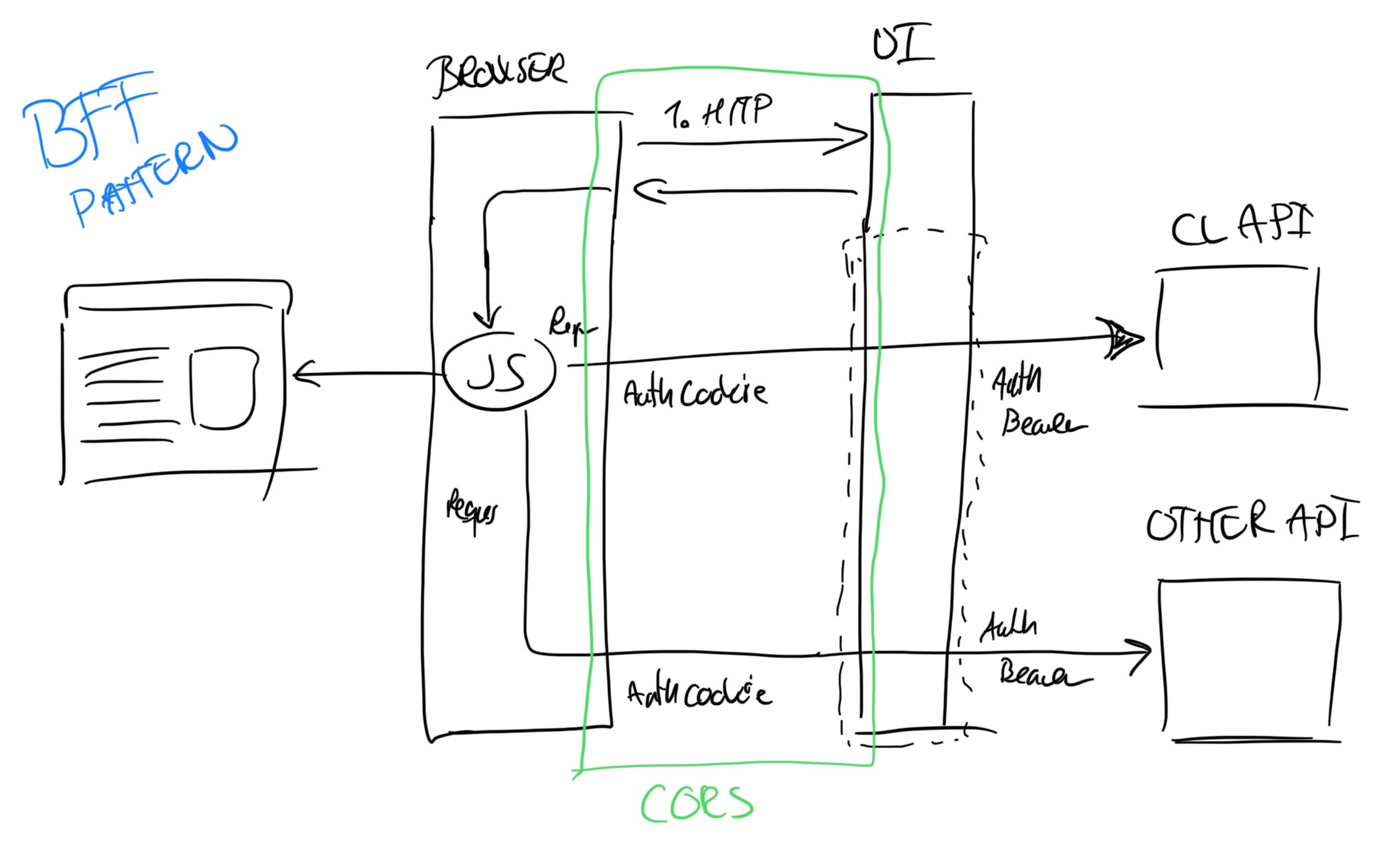
```
var builder = WebApplication.CreateBuilder(args);

// Add services to the container.
...
var app = builder.Build();

// Configure the HTTP request pipeline.
if (app.Environment.IsDevelopment())
{
    app.UseMigrationsEndPoint();
}
else
{
    app.UseExceptionHandler("/Error");
    app.UseHsts();
}

app.UseHttpsRedirection();
app.UseStaticFiles();
// app.UseCookiePolicy();

app.UseRouting();
// app.UseRequestLocalization();
// app.UseCors();

app.UseAuthentication();
app.UseAuthorization();
// app.UseSession();
// app.UseResponseCompression();
// app.UseResponseCaching();

app.MapRazorPages();
app.MapControllerRoute(
    name: "default",
    pattern: "{controller=Home}/{action=Index}/{id?}");

app.Run();
```

EXAMPLE POLICY

```
// configure Cors Policy
builder.Services.AddCors(
    p => p.AddDefaultPolicy(
        settings => settings
            .WithOrigins("https://local.kefo.no")
            .AllowAnyHeader()
            .AllowAnyMethod())
    );
app.UseCors();
```

```
// configure Cors Policy
var _policyName = "localkefo";
builder.Services.AddCors(
    p => p.AddPolicy(name: _localkefo,
        settings => settings
            .WithOrigins("https://local.kefo.no")
            .AllowAnyHeader()
            .AllowAnyMethod())
    );
app.UseCors(_policyName);
```

BFF



https://www.oreilly.com/library/view/building-microservices-2nd/9781492034018/

SOURCES

- Mozilla CORS
- Azure Specifics
- Implement CORS
- Tutorial: Host RESTful API with CORS - Azure App Service
- ASP.NET Core Middleware
- Fetch Standard
- GitHub spydx/kefo-azure-cors