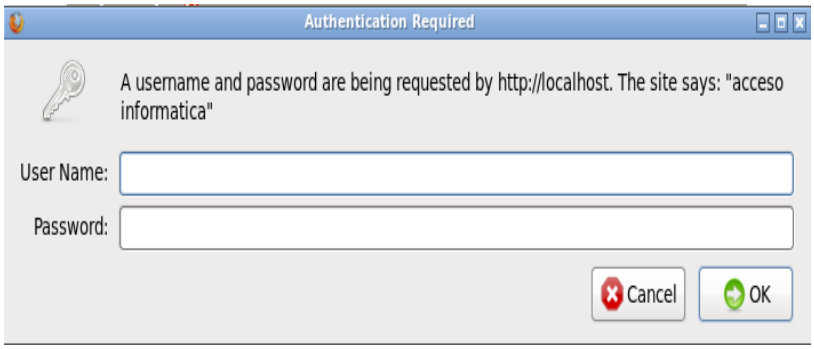
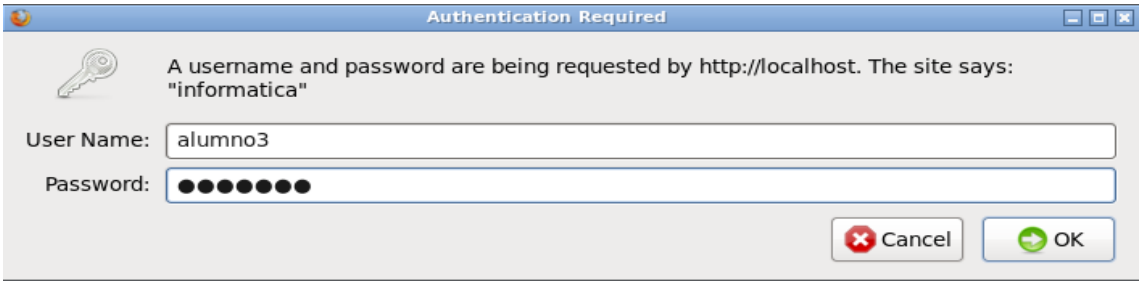
contraseñas (*AuthUserFile*), el archivo que contiene los grupos (*AuthGroupFile*) y quien podrá acceder (Require user alumno2 (usuarios), group gdeap (grupos) o valid-user (todo usuario validado en el archivo)). Tras reiniciar el servicio apache, al intentar acceder mediante el navegador a localhost/DEAP o a localhost/DEWSV pedirá los datos correspondientes.

```
<Directory /var/www/DEAP/>
        AuthType Basic
        AuthName "acceso informatica"
        AuthUserFile /etc/apache2/dajoevbc
        Require valid-user
</Directory>
<Directory /var/www/DEWNC>
        AuthType Digest
        AuthName "informatica"
        AuthUserFile /etc/apache2/dajoevdg
        AuthGroupFile /etc/apache2/grudeap
        Require group gdewnc
</Directory>
```





FICHEROS .htaccess y mod_rewrite

El archivo **.htaccess** es un archivo de configuración de directorios de Apache utilizado para personalizar la configuración de directivas y parámetros definidos en el archivo de configuración principal del alojamiento por parte de los administradores de páginas web. Tiene una gran variedad de usos, entre las que destacan las operaciones realizadas directamente en los apartados anteriores y la **reescritura de URLs** para hacerlas amigables, utilizar un dominio concreto, evitar hotlinking o presentar distintas páginas según unas condiciones como el navegador utilizado o el día actual. Dado que no se ha explicado esta característica, para practicar con los archivos vamos a utilizarla.

El conocido archivo .htaccess podría llamarse de otra forma, dado que proviene de la directiva **AccessFileName** en apache2.conf, con dicho valor por defecto, pero su repercusión y su utilidad hace que en todos los hosting conocidos que permiten dicha habilidad los archivos sean los mencionados. El archivo .htaccess debe colocarse en el interior de la carpeta donde se quiere que tenga efecto. Además de las directivas mencionadas hay que tener en cuenta **AllowOverride**, mediante la cual se indica que tipo de directivas o pueden dar problemas de servidor de las presentes en el contenido del fichero, va a poder atender el servidor y cuales va a omitir. Las opciones para esta directiva son *None* (Ninguna), *All* u opciones válidas entre las siguientes posibilidades: AuthConfig, FileInfo, Indexes, Limit y Options.

Para utilizar el módulo **Rewrite** el primer paso es comprobar si está habilitado y habilitarlo en caso de que no lo esté. Es un módulo muy extenso y completo que aplica un patrón un patrón a una URL y realiza sustituciones en ella como operación más básica y por ella será la utilizada como punto de inicio. Las reglas se escriben mediante la directiva **RewriteRule** basándose en patrones de búsqueda para reconocer los valores sobre los que debe reescribir. Para poder hacer patrones elaborados, primeramente hay que conocer los caracteres disponibles:

- **.** (cualquier carácter)
- ***** (0 ó + caracteres precedentes)
- **+** (1 ó + caracteres precedentes)
- **{}** (nº caracteres contiguos; 2 valores separados por , el 2º indica nº máximo repeticiones)

- **?** (0 ó 1 vez)
- **$** (final de la cadena)
- ^ (inicio de la cadena ó que no incluya dicho carácter si se encuentra al inicio de un rango [])
- [] (agrupar caracteres)
- **-** (rango de caracteres en [])
- **()** (grupo de caracteres)
- **|** (indicar varias opciones, utilizado principalmente junto a ())
- \ (carácter de escape, el siguiente carácter toma o deja de tener un sentido especial, por ejemplo, \. no sería cualquier carácter, sino específicamente el punto ó \d representa cualquier dígito del 0 al 9, \s un espacio en blanco, \D cualquiera que no sea un dígito, etc.)

También es importante conocer los **flags** (banderas) utilizados al final de las directivas de reglas para que Apache conozca como interpretarla y manejarla, por ejemplo, que no sea case-sensitive, detener el proceso si no coincide una comparación, etc. Se indican entre córchetes y pueden utilizarse varias en una misma regla separadas por comas.

- **C** (encadena la siguiente regla)
- **CO=cookie** (especifica cookies)
- **E=var:value** (establece valor a una variable)
- **F** (envía una cabecera 403 al usuario)
- **G** (muestra cabecera 410 de recurso que estuvo disponible, ya no lo está)
- **H=handler**(especifica un manejador concreto)
- **L** (termina de procesar reglas)
- **N** (procesa la regla hasta que no se cumpla)
- **NC** (no es case-sensitive)
- **NE** (no utiliza caracteres especiales como #)
- **NS** (ignora la regla si es una subpetición)
- **P** (maneja como proxy)
- **PT** (procesa URLs con mods adiccionales, ej. alias)
- **QSA** (cuando la URL contiene la cadena buscada, el comportamiento habitual es descartar la cadena buscada y sustituirla por la generada, pero con ésta bandera se combina)
- **R** (redirección temporal)
- **R = 301** (redirección permanente)
- **S=n** (salta las n siguientes reglas)
- **T=mime-type** (establece un mime type concreto)

Con todo lo anterior, una reescritura básica sería para toda URL que contenga una palabra, por ejemplo, un directorio info que sea redirecionado a otra URL, por ejemplo, evg.es:

```
RewriteEngine On
RewriteRule ^/info$ http://www.evg.es
```

Como se puede ver en la imagen, para poder utilizar cualquier reescritura hay que activar su tecnología mediante la directiva **RewriteEngine On**. Después de haber visto todo lo anterior es evidente pensar que se pueden realizar reglas más complicadas y útiles

Urls amigables

RewriteRule ^/daw/$ ciclo.php?opcion=1 [L]

HOST VIRTUALES (VIRTUAL HOST)

MÓDULOS

HTTPS y certificados digitales