# Apache Server Configs v2.11.0 / MIT License
# https://github.com/h5bp/server-configs-apache

# (!) Using `.htaccess` files slows down Apache, therefore, if you have
# access to the main server configuration file (which is usually called
# `httpd.conf`), you should add this logic there.
#

# ####################################################################
# # CROSS-ORIGIN
# # ####################################################################

# ---
# ! Cross-origin requests
# ---

# Allow cross-origin requests.
#
# https://developer.mozilla.org/en-US/docs/Web/HTTP/Access_control_CORS
# http://enable-cors.org/
# http://www.w3.org/TR/cors/

<IfModule mod_headers.c>
    Header set Access-Control-Allow-Origin "*
</IfModule>

# ---
# ! Cross-origin images
# ---

# Send the CORS header for images when browsers request it.
#

<IfModule mod_setenvif.c>
    <IfModule mod_headers.c>
        <FilesMatch "\.(bmp|cur|gif|ico|jpe|jpg|png|svg|xml|webp)$">
            SetEnvIf Origin ".* IS_CORS"
            Header set Access-Control-Allow-Origin "*" env=!IS_CORS
        </FilesMatch>
    </IfModule>
</IfModule>
header Set Access-Control-Allow-Origin = ENV-13.CORS
</FilesMatch>
</IfModule>
</IfModule>

#----------------------------------------------------------------------
# ! Cross-origin web fonts
#----------------------------------------------------------------------

# Allow cross-origin access to web fonts.

<IfModule mod_headers.c>
  <FilesMatch "\.eot\.otf\.tt[tf][c]l\.woff2?">$
    Header set Access-Control-Allow-Origin "*"
  </FilesMatch>
</IfModule>

#----------------------------------------------------------------------
# ! Cross-origin resource timing
#----------------------------------------------------------------------

# Allow cross-origin access to the timing information for all resources.
# If a resource isn't served with a `Timing-Allow-Origin` header that
# would allow its timing information to be shared with the document,
# some of the attributes of the `PerformanceResourceTiming` object will
# be set to zero.
# http://www.w3.org/TR/resource-timing/
# http://www.stevensouders.com/blog/2014/06/21/resource-timing-practical-tips/

# <IfModule mod_headers.c>
#   Header set Timing-Allow-Origin: "*"
# </IfModule>

#.Errorfs
#----------------------------------------------------------------------
# ! Custom error messages/pages
#----------------------------------------------------------------------

# Customize what Apache returns to the client in case of an error.
# https://httpd.apache.org/docs/current/mod/core.html#errordocument

ErrorDocument 404 /404.html

# Disable the pattern matching based on filenames.
# This setting prevents Apache from returning a 404 error as the result
# of a rewrite when the directory with the same name does not exist.
# https://httpd.apache.org/docs/current/content-negotiation.html#multiviews

Options -MultiViews

# Force Internet Explorer 8/9/10 to render pages in the highest mode
# available in the various cases when it may not.
# https://hsivonen.fi/doctype/#ie8

# (!) Starting with Internet Explorer 11, document modes are deprecated.
# If your business still relies on older web apps and services that were
# designed for older versions of Internet Explorer, you might want to
# consider enabling `Enterprise Mode` throughout your company.
<IfModule mod_headers.c>

  Header set X-UA-Compatible "IE=edge"
  # `mod_headers` cannot match based on the content-type, however,
  # the `X-UA-Compatible` response header should be send only for
  # HTML documents and not for the other resources.

<FilesMatch "^\.
(appcachef atoml bbaw1 bmpi cxml cssl curl eotl f4l(abpv)flvl geojsonl gifl htc1 icol jpe?
  gl jssl json(ld)?
  l m4l avl manifest mapl mp4l oexl og[agv]l opusl otfl pdfl pngfl rdfl rssl safarixtzl svgz?
  lswfl toposjsonl tt[cf]l txtl vcardl vcfl vttl webappl web[mp]l woff2?l xloc1 xml1 xpi)"$">

    Header unset X-UA-Compatible

  </FilesMatch>
</IfModule>

# ----------------------------------------------------------
#
# Iframes cookies
#
# ----------------------------------------------------------

# Allow cookies to be set from iframes in Internet Explorer.
#
# http://www.w3.org/TR/2000/CR-P3P-20001215/

# <IfModule mod_headers.c>
#    Header set P3P "policyref="/w3c/p3p.xml", CP="IDC DSP COR ADM DEVi TAIi PSA PSD
#    IVAi IVDi CONi HIS OUR IND CNT"
# </IfModule>

# ################################################################################
# # MEDIA TYPES AND CHARACTER ENCODINGS
# ################################################################################

# ----------------------------------------------------------
#
# Media types
#
# ----------------------------------------------------------

# Serve resources with the proper media types (f.k.a. MIME types).
#
# https://www.iana.org/assignments/media-types/media-types.xhtml
# https://httpd.apache.org/docs/current/mod/mod_mime.html#addtype
<IfModule mod_mime.c>

# Data interchange

AddType application/json json map topojson
AddType application/ld+json jsonld
AddType application/vnd.geo+json geojson
AddType application/xml atom rdf rss xml

# JavaScript

# Normalize to standard type.
# https://tools.ietf.org/html/rfc4329#section-7.2

AddType application/javascript js

# Manifest files

# If you are providing a web application manifest file (see
# the specification: https://w3c.github.io/manifest/), it is
# recommended that you serve it with the `application/manifest+json`
# media type.
#
# Because the web application manifest file doesn't have its
# own unique file extension, you can set its media type either
# by matching:
#
# 1) the exact location of the file (this can be done using a
#    directive such as `<Location>`, but it will NOT work in
#    the `.htaccess` file, so you will have to do it in the main
#    server configuration file or inside of a `<VirtualHost>`
#    container)
#
#    e.g.:
#
#    <Location "/.well-known/manifest.json">
#      AddType application/manifest+json json
#    </Location>
#
# 2) the filename (this can be problematic as you will need to
# ensure that you don't have any other file with the same name
# as the one you gave to your web application manifest file)
#
# e.g:
#
# <Files "manifest.json">
#    AddType application/manifest+json        json
# </Files>

AddType application/x-web-app-manifest+json webapp
AddType text/cache-manifest appcache manifest

# Media files

AddType audio/mp4 f4a f4b m4a
AddType audio/ogg oga ogg opus
AddType image/bmp bmp
AddType image/webp webp
AddType video/mp4 f4v f4p m4v mp4
AddType video/ogg ogv
AddType video/webm webm
AddType video/x-flv flv
AddType image/svg+xml svg svgz

# Serving `.ico` image files with a different media type
# prevents Internet Explorer from displaying then as images:
# https://github.com/h5bp/html5-
boilerplate/commit/37b5fec090d00f38de64b591bcddcb205aadf8ee

AddType image/x-icon cur ico

# Web fonts

AddType application/font-woff woff
AddType application/font-woff2 woff2
AddType application/vnd.ms-fontobject eot

# Browsers usually ignore the font media types and simply sniff
# the bytes to figure out the font type.
# https://mimesniff.spec.whatwg.org/#matching-a-font-type-pattern
#
# However, Blink and WebKit based browsers will show a warning in the console if the following font types are served with any other media types.

```
AddType application/x-font-ttf  ttc ttf
AddType font/opentype          otf

# Other

AddType application/octet-stream safarientz
AddType application/x-bb-appworld bbaw
AddType application/x-chrome-extension crx
AddType application/x-opera-extension oex
AddType application/x-xpinstall   xpi
AddType text/vcard              vcard vcf
AddType text/vnd.rim.location.xloc xloc
AddType text/vtt                vtt
AddType text/x-component        htc
```

```html
</IfModule>
```

# Character encodings

# Serve all resources labeled as `text/html` or `text/plain` with the media type `charset` parameter set to `UTF-8`.

```
AddDefaultCharset utf-8
```

```
# Serve the following file types with the media type `charset` parameter set to `UTF-8`.

```
<IfModule mod_mime.c>
    AddCharset utf-8 .atom 
</IfModule>
```
# #Rewrites

# (1) Turn on the rewrite engine (this is necessary in order for
# the `RewriteRule` directives to work).
#
# https://httpd.apache.org/docs/current/mod/mod_rewrite.html#RewriteEngine
#
# (2) Enable the `FollowSymlinks` option if it isn't already.
#
# https://httpd.apache.org/docs/current/mod/core.html#options
#
# (3) If your web host doesn't allow the `FollowSymlinks` option,
# you need to comment it out or remove it, and then uncomment
# the `Options +SymLinksIfOwnerMatch` line (4), but be aware
# of the performance impact.
#
# https://httpd.apache.org/docs/current/misc/perf-tuning.html#symlinks
#
# (4) Some cloud hosting services will require you set `RewriteBase`.
#
http://www.rackspace.com/knowledge_center/frequently-asked-question/why-is-modrewrite-not-working-on-my-site

https://httpd.apache.org/docs/current/mod/mod_rewrite.html#rewritebase

(5) Depending on how your server is set up, you may also need to
use the `RewriteOptions` directive to enable some options for
the rewrite engine.

https://httpd.apache.org/docs/current/mod/mod_rewrite.html#rewriteoptions

```html
<IfModule mod_rewrite.c>

# (1)
RewriteEngine On

# (2)
Options +FollowSymlinks

# (3)
# Options +SymLinksIfOwnerMatch

# (4)
# RewriteBase /

# (5)
# RewriteOptions <options>

# Adaptive-Images -----------------------------------------------

# Add any directories you wish to omit from the Adaptive-Images process on a new line, as follows:
# RewriteCond %{REQUEST_URI} !ignore-this-directory
# RewriteCond %{REQUEST_URI} !and-ignore-this-directory-too

RewriteCond %{REQUEST_URI} !assets

# don't apply the AI behaviour to images inside AI's cache folder:
RewriteCond %{REQUEST_URI} !ai-cache

# Send any GIF .JPG or PNG request that IS NOT stored inside one of the above
# Send any exit one of the request that is not stored inside one of the above directories
# to adaptive-images.php so we can select appropriately sized versions

RewriteRule \.(?:jpe?g|gif|png)$ adaptive-images.php

# END Adaptive-Images

</IfModule>

# ________________________________________________________________________________
# ! Forcing `https://`
# ________________________________________________________________________________

# Redirect from the `http://` to the `https://` version of the URL.
# https://wiki.apache.org/httpd/RewriteHTTPToHTTPS

# <IfModule mod_rewrite.c>
#    RewriteEngine On
#    RewriteCond %{HTTPS} !=on
#    RewriteRule ^.*$ https://%{HTTP_HOST}/$1 [R=301,L]
# </IfModule>

# ________________________________________________________________________________
# ! Suppressing `/` Forcing the `www.` at the beginning of URLs
# ________________________________________________________________________________

# The same content should never be available under two different URLs, especially not with and without `www.` at the beginning.
# This can cause SEO problems (duplicate content), and therefore, you should choose one of the alternatives and redirect the other one.
#
# By default `Option 1` (no `www.`) is activated.
# http://no-www.org/faq.php?q=class_b
#
# If you would prefer to use `Option 2`, just comment out all the lines from `Option 1` and uncomment the ones from `Option 2`.
#
# (!) NEVER USE BOTH RULES AT THE SAME TIME!
# Option 1: rewrite www.example.com → example.com

```html
<IfModule mod_rewrite.c>
  RewriteEngine On
  RewriteCond %{HTTPS} !=on
  RewriteCond %{HTTP_HOST} ^www\.(.+)$ [NC]
  RewriteRule ^ http://%1 %{REQUEST_URI} [R=301,L]
</IfModule>
```

# Option 2: rewrite example.com → www.example.com

# Be aware that the following might not be a good idea if you use "real" subdomains for certain parts of your website.

```html
# <IfModule mod_rewrite.c>
#   RewriteEngine On
#   RewriteCond %{HTTPS} !=on
#   RewriteCond %{HTTP_HOST} ^www\.(.+)$ [NC]
#   RewriteCond %{SERVER_ADDR} !=127.0.0.1
#   RewriteCond %{SERVER_ADDR} !=:1
#   RewriteRule ^ http://www %{REQUEST_URI} [R=301,L]
# </IfModule>
```

#ppard#pledge

# SECURITY

# Protect website against clickjacking.

# The example below sends the `X-Frame-Options` response header with the value `DENY`, informing browsers not to display the content of the web page in any frame.

"
# This might not be the best setting for everyone. You should read
# about the other two possible values the `X-Frame-Options` header
# field can have: `SAMEORIGIN` and `ALLOW-FROM`.
#
# Keep in mind that while you could send the `X-Frame-Options` header
# for all of your website’s pages, this has the potential downside that
# it forbids even non-malicious framing of your content (e.g.: when
# users visit your website using a Google Image Search results page).
#
# Nonetheless, you should ensure that you send the `X-Frame-Options` header
# for all pages that allow a user to make a state changing
# operation (e.g: pages that contain one-click purchase links, checkout
# or bank-transfer confirmation pages, pages that make permanent
# configuration changes, etc.).
#
# Sending the `X-Frame-Options` header can also protect your website
# against more than just clickjacking attacks;
#
# https://tools.ietf.org/html/rfc7034
# https://www.owasp.org/index.php/Clickjacking
#
# <IfModule mod_headers.c>
# Header set X-Frame-Options "DENY"
# # `mod_headers` cannot match based on the content-type, however,
# # the `X-Frame-Options` response header should be send only for
# # HTML documents and not for the other resources.
# <FilesMatch "\.\(appcache|atoml|baw|bml|cr|x|csl|curl|eot|f4\[abp\]|f1|gif|htx|ico|jpg|jpe?|gl|js|json\(ld\)?|im|manifest|mapl|mp4lx|og\[agv]\]|opus|otf|pdf|pngl|rdfl|rsslx|safariextzl|svgz\?|swfl|topojzonl|tt\[cf]\]|txtlx|vcardlx|vcfl|vttlx|webapp|web\[mp]\]|woff2?|\x1ocl|xml|xpi\)\$">
# Header unset X-Frame-Options
# </FilesMatch>
# </IfModule>
# I Content Security Policy (CSP)

# Mitigate the risk of cross-site scripting and other content-injection
# attacks.

# This can be done by setting a `Content Security Policy` which
# whitelists trusted sources of content for your website.

# The example header below allows ONLY scripts that are loaded from the
# current website's origin (no inline scripts, no CDN, etc). That almost
# certainly won't work as-is for your website!

# For more details on how to craft a reasonable policy for your website,
# (or the specification: http://www.w3.org/TR/CSP11/). Also, to make
# things easier, you can use an online CSP header generator such as:
# http://cspisawesome.com/.

# <IfModule mod_headers.c>
# Header set Content-Security-Policy "script-src 'self'; object-src 'self'"
# # `mod_headers` cannot match based on the content-type, however,
# # the `Content-Security-Policy` response header should be send
# # only for HTML documents and not for the other resources.
# # <FilesMatch ".*">
# <appcachel atoml bbawl bmpl crxl cssl curl eot f4[labpv]1 f1v1 geojsonl gifl htcl icol jpe?
# gl jsli json(1d)?
# |m4avj manifest mapl mp4 oexl og[agv]1 opusl otfl pdfl pngl rdfl rssl safarilextzi svgz?
# |swfl topjsonl tt[cf]1 txtl vcardl vcf1 vttl webappl web[mp]1 woff2?! xlocl xml1 xpi">"
# # Header unset Content-Security-Policy
# # </FilesMatch>
# </IfModule>

# File access

# Block access to directories without a default document.

# You should leave the following uncommented, as you shouldn't allow
# anyone to surf through every directory on your server (which may
# includes rather private places such as the CMS's directories.

<IfModule mod_autoindex.c>
  Options -Indexes
</IfModule>

# Block access to all hidden files and directories with the exception of
# the visible content from within the `./well-known/` hidden directory.
#
# These types of files usually contain user preferences or the preserved
# state of an utility, and can include rather private places like, for
# example, the `.git` or `.svn` directories.
#
# The `./well-known/` directory represents the standard (RFC 5785) path
# prefix for "well-known locations" (e.g.: `./well-known/manifest.json`,
# `./well-known/keybase.txt`), and therefore, access to its visible
# content should not be blocked.
#
# https://www.mnot.net/blog/2010/04/07/well-known
# https://tools.ietf.org/html/rfc5785

<IfModule mod_rewrite.c>
  RewriteEngine On
  RewriteCond %{REQUEST_URI} "!([^/\./.well-known/[^/\]+/\?]+)" [NC]
  RewriteCond %{SCRIPT_FILENAME} -d [OR]
  RewriteCond %{SCRIPT_FILENAME} -f
  RewriteRule "\.(git|svn)" - [F]
</IfModule>

# Block access to files that can expose sensitive information.
#
# By default, block access to backup and source files that may be
# left by some text editors and can pose a security risk when anyone
# has access to them.
#
# http://feross.org/cmssploit/
Update the `<FilesMatch>` regular expression from below to include any files that might end up on your production server and can expose sensitive information about your website. These files may include: configuration files, files that contain metadata about the project (e.g.: project dependencies), build scripts, etc.

```
<FilesMatch "(^#.*#|\.\(bak|conf|dist|fla|incl|log|psd|sh|sql|sw[og]p)\)\$">
  
  # Apache < 2.3
  <IfModule !mod_authz_core.c>
    Order allow,deny
    Deny from all
    Satisfy All
  </IfModule>

  # Apache ≥ 2.3
  <IfModule mod_authz_core.c>
    Require all denied
  </IfModule>

</FilesMatch>
```

# Force client-side SSL redirection.
#
# If a user types `example.com` in their browser, even if the server redirects them to the secure version of the website, that still leaves a window of opportunity (the initial HTTP connection) for an attacker to downgrade or redirect the request.
#
# The following header ensures that browser will ONLY connect to your server via HTTPS, regardless of what the users type in the browser's address bar.
#
# (!) Remove the `includeSubDomains` optional directive if the website's subdomains are not using HTTPS.

# <IfModule mod_headers.c>
#   Header set Strict-Transport-Security "max-age=16070400; includeSubDomains"
# </IfModule>

# 1 Reducing MIME type security risks

# Prevent some browsers from MIME-sniffing the response.
#
# This reduces exposure to drive-by download attacks and cross-origin data leaks, and should be left uncommented, especially if the server is serving user-uploaded content or content that could potentially be treated as executable by the browser.
#
# http://www.slideshare.net/hasegawayosuke/owasp-hasegawa
# https://mimesniff.spec.whatwg.org/

<IfModule mod_headers.c>
   Header set X-Content-Type-Options "nosniff"
</IfModule>

# 1 Reflected Cross-Site Scripting (XSS) attacks

# (1) Try to re-enable the cross-site scripting (XSS) filter built into most web browsers.
#
# The filter is usually enabled by default, but in some cases it may be disabled by the user. However, in Internet Explorer for example, it can be re-enabled just by sending the `X-XSS-Protection` header with the value of `1`. 
#
# (1) Prevent web browsers from rendering the web page if a potential
# reflected (a.k.a non-persistent) XSS attack is detected by the
# filter.
#
# By default, if the filter is enabled and browsers detect a
# reflected XSS attack, they will attempt to block the attack
# by making the smallest possible modifications to the returned
# web page.
#
# Unfortunately, in some browsers (e.g.: Internet Explorer),
# this default behavior may allow the XSS filter to be exploited,
# thereby, it's better to inform browsers to prevent the rendering
# of the page altogether, instead of attempting to modify it.
#
#
# (!) Do not rely on the XSS filter to prevent XSS attacks! Ensure that
# you are taking all possible measures to prevent XSS attacks, the
# most obvious being: validatign and sanitizing your website's inputs.
#
# [![OWASP](https://www.owasp.org/index.php/Cross-site_Scripting_%28XSS%29)](https://www.owasp.org/index.php/Cross-site_Scripting_%28XSS%29)

# <IfModule mod_headers.c>
#    # (1) (2)
#    # Header set X-XSS-Protection "1; mode=block"
#    # `mod_headers` cannot match based on the content-type, however,
#    # the `X-XSS-Protection` response header should be send only for
#    # HTML documents and not for the other resources.
#    <FilesMatch "
(appcache|atom|bml|br|c|css|r|curl|eot|f4|[abp]|f|fl|gif|htc|ico|jpe?|gl|js|json(1d)?)?
/l4[av]|manifest|map|m|mp4|m|oex|x|og[agv]|opus|otf|pdf|png|rdf|rss|safari_extz|svgz?
/swf|topojson|ttt[cf]|txt|vcard|vcf|vtt|webapp|web[mp]|woff2?[1|خل|xoc|xxml|xpi]"">
#     Header unset X-XSS-Protection
# </FilesMatch>
# </IfModule>
# Server software information
#
# Prevent Apache from sending in the `Server` response header its exact version number, the description of the generic OS-type or information about its compiled-in modules.
#
# (!) The `ServerTokens` directive will only work in the main server configuration file, so don't try to enable it in the `.htaccess` file!
#
# https://httpd.apache.org/docs/current/mod/core.html#server_tokens

# ServerTokens Prod

# WEB PERFORMANCE
#
# Compress all output labeled with one of the following media types.
#
# (!) For Apache versions below version 2.3.7 you don't need to enable `mod_filter` and can remove the `</IfModule mod_filter.c>`

<IfModule mod_deflate.c>
    # Force compression for mangled `Accept-Encoding` request headers

    <IfModule mod_setenvif.c>
        <IfModule mod_headers.c>
            SetEnvIfNoCase "^Accept-Encoding$ x-cept-encoding|X-cept-Encoding|X-{15}|" ~(15) -1(15)\$ "^((gzip|deflate)\.\*,?\s*)+1 [x"-]{4,13}\$ HAVE_Accept-Encoding
            RequestHeader append Accept-Encoding "gzip, deflate" env=HAVE_Accept-Encoding
        </IfModule>
    </IfModule>
</IfModule>

# Compress all output labeled with one of the following media types.
#
# (!) For Apache versions below version 2.3.7 you don't need to enable `mod_filter` and can remove the `</IfModule mod_filter.c>`
<IfModule mod_filter.c>

AddOutputFilterByType DEFLATE "application/atom+xml" \
  "application/javascript" \
  "application/json" \
  "application/ld+json" \
  "application/manifest+json" \
  "application/rdf+xml" \
  "application/rss+xml" \
  "application/schema+json" \
  "application/vnd.geo+json" \
  "application/vnd.ms-fontobject" \
  "application/x-font-ttf" \
  "application/x-javascript" \
  "application/x-web-app-manifest+json" \
  "application/xhtml+xml" \
  "application/xml" \
  "font/eot" \
  "font/opentype" \
  "image/bmp" \
  "image/svg+xml" \
  "image/vnd.microsoft.icon" \
  "image/x-icon" \
  "text/cache-manifest" \
  "text/css" \
  "text/html" \
  "text/javascript" \
  "text/plain" \
  "text/vcard" \
  "text/vnd.rim.location.xloc" \
  "text/vtt" \
  "text/x-component" \
  "text/x-cross-domain-policy" \
  "text/xml"

</IfModule>
# Map the following filename extensions to the specified encoding type in order to make Apache serve the file types with the appropriate `Content-Encoding` response header. (do note that this will NOT make Apache compress them!).

# If these file types would be served without an appropriate `Content-Enable` response header, client applications (e.g.: browsers) wouldn't know that they first need to uncompress the response, and thus, wouldn't be able to understand the content.

# https://httpd.apache.org/docs/current/mod/mod_mime.html#addencoding

```xml
<IfModule mod_mime.c>
  AddEncoding gzip svgz
</IfModule>
</IfModule>
```

# Prevent intermediate caches or proxies (e.g.: such as the ones used by mobile network providers) from modifying the website's content.


# (!!) If you are using `mod_pagespeed`, please note that setting the `Cache-Control: no-transform` response header will prevent `PageSpeed` from rewriting `HTML` files, and, if the `ModPagespeedDisableRewriteOnNoTransform` directive isn't set to `off`, also from rewriting other resources.

# https://developers.google.com/speed/pagespeed/module/configuration#notransform

# <IfModule mod_headers.c>
```

```
#   Header merge Cache-Control "no-transform"
# </IfModule>

# ------------------------------------------------------------------------
#  ETags
# ------------------------------------------------------------------------

# Remove `ETags` as resources are sent with far-future expires headers.
# https://developer.yahoo.com/performance/rules.html#etags
# https://tools.ietf.org/html/rfc7232#section-2.3

`FileETag None` doesn't work in all cases.
<IfModule mod_headers.c>
    Header unset ETag
</IfModule>

FileETag None

# ------------------------------------------------------------------------
#  Expires headers
# ------------------------------------------------------------------------

# Serve resources with far-future expires headers.
# (!) If you don't control versioning with filename-based
# cache busting, you should consider lowering the cache times
# to something like one week.
# https://httpd.apache.org/docs/current/mod/mod_expires.html

<IfModule mod_expires.c>

    ExpiresActive on
    ExpiresDefault "access plus 1 month"

    # CSS
    ExpiresByType text/css "access plus 1 year"

    # Data interchange
    ExpiresByType application/atom+xml "access plus 1 hour"
<table>
<thead>
<tr>
<th>ExpiresByType</th>
<th>Content Type</th>
<th>Expires</th>
</tr>
</thead>
<tbody>
<tr>
<td>ExpiresByType application/rdf+xml</td>
<td></td>
<td>&quot;access plus 1 hour&quot;</td>
</tr>
<tr>
<td>ExpiresByType application/rss+xml</td>
<td></td>
<td>&quot;access plus 1 hour&quot;</td>
</tr>
<tr>
<td>ExpiresByType application/json</td>
<td></td>
<td>&quot;access plus 0 seconds&quot;</td>
</tr>
<tr>
<td>ExpiresByType application/ld+json</td>
<td></td>
<td>&quot;access plus 0 seconds&quot;</td>
</tr>
<tr>
<td>ExpiresByType application/schema+json</td>
<td></td>
<td>&quot;access plus 0 seconds&quot;</td>
</tr>
<tr>
<td>ExpiresByType application/vnd.geo+json</td>
<td></td>
<td>&quot;access plus 0 seconds&quot;</td>
</tr>
<tr>
<td>ExpiresByType application/xml</td>
<td></td>
<td>&quot;access plus 0 seconds&quot;</td>
</tr>
<tr>
<td>ExpiresByType text/xml</td>
<td></td>
<td>&quot;access plus 0 seconds&quot;</td>
</tr>
</tbody>
</table>

# Favicon (cannot be renamed!) and cursor images
<table>
<thead>
<tr>
<th>ExpiresByType</th>
<th>Content Type</th>
<th>Expires</th>
</tr>
</thead>
<tbody>
<tr>
<td>ExpiresByType image/vnd.microsoft.icon</td>
<td></td>
<td>&quot;access plus 1 week&quot;</td>
</tr>
<tr>
<td>ExpiresByType image/x-icon</td>
<td></td>
<td>&quot;access plus 1 week&quot;</td>
</tr>
</tbody>
</table>

# HTML
<table>
<thead>
<tr>
<th>ExpiresByType</th>
<th>Content Type</th>
<th>Expires</th>
</tr>
</thead>
<tbody>
<tr>
<td>ExpiresByType text/html</td>
<td></td>
<td>&quot;access plus 0 seconds&quot;</td>
</tr>
</tbody>
</table>

# JavaScript
<table>
<thead>
<tr>
<th>ExpiresByType</th>
<th>Content Type</th>
<th>Expires</th>
</tr>
</thead>
<tbody>
<tr>
<td>ExpiresByType application/javascript</td>
<td></td>
<td>&quot;access plus 1 year&quot;</td>
</tr>
<tr>
<td>ExpiresByType application/x-javascript</td>
<td></td>
<td>&quot;access plus 1 year&quot;</td>
</tr>
<tr>
<td>ExpiresByType text/javascript</td>
<td></td>
<td>&quot;access plus 1 year&quot;</td>
</tr>
</tbody>
</table>

# Manifest files
<table>
<thead>
<tr>
<th>ExpiresByType</th>
<th>Content Type</th>
<th>Expires</th>
</tr>
</thead>
<tbody>
<tr>
<td>ExpiresByType application/manifest+json</td>
<td></td>
<td>&quot;access plus 1 year&quot;</td>
</tr>
<tr>
<td>ExpiresByType application/x-web-app-manifest+json</td>
<td></td>
<td>&quot;access plus 0 seconds&quot;</td>
</tr>
<tr>
<td>ExpiresByType text/cache-manifest</td>
<td></td>
<td>&quot;access plus 0 seconds&quot;</td>
</tr>
</tbody>
</table>

# Media files
<table>
<thead>
<tr>
<th>ExpiresByType</th>
<th>Content Type</th>
<th>Expires</th>
</tr>
</thead>
<tbody>
<tr>
<td>ExpiresByType audio/ogg</td>
<td></td>
<td>&quot;access plus 1 month&quot;</td>
</tr>
<tr>
<td>ExpiresByType image/bmp</td>
<td></td>
<td>&quot;access plus 1 month&quot;</td>
</tr>
<tr>
<td>ExpiresByType image/gif</td>
<td></td>
<td>&quot;access plus 1 month&quot;</td>
</tr>
<tr>
<td>ExpiresByType image/jpeg</td>
<td></td>
<td>&quot;access plus 1 month&quot;</td>
</tr>
<tr>
<td>ExpiresByType image/png</td>
<td></td>
<td>&quot;access plus 1 month&quot;</td>
</tr>
<tr>
<td>ExpiresByType image/svg+xml</td>
<td></td>
<td>&quot;access plus 1 month&quot;</td>
</tr>
<tr>
<td>ExpiresByType video/mp4</td>
<td></td>
<td>&quot;access plus 1 month&quot;</td>
</tr>
<tr>
<td>ExpiresByType video/ogg</td>
<td></td>
<td>&quot;access plus 1 month&quot;</td>
</tr>
<tr>
<td>ExpiresByType video/webm</td>
<td></td>
<td>&quot;access plus 1 month&quot;</td>
</tr>
</tbody>
</table>

# Web fonts
# Embedded OpenType (EOT)
ExpiresByType application/vnd.ms-fontobject "access plus 1 month"
ExpiresByType font/eot "access plus 1 month"

# OpenType
ExpiresByType font/opentype "access plus 1 month"

# TrueType
ExpiresByType application/x-font-ttf "access plus 1 month"

# Web Open Font Format (WOFF) 1.0
ExpiresByType application/font-woff "access plus 1 month"
ExpiresByType application/x-font-woff "access plus 1 month"
ExpiresByType font/woff "access plus 1 month"

# Web Open Font Format (WOFF) 2.0
ExpiresByType application/font-woff2 "access plus 1 month"

# Other
ExpiresByType text/x-cross-domain-policy "access plus 1 week"

</IfModule>

# ---------------------------------------------------------------
# / File concatenation                                       /
# ---------------------------------------------------------------

# Allow concatenation from within specific files.
#
# e.g.:
#
# If you have the following lines in a file called, for
# example, `main.combined.js`:
#
# <![--#include file="js/jquery.js" -->
# <![--#include file="js/jquery.timer.js" -->
#
# Apache will replace those lines with the content of the
# specified files.
#
# <IfModule mod_include.c>
<FilesMatch "\.combined\.js$">
  Options +Includes
  AddOutputFilterByType INCLUDES application/javascript
  AddOutputFilterByType INCLUDES application/x-javascript
  AddOutputFilterByType INCLUDES text/javascript
  SetOutputFilter INCLUDES
</FilesMatch>
</IfModule>

#  ----------------------------------------------------------------------
#  ! Filename-based cache busting                                      !
#  !  ----------------------------------------------------------------------

# If you're not using a build process to manage your filename version
# revving, you might want to consider enabling the following directives
# to route all requests such as `!/style.12345.css` to `!/style.css`.
# To understand why this is important and even a better solution than
# using something like `*.css?v23!`, please see:
# http://www.stevesouders.com/blog/2008/08/23/revving-filenames-dont-use-querystring/

<IfModule mod_rewrite.c>
  RewriteEngine On
  RewriteCond %{REQUEST_FILENAME} !-f
  RewriteRule ^(.+)/(\d+)/(bmp|css|curl|gif|ico|jpe?g|js|png|svgz?|webp)$ $1.$3 [L]
</IfModule>