```
## Use the front controller as index file. It serves as a fallback solution when
## every other rewrite/redirect fails (e.g. in an aliased environment without
## mod_rewrite). Additionally, this reduces the matching process for the
## start page (path "/") because otherwise Apache will apply the rewriting rules
## to each configured DirectoryIndex file (e.g. index.php, index.html, index.pl).
#DirectoryIndex app.php
#
#<IfModule mod_rewrite.c>
#    RewriteEngine On
#
#    # Determine the RewriteBase automatically and set it as environment variable.
#    # If you are using Apache aliases to do mass virtual hosting or installed the
#    # project in a subdirectory, the base path will be prepended to allow proper
#    # resolution of the app.php file and to redirect to the correct URI. It will
#    # work in environments without path prefix as well, providing a safe, one-size
#    # fits all solution. But as you do not need it in this case, you can comment
#    # the following 2 lines to eliminate the overhead.
#    RewriteCond %{REQUEST_URI}::$1 ^(/.+)/(.*)::\2$
#    RewriteRule ^(.*) - [E=BASE:%1]
#
#    # Sets the HTTP_AUTHORIZATION header removed by apache
#    RewriteCond %{HTTP:Authorization} .
#    RewriteRule .* - [E=HTTP_AUTHORIZATION:%{HTTP:Authorization}]
#
#    # Redirect to URI without front controller to prevent duplicate content
#    # (with and without `/app.php`). Only do this redirect on the initial
#    # rewrite by Apache and not on subsequent cycles. Otherwise we would get an
#    # endless redirect loop (request -> rewrite to front controller ->
#    # redirect -> request -> ...).
#    # So in case you get a "too many redirects" error or you always get redirected
#    # to the start page because your Apache does not expose the REDIRECT_STATUS
#    # environment variable, you have 2 choices:
#    # - disable this feature by commenting the following 2 lines or
#    # - use Apache >= 2.3.9 and replace all L flags by END flags and remove the
#    #   following RewriteCond (best solution)
#    RewriteCond %{ENV:REDIRECT_STATUS} ^$
#    RewriteRule ^app\.php(/(.*)|$) %{ENV:BASE}/$2 [R=301,L]
#
#    # If the requested filename exists, simply serve it.
#    # We only want to let Apache serve files and not directories.
#    RewriteCond %{REQUEST_FILENAME} -f
#    RewriteCond %{REQUEST_FILENAME} -f
#    RewriteRule .? - [L]
#
#    # Rewrite all other queries to the front controller.
#    RewriteRule .? %{ENV:BASE}/app.php [L]
#</IfModule>
#
#<IfModule !mod_rewrite.c>
#    <IfModule mod_alias.c>
#        # When mod_rewrite is not available, we instruct a temporary redirect of
#        # the start page to the front controller explicitly so that the website
#        # and the generated links can still be used.
#        RedirectMatch 302 ^/$ /app.php/
#        # RedirectTemp cannot be used instead
#    </IfModule>
#</IfModule>

<IfModule mod_rewrite.c>
    RewriteEngine On
    RewriteCond %{REQUEST_URI}::$1 ^(/.+)/(.*)::\2$
    RewriteRule ^(.*) - [E=BASE:%1]
    RewriteCond %{ENV:REDIRECT_STATUS} ^$
    RewriteRule ^app_dev.php(/(.*)|$) %{ENV:BASE}/$2 [R=301,L]     ##### this is the part
that you     should tweak, have the .htaccess point the request to app_dev.php, since the
routing.yml is empty initially
    RewriteCond %{REQUEST_FILENAME} -f
    RewriteRule .? - [L]
    RewriteRule .? %{ENV:BASE}/app_dev.php [L]        ##### this is the part that you
should tweak, have the .htaccess point the request to app_dev.php, since the routing.yml
is empty initially
</IfModule>

<IfModule !mod_rewrite.c>
    <IfModule mod_alias.c>
        # When mod_rewrite is not available, we instruct a temporary redirect of
        # the startpage to the front controller explicitly so that the website
        # and the generated links can still be used.
        RedirectMatch 302 ^/$ /app.php/
        # RedirectTemp cannot be used instead
    </IfModule>
</IfModule>
```