Ramón Tamarit Agusti

FUNDAMENTOS DE INFORMÁTICA EN ENTORNOS BIOINFORMÁTICOS

PEC2 — Segunda Prueba de evaluación continua

DISEÑO DE UN SERVIDOR WEB PARA LA ANOTACION DE GENES EN UNA SECUENCIA GENOMICA

1 Introducción.

Preliminares

Si estoy escribiendo la introducción de este trabajo es por que considero que he llegado al objetivo que me había propuesto. Como profesional de la informática siempre me ha picado el gusanillo de hacer un sistema basado en web desde cero. No puedo negar que durante mi vida laboral he dirigido algún pequeño proyecto con esta orientación, pero nunca lo he hecho tecleando yo mismo el código, y siempre hemos usado alguna herramienta que te abstraía del HTML y de los scripts.

Cuando leí el enunciado de la PEC, me asombre un poco. No me parecía complicada, sino costosa. Costosa por varias razones:

a. Tengo y creo que tenemos (todos los alumnos del curso) poca experiencia en anotación de genes, y antes de programar algo hay que saber medianamente para que se hace y que espera el usuario de tu programa.

b. Personalmente nunca había programado en PHP ni en perl, más allá de lo que hemos visto en este curso.

También me asombro bastante la segunda pregunta. ¿Por qué? Muy sencillo, normalmente cuando se diseña una aplicación de base de datos, la parte de informes estadísticos se deja como última fase del desarrollo. Es decir, se hace un esbozo de lo que podría ser para vendérselo al cliente, pero hay que esperar a tener los "generadores de datos" en producción para poder programar funcionalidades. En este trabajo ha seguido esta lógica, me he olvidado del "punto 2" hasta que no he tenido más o menos en producción el "punto 1".

Objetivos y medios

Como programador encuentro que la única forma de aprender un lenguaje correctamente es practicando y construyendo aplicaciones. Primero una sencilla, y luego añadiendo funcionalidades más complicadas de programar. Por eso me he puesto como primer objetivo: Programar una aplicación funcional.

El siguiente objetivo es cumplir los plazos. Es decir tener un aplicación funcional y además lista como mucho el 30 de diciembre. Es un objetivo realista, así ocurre en la vida real, a los programadores nos encargan programas que hagan cosas útiles y nos piden que los entreguemos tal día. Normalmente al cliente no le importa si sabes php o nunca has instalado apache, y así será también en un laboratorio de bioinformática.

Como bien dice el enunciado de la PEC, podemos usar el lenguaje que queramos y como queramos. En la vida real también ocurre esto. Te encargan un programa y tienes que usar todos los medios y herramientas que necesites para cumplir con los objetivos (dentro del coste asociado). En el caso que me ocupa, por ejemplo he usado IDEs gráficos de MySQL, para acelerar el proceso de desarrollo, y también una aplicación que auto-genera código php a partir de tablas o consultas SQL. No soy masoquista programando, un programador tiene que usar todas las herramientas a su alcance para cumplir los objetivos. No obstante soy un newby en esto del desarrollo web, lo mío es el c, pascal, asm y otros sistemas, así que paciencia con los bugs.

La aplicación hasta hoy (no voy a añadir más código a esta beta más o menos estable) la puedes (podéis) encontrar rodando en uno de los servidores con los que trabajo en la dirección:

http://213.96.94.93/Inicio.html

Si deseáis ver el sistema bajo el que corre he dejado la página administrativa de xampp accesible:

http://213.96.94.93/xampp/

El password y usuario de MySQL es “admin” “admin”. Sirve tanto para consultar las tablas (muuuuuy sencillas) que he diseñado, como para acceder a las paginas protegidas del servidor de anotación. No obstante el servidor de MySQL lo he dejado sin protección y desde la pagina administrativa se puede acceder a el para ver las tablas, etc.

Los fuentes los he dejado en una carpeta (/home/rtamarit/anotagen/) de mi home del servidor de UOC. Ojo por que están tanto los ficheros ultima versión como los que he ido usando como pruebas.

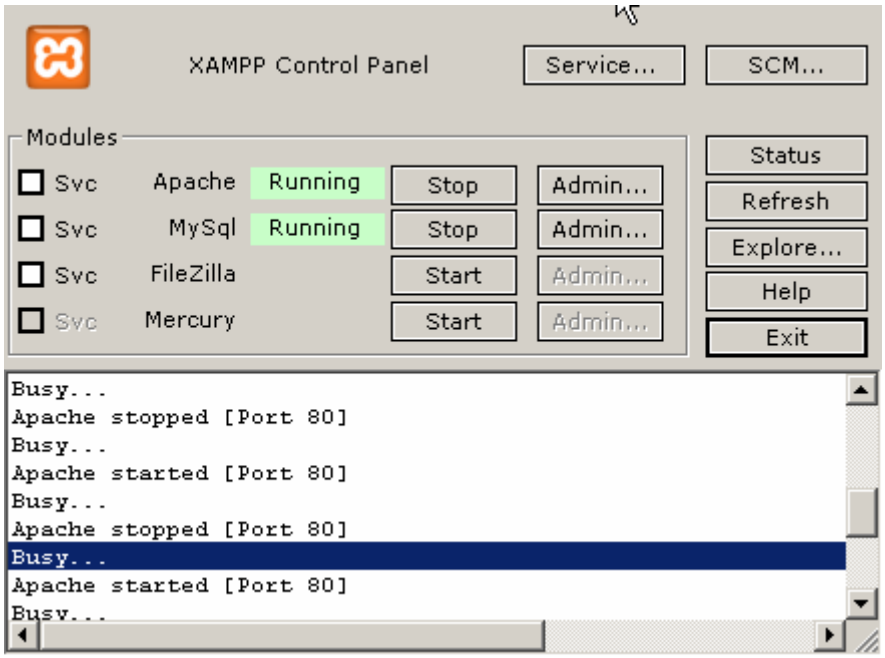
Si alguien del curso desea (si el profe lo considera) ver como ha quedado no hay ningún problema, dejaré el servidor rodando un par de semanas (me lo han dejado donde trabajo). Creo que educativamente puede estar muy bien, ver y entender, como es el proceso de desarrollo de una aplicación, sobre todo para el que este es su primer contacto con la programación.

Dicho esto…. Comienzo la explicación.

2 Aplicaciones y configuración del sistema

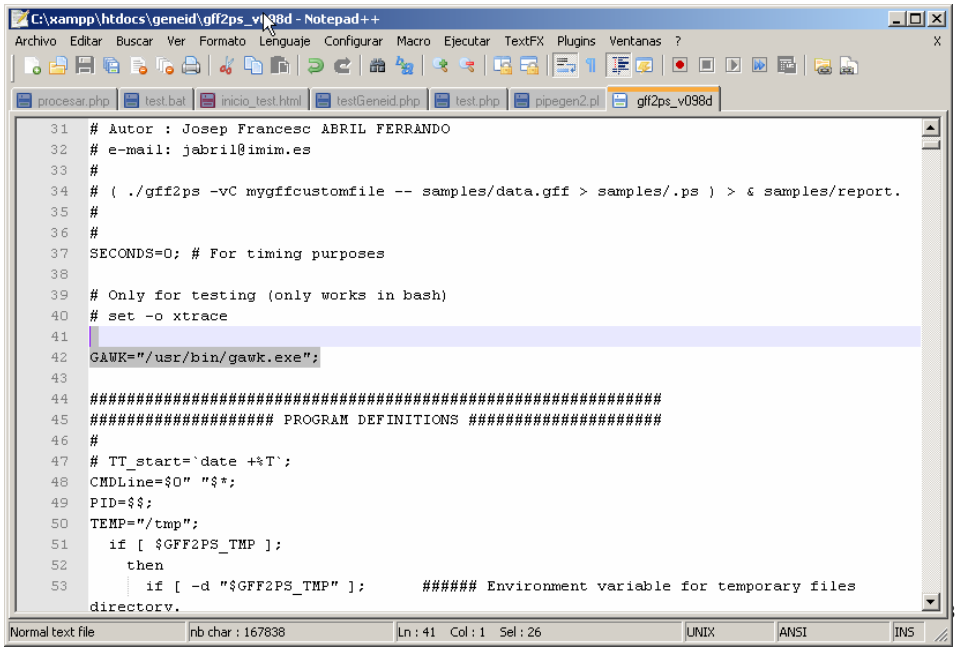
Servidor Apache, MySQL. Y cygwin.

Como servidor he usado la distribución XAMPP (disponible en http://www.apachefriends.org/en/xampp-windows.html), que instala perl, php, apache y mysql conjuntamente y sin problemas. Cygwin también hay que instalarlo para poder ejecutar correctamente los scripts.



Configuración de gff2ps.

Únicamente, asegurarse de tener una copia de gawk.exe (gawk para Windows o el distribuido con cygwin) (esta la carpeta \cygwin\bin\) en /usr/bin, o bien modificar la ruta completa.



Geneid

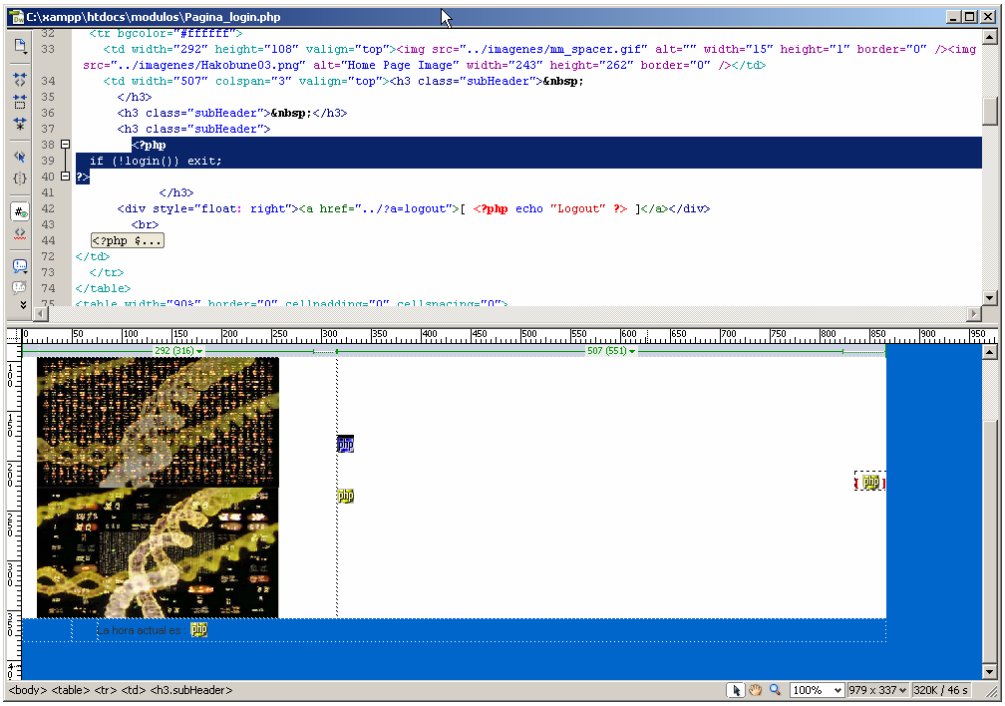
El programa geneid se copia en el raíz de la web. Usaré la versión compilada para Windows. (geneid.exe, que ya explique como la obtenía en la PEC1).

Deamweaver

Para diseñar la apariencia y navegabilidad del web he usado dramweaver en su versión CS3.

Desde la pagina de adobe se puede descargar una versión de prueba.

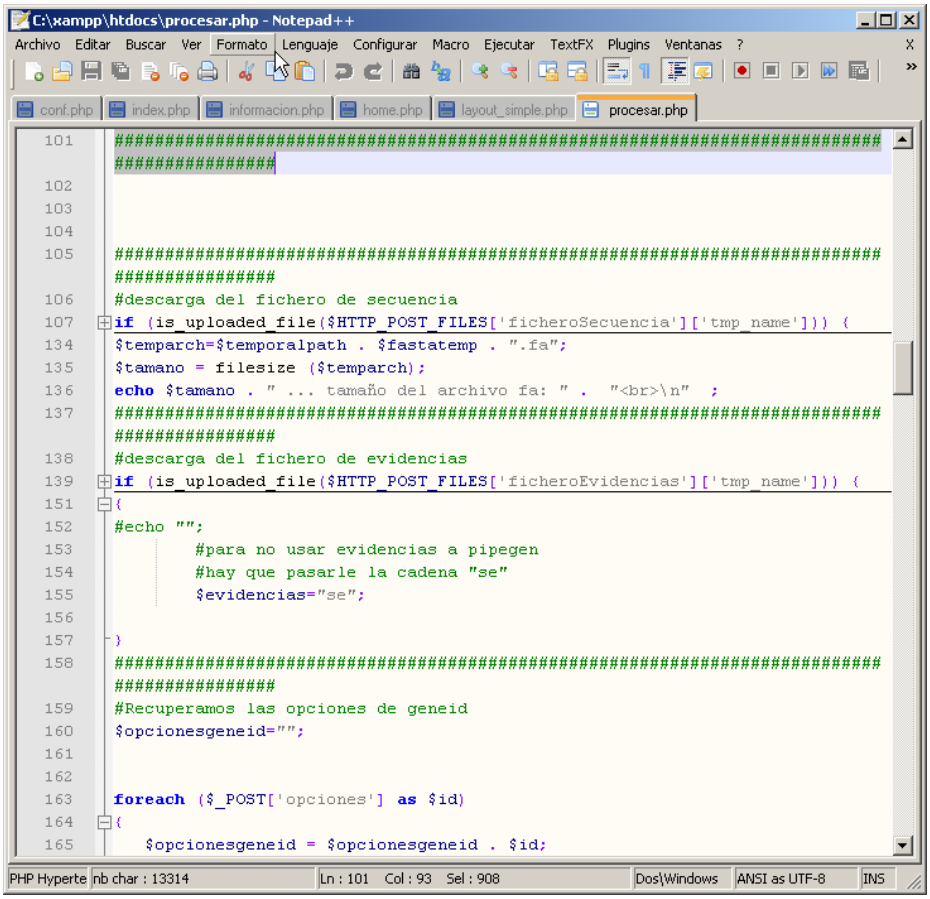
La ventaja de usar este IDE es clara, permite separar y maquetar visualmente las partes de scritp php del codigo HTML. Asi como copiar y pegar de una pagina a otra los scripts. Implementa funcionalidades de ocultación de código (subrutinas) y "entiende" perfectamente la sintaxis de php (ver el ejemplo).



Notepad++

Que decir de esta herramienta¡¡. Sin ella la aplicación no habría sido posible. Es un bloc de notas que interpreta la sintaxis de casi todos los lenguajes de programación. Permite de "todo".

La recomiendo a todo programador. Se puede descargar de:
http://sourceforge.net/projects/notepad-plus/



MySQL Maestro y Data Wizard for MySQL

Son dos muy buenas herramientas para manejar el MySQL desde Windows y generar código y consultas rapidamente.

Las versiones de prueba se pueden descargar de:

http://www.sqlmaestro.com/download/#f1

Ejercicio 1 – Protocolo de anotación.

Se desea diseñar una aplicación WEB en Perl o PHP para anotar regiones genómicas. Debéis definir un protocolo para anotar una región genómica con el programa geneid (accesible en http://genome.imim.es/software/geneid/index.html) y sus diferentes opciones. Se desea visualizar en la salida las predicciones en formato texto y una imagen realizada con el programa gff2ps posteriormente convertida a formato JPG: http://genome.imim.es/software/gfftools/GFF2PS.html.

El funcionamiento seria el siguiente:

1. El cliente desde su navegador envía una secuencia FASTA y selecciona algunas opciones del programa geneid (ver los genes, los exones (tipos), las señales (tipos))

2. El programa geneid recibe los valores de entrada (secuencia y opciones) y se ejecuta en una maquina remota

3. El programa gff2ps recibe las predicciones de los genes y crea un dibujo

4. Las predicciones en formato textual y la imagen generada son enviadas de vuelta al cliente en forma de pagina WEB para que las visualice

3 Definición del proyecto.

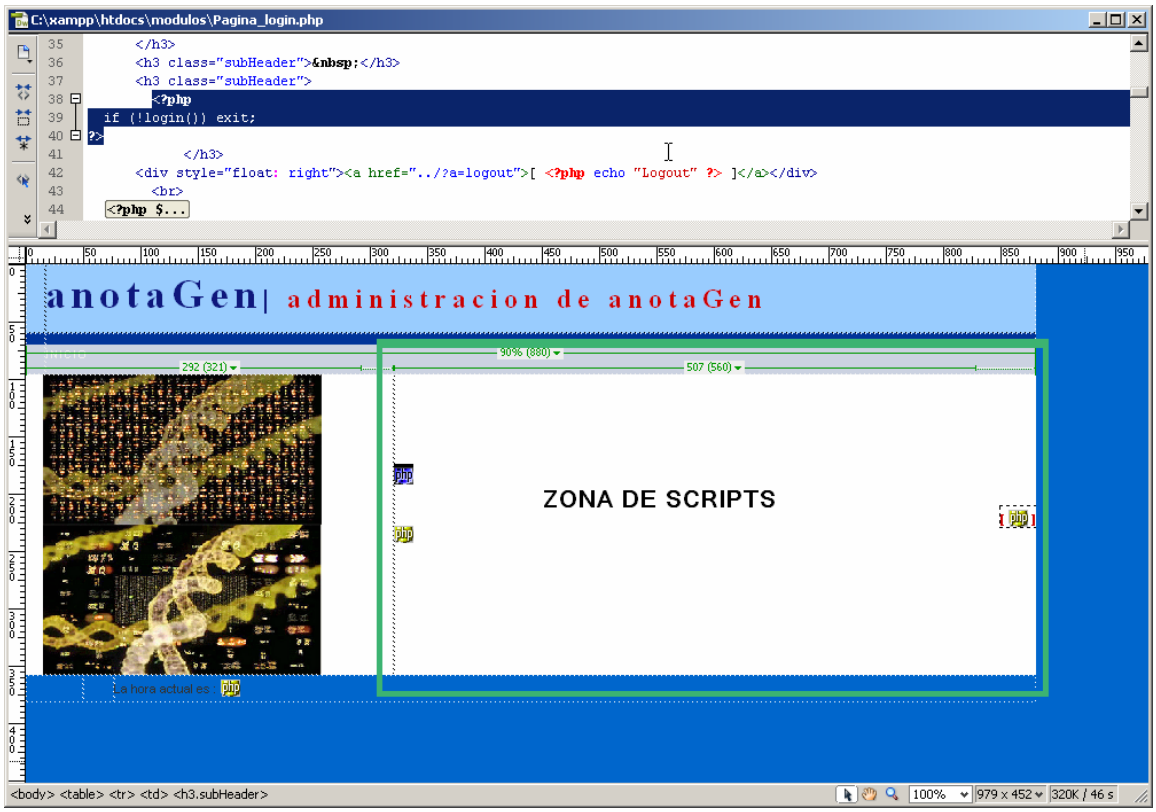
Se trata de construir una pequeña aplicación web con las siguientes peculiaridades:

- Se instalará en un sistema operativo Windows XP, pero deberá ser fácilmente portable a un sistema unix o Linux.
- Trabajará en un servidor Apache con los módulos de php y perl instalados (en Windows se supone).
- Será modular: tendrá una página inicial que desviará al usuario a las diferentes partes de la aplicación. Las partes de la aplicación entrarán como módulos de forma que se puedan añadir nuevos o modificar los existentes manteniendo el formato de la misma.
- La interface de la aplicación se programará en php. La parte de la aplicación que realice los cálculos con geneid y gff2ps se programará en perl. Los scripts de perl serán llamados desde el código php. Uno de los módulos de php se encargará de presentar los resultados al usuario.
- Se dejará preparada la aplicación para implementar el ejercicio 2 del PEC.

4 Hoja de plantilla.

Antes que nada explicar que primero he construido y probado todos los scripts sin formato HTML, y después he elegido un diseño más o menos manejable y he montado todos los vínculos de la web.

Para ello me he construido una plantilla HTML en donde he ido reservando las zonas en donde incluir los scripts. Esto se aprecia muy bien desde Dreamweaver. Veamos una captura de la página de login para entender el proceso:



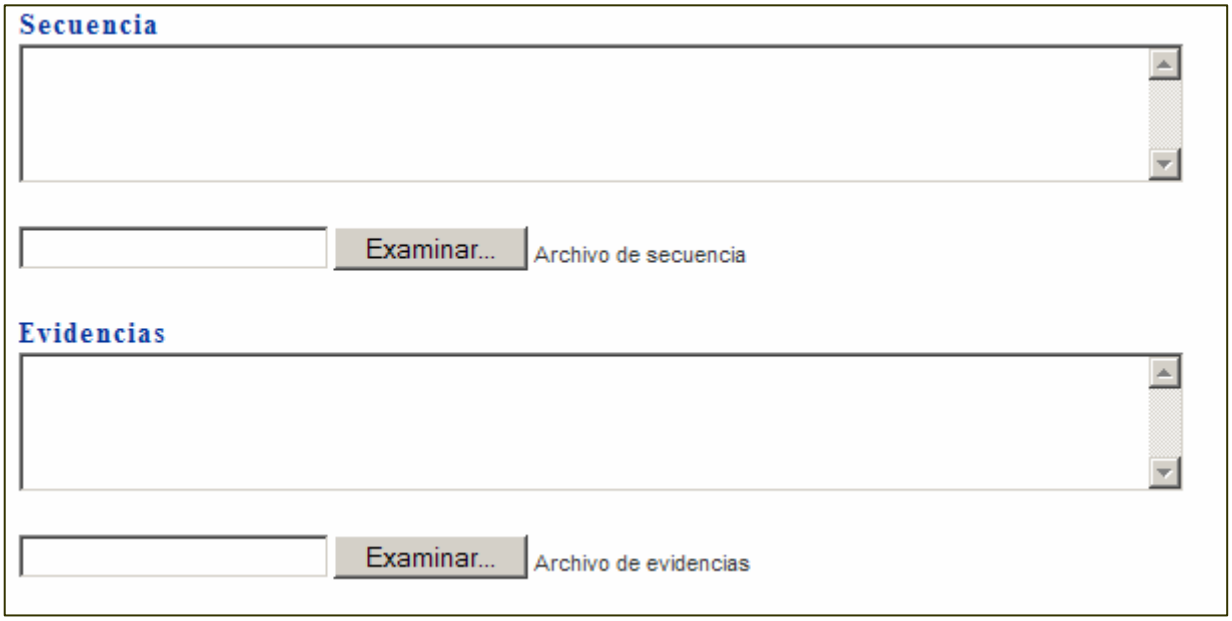
En el recuadro verde esta la zona de scripts. En la imagen he seleccionado una de las marcas de scritp y en la parte superior aparece resaltado el código php correspondiente. De esta forma una vez programado los trozos de código, los he ido insertando en la plantilla. Las subrutinas están normalmente al final de cada página.

5 Preparación del formulario de entrada de datos.

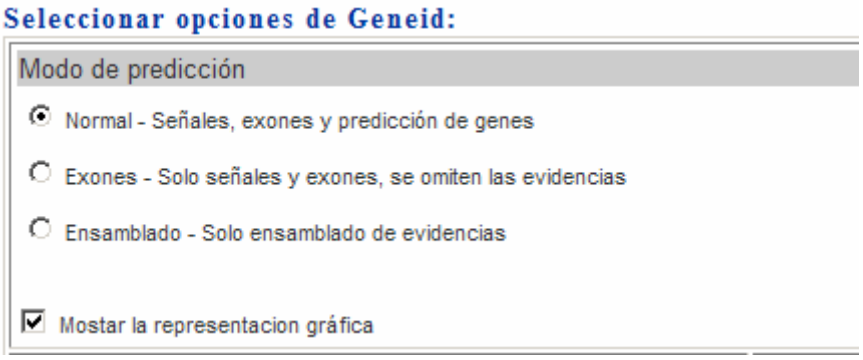
El formulario de entrada de datos (anotar.html) es un simple documento HTML (con el formato de la plantilla) con los correspondientes campos para introducir:

1. La secuencia en formato fasta, o bien subir al servidor un fichero desde la maquina cliente.

2. El fichero gff de evidencias, también se puede pegar en un cuadro de texto, o bien seleccionar el archivo y subirlo al servidor

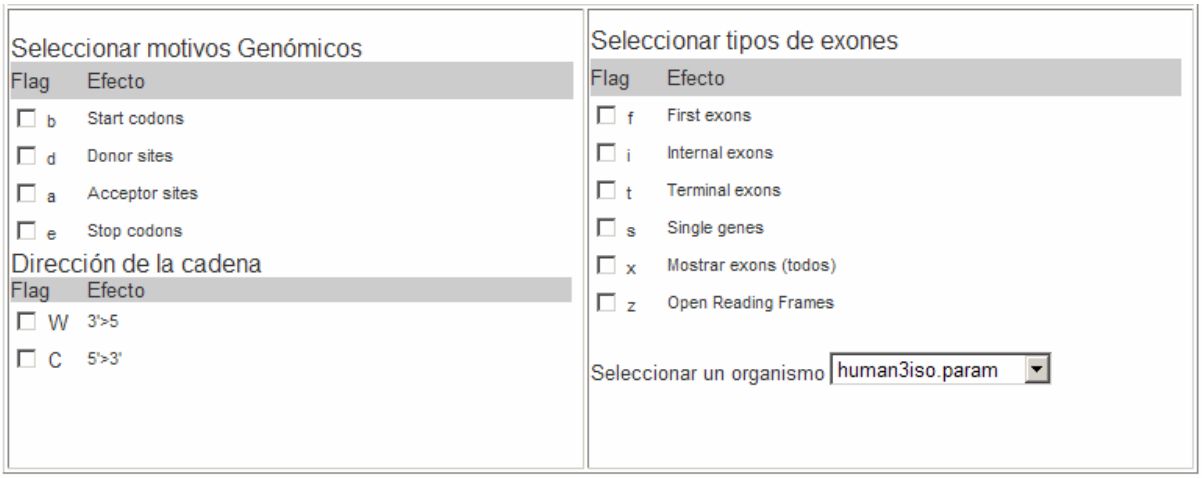


3. El modo de cálculo de geneid. Se seleccionan con tres botones de selección excluyentes.



4. Si se desea obtener la representación gráfica por gff2ps hay que marcar la casilla de verificación

5. Los motivos geonómicos y tipos de exones a mostrar se seleccionan desde las casillas de verificación, se pueden marcar todas las que se quieran (geneid lo permite).



6. Se puede seleccionar de un desplegable el fichero de parámetros que usará geneid.

7. Por defecto geneid si no le decimos nada buscará en las dos direcciones de la secuencia de ADN. Si únicamente deseamos buscar genes en una de las direcciones hay que marcar la casilla correspondiente.

Por lo demás, este documento no tiene nada de especial. El form se envía a un script PHP (procesar.php) mediante POST, y allí se procesaran los datos.

6 Procesado de los datos

Los datos llegan al script procesar.php por medio del metodo post. Veamos las partes interesantes del script.

- Uso un archivo de configuración para guardar las rutas del servidor y otras variables, se carga en la linea 3 con el include.

- Las variables $inicio y $fin calculan el tiempo de procesado. Se usa la función microtime que nos da el tiempo en un float con precision decimal de milisegundos.

```
1.  <?php
2.  ##ARCHIVO DE CONFIGURACION
3.  include('conf.php');
4.
5.
6.  #Inicio del calculo
7.  $inicio=microtime(get_as_float);
8.
```

- La IP del cliente la obtenemos con el trozo de código desde la línea 11 hasta la línea 30. Uso este método por que me di cuenta que desde mi casa no se obtenia bien la IP por navegar a través de Proxy.

```
9.  ###obtener la ip del usuario#################################
10. ## Incluso si se esta detras de un proxi###################
11. if ($_SERVER) {
12. if ( $_SERVER[HTTP_X_FORWARDED_FOR] ) {
13. $realip = $_SERVER["HTTP_X_FORWARDED_FOR"];
14. } elseif ( $_SERVER["HTTP_CLIENT_IP"] ) {
15. $realip = $_SERVER["HTTP_CLIENT_IP"];
16. } else {
17. $realip = $_SERVER["REMOTE_ADDR"];
18. }
19. } else {
20. if ( getenv( 'HTTP_X_FORWARDED_FOR' ) ) {
21. $realip = getenv( 'HTTP_X_FORWARDED_FOR' );
22. } elseif ( getenv( 'HTTP_CLIENT_IP' ) ) {
23. $realip = getenv( 'HTTP_CLIENT_IP' );
24. } else {
25. $realip = getenv( 'REMOTE_ADDR' );
26. }
27. }
28.
29. $usrIP=$realip;
```

- En el trozo siguiente “calculo” el identificador que le asignaré al cálculo. Este me servirá posteriormente para

    a. Crear un directorio particular para el calculo y almacenar los resultados,

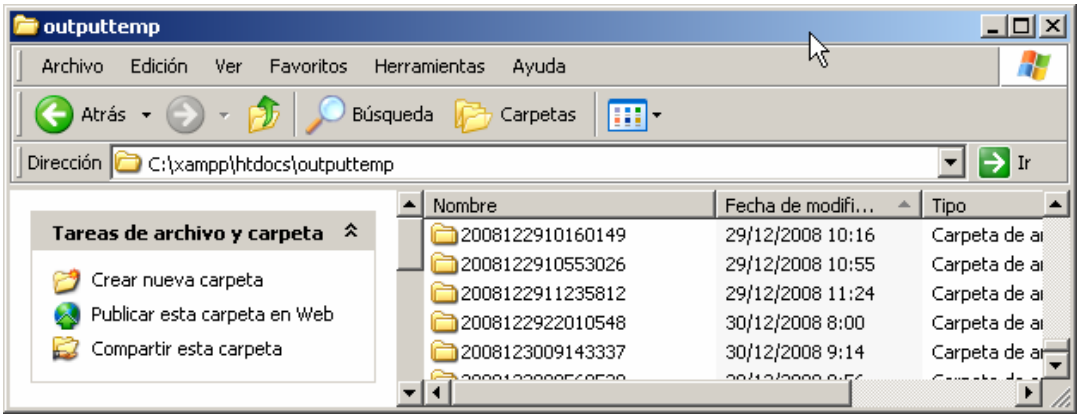
    b. nombrar los archivos intermedios con este identificador,

    c. será el índice de la tabla de cálculos (Pregunta 2)

  El identificador se calcula uniendo el año, mes, día, hora, minutos, segundos, y un número aleatorio entre 100 y 999. De esta forma me aseguro de que será un índice único.

```
30. #le añadimos un numero aleatorio. (por si dos usuarios piden un claculo a la vez)
31. #$outputtemppath se define en conf.php
32. $fastatemp= date("YmdHis"). rand(100,999);
33. $temporalpath = $outputtemppath . $fastatemp . "/";
34. $temporalpathimeges= "outputtemp/" . $fastatemp . "/";
35.
36. while(list($key, $Value) = each($_POST)){
37. echo $key ." : ". $Value . "<br>";
38. }
```

- Una vez tengo el identificador creo el directorio temporal. Los directorios se crean dentro de raizdelsitio/outputtem/

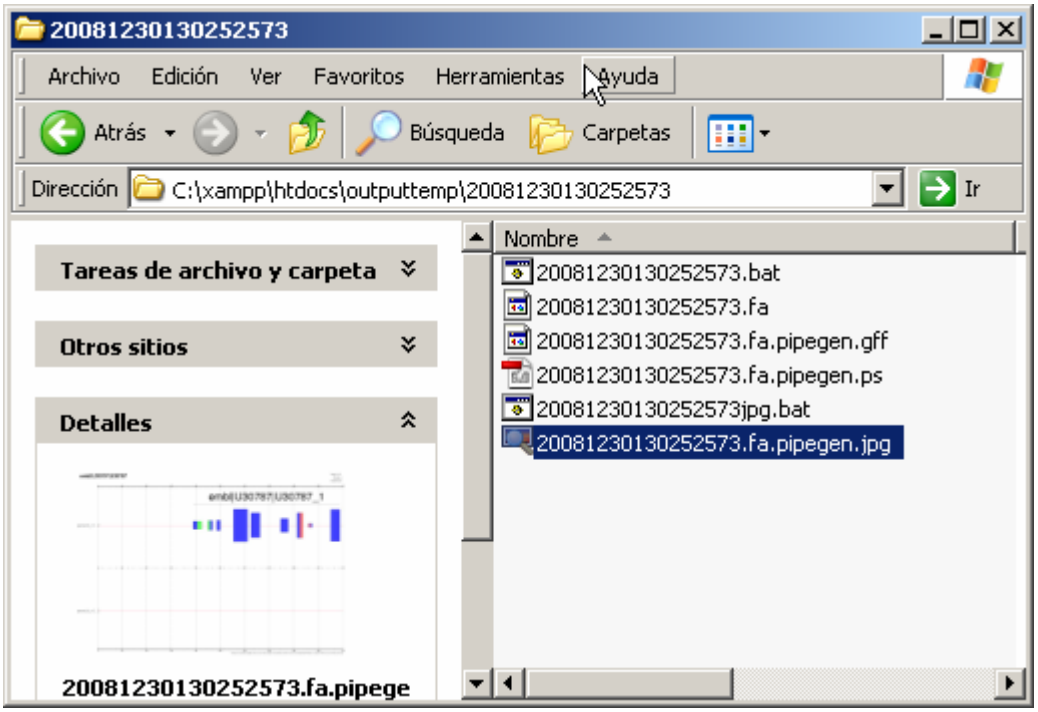



```
39.
40.
41. ####
42. #Creamos el directorio temporal
43. mkdir($temporalpath);
44.
45. #nos pasamos al directorio de base ya que por defecto estaremos en /modulares/
46. chdir ("/xampp/htdocs/");
47.  echo "Temporal :" . $temporalpath . "<br>\n";
48.  echo "Su IP    :" . $usrIP . "<br>\n";
49. #echo $_FILES['ficheroSecuencia']['error']. "<br>\n";
50.  echo $_FILES['ficheroSecuencia']['name']. "<br>\n";
51.  echo $_FILES['ficheroSecuencia']['size']. "<br>\n";
```

```
#echo $_FILES['ficheroSecuencia']['tmp_name']. "<br>\n";
#echo $_FILES['ficheroSecuencia']['type']. "<br>\n";
##########################################################################################
####
#si quisieramos poner un filtro para el tamaño de la secuencia etc aqui seria un buen
sitio.

##########################################################################################
####
```

- Seguidamente capturamos el fichero de la secuencia y el fichero de evidencias si procede. Y se guardan en su directorio particular. En la imagen vemos como se han nombrado los ficheros generados. Para informar de que el usuario no ha enviado evidencias uso la variable $evidencias. Si no llegan evidencias vale “se”, y si llegan vale el nombre del fichero.




```


##########################################################################################
####
#descarga del fichero de secuencia
if (is_uploaded_file($HTTP_POST_FILES['ficheroSecuencia']['tmp_name'])) {
        $FicheroUploadSecuencia = $fastatemp . ".fa";

    #echo $FicheroUploadSecuencia. "<br>\n";
    #copiamos el fichero al directorio temporal
        if (move_uploaded_file($HTTP_POST_FILES['ficheroSecuencia']['tmp_name'],
$temporalpath . $FicheroUploadSecuencia))
        {
        #echo "OK";
        };
    #echo "El archivo ha sido cargado correctamente.";

     #echo "Fichero de secuencia descargado";
} else {
        #echo "Fichero pasado por casilla de texto";
```

```
76.         #recuperamos la secuencia fasta del cuado de texto
77.     #echo "Variable \$secuencia: " . $HTTP_POST_VARS["secuencia"] . "<br>\n";
78.         #la guardamos en el fichero temporal
79.         $archivofa = fopen($temporalpath . $fastatemp . ".fa" , "w");
80.         if ($archivofa)
81.         {
82.                 fputs ($archivofa, $HTTP_POST_VARS["secuencia"]);
83.         }
84.
85.
86.         fclose ($archivofa);
87.
88. }
89. $temparch=$temporalpath . $fastatemp . ".fa";
90. $tamano = filesize ($temparch);
91. echo $tamano . " ... tamaño del archivo fa: " .  "<br>\n"  ;
92. ##########################################################################################
    ####
93. #descarga del fichero de evidencias
94. if (is_uploaded_file($HTTP_POST_FILES['ficheroEvidencias']['tmp_name'])) {
95.         $FicheroUploadEvidencias = $fastatemp . ".ev.fa";
96.         #copiamos el fichero al directorio temporal
97.
98.         if (move_uploaded_file($HTTP_POST_FILES['ficheroEvidencias']['tmp_name'],
    $temporalpath . $FicheroUploadEvidencias))
99.         {
100.            #echo "OK";
101.                    $evidencias=$temporalpath . $FicheroUploadEvidencias;
102.            };
103.
104.        #echo "Fichero de evidencias descargado";
105.    } else
106.    {
107.    #echo "";
108.            #para no usar evidencias a pipegen
109.            #hay que pasarle la cadena "se"
110.                    $evidencias="se";
111.
112.    }
```

- Ahora capturamos las opciones de geneid. Cada botón de selección de la ficha anotar.html tiene asignado su correspondiente argumento para geneid, así que no tenemos más que unir los que se hayan marcado. Se guarden en la variable $opcionesgeneid

```
113.    ##########################################################################################
    ########
114.    #Recuperamos las opciones de geneid
115.    $opcionesgeneid="";
116.
117.
118.    foreach ($_POST['opciones'] as $id)
119.    {
120.       $opcionesgeneid = $opcionesgeneid . $id;
121.       echo "Variable \$id: " . $id . "<br>";
122.    }
123.    echo "Variable \$opciones: " . $opcionesgeneid . "<br>";
124.    $opcionesgeneid = $opcionesgeneid . "P";
125.    ##########################################################################################
    ########
```

- Para capturar el modo de cálculo uso una estructura switch con tres casos:

    1. normal: Es el modo de ejecución por defecto de geneid. El motor de búsqueda de genes los buscará y si le podemos dar evidencias o no.

    2. exones: Se fuerza a no usar las evidencias aunque el usuario las haya enviado.

    3. ensamblado: Se desactiva el motor de búsqueda (con –o) y se fuerza a que el usuario envié las evidencias.

```
126.    #Inclusion del modo de calculo
127.    $mododecalculo=$_POST['Modo'];
128.    switch($mododecalculo){
129.            case "normal";
130.        ## Nada ejecutamos normalemente geneid
131.                            break;
132.            case "exones";
133.        ## Nada ejecutamos normalemente geneid pero sin
134.        ## archivo de evidencias
135.            $evidencias="se";
136.            #Añadimos el flag O para desactivar la busqueda de genes
137.            $opcionesgeneid . "o";
138.                            break;
139.            case "ensamblado";
140.        ## Nada ejecutamos normalemente geneid pero sin
141.        ## archivo de evidencias
142.            if ($evidencias=="se"){
143.                    echo "<br>\n" . "ERROR: Faltan las evidencias" . "<br>\n";
144.                    exit;
145.            }
146.        #Añadimos el flag O para desactivar la busqueda de genes
147.            $opcionesgeneid . "o";
148.                            break;
149.            default:
150.        ##equivalente a normal
151.
152.                            break;
153.    }
154.    ##########################################################################################
    ########
```

- Capturamos si el usuario desea visualizar el resultado gráfico, lo almacenamos en $mostrarimagen, que puede valer “mostrar” o “nomostrar”

```
155.
156.    #Mostar la imagen
157.    #$mostarimagen=$_POST['grafico'];
158.    if ($_POST['grafico']=="mostrar") {
159.    $mostrarimagen = "mostrar";
160.    #echo "mostrar imegen  = " . $mostrarimagen;
161.    } else
162.    {
163.    $mostrarimagen = "nomostrar";
164.    #echo "mostrar imegen  = " . $mostrarimagen;
165.    }
166.
167.
168.
```

- Recuperamos el fichero de parámetros que desea el usuario. Los posibles están ya parametrizados en el form, así que únicamente nos falta añadirle la ruta. Por motivos de seguridad, no se debe añadir la ruta (y tampoco se debería añadir el nombre del archivo)

```
169.    #recuperamos el fichero de parametros de geneid
170.    $paramgeneid= $_POST['organismo'];
171.
```

- Ahora ya tenemos todos los datos necesarios y hay que enviarlos a un script en perl llamado pipegen2.pl. Este scritp en realidad no es necesario ya que lo mismo lo podríamos hacer desde php, pero como ya lo tenía hecho de la PEC1 he considerado que estaría bien usarlo. A pipegen2 hay que facilitarle los siguientes datos (por orden):

    1. Las opciones de geneid: Sin el "-" ni el parámetro G, ya que por defecto lo incluyo en el script. (Se supone que lo que queremos es una salida para gff2ps)
    2. La ruta de trabajo: En donde tenemos guardado el fichero fasta, y en donde se guardarán los resultados, el fichero gff y el postcript de salida de gff2ps
    3. El archivo fasta a calcular
    4. El fichero de parámetros de geneid que queremos usar
    5. El fichero de evidencias de geneid, si no tenemos hay que pasar la cadena "se"
    6. Una cadena diciendo si queremos que calcule la imagen ("mostrar") o no ("nomostrar").

El resultado se guardará en el directorio de trabajo que le hemos pasado por parámetro. Obtendremos dos archivos: XXXXXXXX.fa.pipegen.gff y XXXXXXXX.fa.pipegen.ps

- El comando se monta en la línea 176. Pero para poder ejecutar correctamente pipegen2 desde Windows debemos invocarlo desde el interprete de comando bash de cygwin….. comienzan los problemas…..

- Ya que estamos trabajando en Windows cuando usamos la función exec llamaremos al interprete de Windows (cmd.exe), así que tenemos que llamar primero a bash y pasarle la cadena completa del comando, esto se hace en la línea 186.

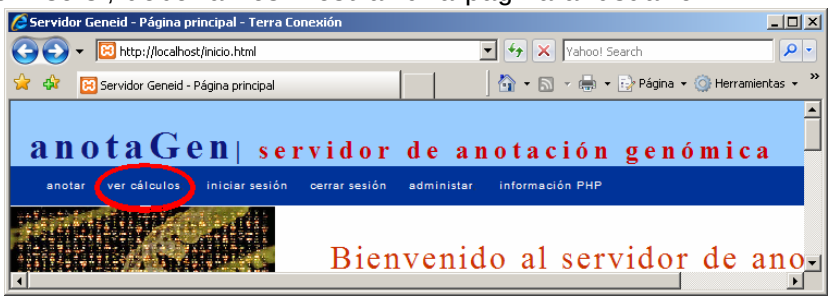
- Bien podríamos ejecutar directamente el comando con exec, pero prefiero guardarlo en un .bat (script de cmd) y ejecutar el bat con exec. De esta forma, podemos ver exactamente lo que esta pasando y rearrancar el script si es necesario a mano (y también hacer todas las pruebas que consideremos necesarias).

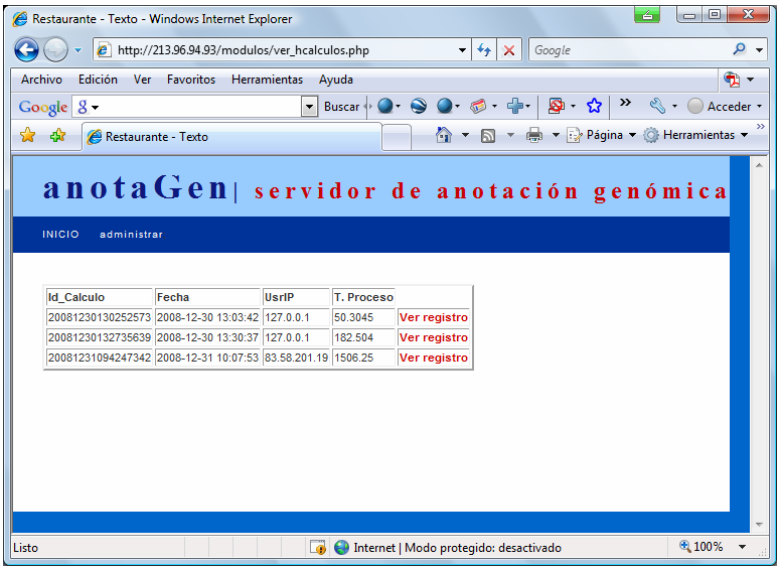
```
172.    ##########################################################################################
    ########
173.    #enviamos el imput a pipegen2.pl
174.    #muy importante..... los guardamos en un bat sino desde el windows no funciona
175.    #
176.    $comandoint =           $rutageneid . "pipegen2.pl " .
177.                            $opcionesgeneid . " " . $rutabasedisco .
```

```
178.                            $temporalpath . " /cygdrive/c" .
179.                            $temporalpath . $fastatemp . ".fa" .
180.                        " ". $rutageneid . "param/" .
181.                            $paramgeneid . " " .
182.                            $evidencias . " " .
183.                            $mostrarimagen;
184.
185.
186.    $comando = "c:\\cygwin\\bin\\bash --login -i -c " . '"' .  $comandoint . '"';
187.
188.    #echo $comando;
189.
190.    $archivobat = fopen($temporalpath . $fastatemp . ".bat" , "w");
191.
192.    if ($archivobat) {
193.            fputs ($archivobat,$comando);
194.    }
195.
196.    fclose ($archivobat);
197.    echo time();
198.
199.    $comandoint = " ". $rutabasedisco . $temporalpath . $fastatemp . ".bat";
200.    $comando = "c:\\cygwin\\bin\\bash --login -i -c " . '"' .  $comandoint . '"';
201.
```

- Siguiente problema. El sistema va muy lento, la llamada al sistema desde apache a bash le asigna muy pocos recursos. Hay un método de asignar más recursos, pero no lo voy a usar, ya que el PC que voy a usar como servidor es en el que normalmente trabajo, al fin y al cabo esto es un sistema de pruebas.

- Para solventar este problema, esta la siguiente solución que no he implementado, pero si probado. Podemos ejecutar el script bat desde un objeto com del sistema (ya lo he usado en otras aplicaciones). Esta forma de trabajar aparte de asignarle muchos más recursos, devuelve el control de la ejecución a php y el script queda rodando en segundo plano. Eso si, deberíamos mostrar una página al usuario indicándole que su proceso esta en curso, y además ofrecerle un link para que pueda acceder al resultado final. Esta parte si que esta implementada, desde el menú cálculos se puede acceder a los resultados de todos los cálculos realizados en el sistema:





- Otro problema. Dado que la ejecución tarda bastante se dispara el límite de ejecución de los scripts. Esto se puede controlar de varias formas, una modificando el php.ini (php.conf en Linux), y otra incluyendo el set_time_limit(600), que resetea el tiempo limite y establece el contador a 600 segundos (línea 204). En la última versión lo he aumentado a 1600.

```
202.    ## ponemos el limite de ejecución a 10 mn y ademas lo resetea ... sino se
203.    ## puede quedar colgado el script y se ensucia la memoria del sistema
204.    set_time_limit(600) ;
205.    #echo $comando;
206.    exec($comando ,$output);
207.    #print_r ($output);
208.    echo time();
209.
210.    ##############################################################################################
211.    #Tb se puede ejecutar asi, pero en segundo plano. Puede servir para calculos
212.    # largos, evitando la limitación de tiempo de script de apache. Al usuario habria
213.    # que hacerle llegar el link con el resultado.
214.    #$WshShell = new COM("WScript.Shell");
215.    #$OExec = $WshShell -> run($comando);
216.    ##############################################################################################
217.
```

- Para convertir el postcript a jpg uso la misma técnica que antes: genero un bat y luego lo ejecuto con exec.

```
218.    ##############################################################################################
    ########
219.    #aqui convierto el ps a jpg
220.    $comandoint = " /cygdrive/c/cygwin/bin/convert.exe -rotate 90  /cygdrive/c" .
    $temporalpath . $fastatemp . ".fa.pipegen.ps "  .  "/cygdrive/c" . $temporalpath .
    $fastatemp . ".fa.pipegen.jpg ";
221.    $comando = "c:\\cygwin\\bin\\bash --login -i -c " . '"' .  $comandoint . '"';
222.    #echo $comandoint;
223.    #echo $comando;
224.
225.
226.    $archivobat = fopen($temporalpath . $fastatemp . "jpg.bat" , "w");
227.
228.    if ($archivobat) {
229.            fputs ($archivobat,$comando);
230.    }
231.
232.    fclose ($archivobat);
233.    echo time();
234.
235.    $comandoint = " ". $rutabasedisco . $temporalpath . $fastatemp . "jpg.bat";
236.    $comando = "c:\\cygwin\\bin\\bash --login -i -c " . '"' .  $comandoint . '"';
237.
238.    #echo $comando;
239.    exec($comando ,$output);
240.    #print_r ($output);
241.    echo time();
242.
```

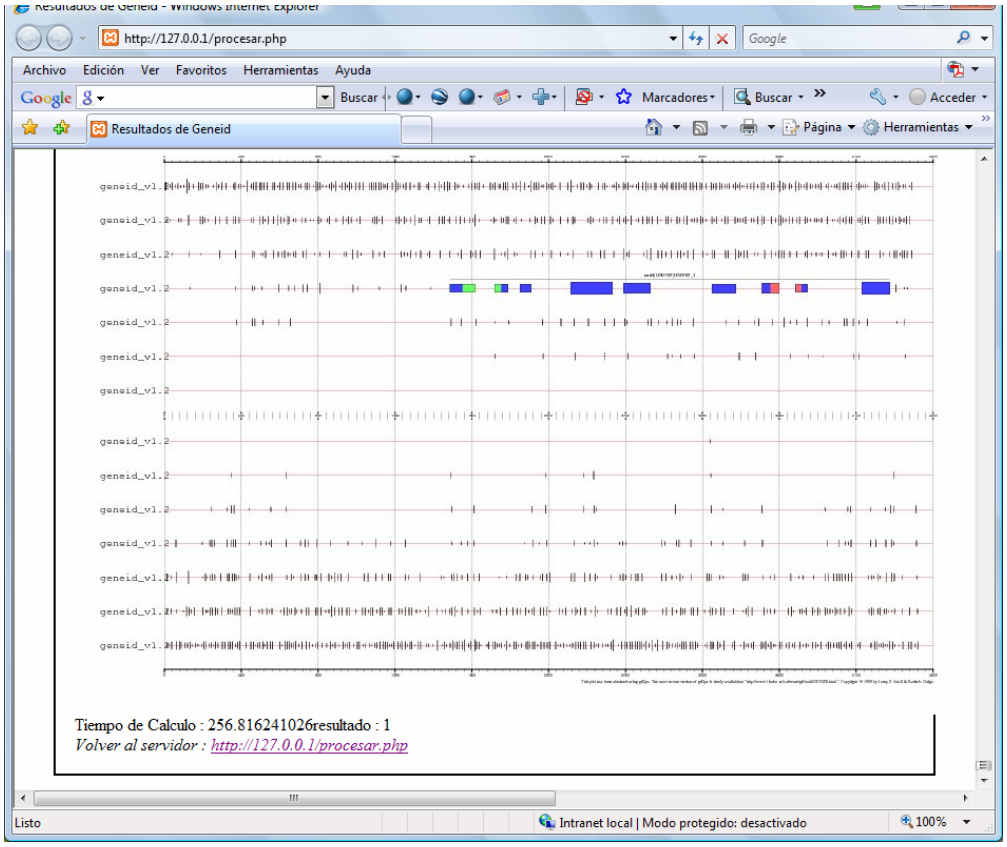
- Ahora deberíamos comprobar si se han generado los ficheros antes de intentar mostarlos, pero esta parte no la he implementado.

- Las líneas siguientes se encargan de mostrar el resultado del fichero gff y del jpg en la página del usuario. Dado que el script php esta embebido en la plantilla no es necesario preocuparse de los formatos.

```
243.    ##########################################################################################
    ########
244.    #mostrar el resultado de geneid
245.    $resultados= $temporalpath . $fastatemp . ".fa.pipegen.gff ";
246.    $fp=fopen($resultados,'r');
247.    $contenido=fread($fp,filesize($resultados));
248.    fclose($fp);
249.
250.    $contenido = "<p>".$contenido."</p>";
251.    ## reemplazamos los saltos de linea...
252.    $contenido = str_replace(chr(10),"</p><p>",$contenido);
253.    echo $contenido;
254.    ##########################################################################################
    ########
255.    #Esta forma de mostrar la imagen da problemas, la cambio por la de mas abajo
256.    #$imagen = imagecreatefromjpeg( $temporalpath . $fastatemp . ".fa.pipegen.jpg");
257.    ## Mostrar la  imagen
258.    #imagejpeg($imagen);
259.
260.    ##Si el usuario marca mostar el grafico
261.    if ($mostrarimagen=="mostrar") {
262.    #Mostar la imagen.
263.    echo    "<br>\n";
264.    echo '<img src="' .  $temporalpathimeges . $fastatemp . '.fa.pipegen.jpg" alt=""
    name="imagen" width="842" height="595" id="imagen" />' ;
265.    echo    "<br>\n";
266.    }
```

La parte que muestra la imagen a veces funciona y a veces no… y no se por que¡¡¡¡.

La imagen está debajo de los resultados en formato gff.



Esta es una muestra con el ejemplo de la PEC1, ¡¡que chula!!

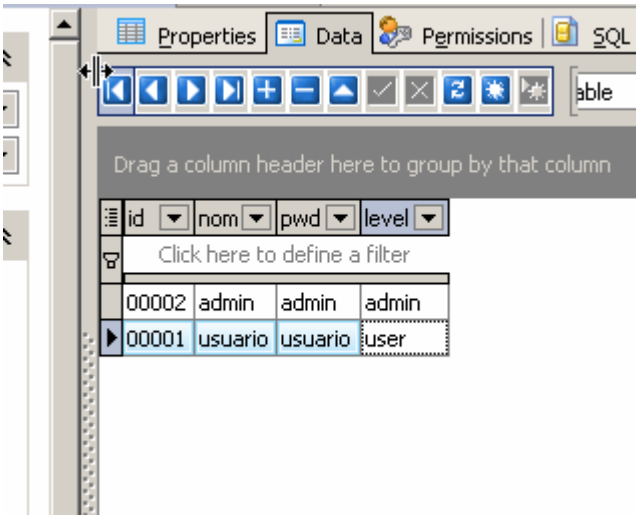
Las siguientes líneas de procesar.php se encargan de recopilar la información y guardarla en una base de datos MySQL. El código de estas líneas se explica en el apartado 8, ya en la pregunta 2.

7 Consideraciones finales de la pregunta 1.

Como hemos visto, hasta aquí he llegado, la falta de tiempo no me ha permitido implementar mas funcionalidades. Las funcionalidades que se han quedado en el tintero son básicamente dos:

Gestión de usuarios

Aunque el usuario puede hacer login desde la opción habilitada en el menú de la página principal, no he implementado que se guarde su id_usuario al realizar un cálculo. En esta versión operativa se guarda la id_usuario= '00001' (ver línea 288).



La idea principal es que cada usuario pudiera guardar y posteriormente visualizar sus propios cálculos. La página que muestra los cálculos ahora mismo muestra los de todos los usuarios. En principio implementar esta funcionalidad no costaría mucho ya que he dejado el sistema preparado para ello.

Como veremos en el ejercicio 2, en algunas partes de la aplicación administrativa si que se reconoce la sesión de usuario y pide el uid passw.

Gestión de imágenes

Tenía pensados dos módulos:

Modulo 1

1. Posibilidad para que el usuario pueda seleccionar entre varios ficheros de parámetros de configuración desde el formulario de entrada, o bien el se lo proporcionara con upload.

2. Igualmente que se ofrezca esta opción desde la gestión de cálculos del usuario.

Modulo 2

Construir un formulario en el que se puedan seleccionar distintas opciones para configurar un fichero de parámetros de gff2ps.

Funcionaría a modo de bloques, más o menos con todas las opciones especificadas en el manual de gff2ps.

Ejercicio 2 – Administración de los resultados.

Los desarrolladores del servidor desean también registrar el número de personas que están utilizando este servicio WEB para pedir un aumento de sueldo. Para ello, debe crearse una base de datos en la que se lleve el registro de los trabajos realizados por la aplicación de anotación de genoma en el servidor. Deben registrarse los siguientes valores para cada ejecución:

- longitud de la secuencia
- tamaño del fichero
- numero de genes en la predicción
- numero de exones en la predicción
- opciones utilizadas
- tiempo de ejecución
- maquina del cliente
- etc...

Esta base de datos será consultada a través de la WEB por el administrador del servidor. Imaginad que sois vosotros mismos, y comentad la estructura de las tablas MySQL que necesitáis: mencionad ejemplos de interrogaciones a la base de datos que pensáis que serian mas comunes para el objetivo que deseamos, así como la estructura de la pagina WEB

8 Almacenamiento de los datos

Lo primero es guardar los datos del cálculo.

- Se guardarán en una BD que he llamado “anotagen”.
- La tabla hcalculos_rta contendrá los datos del cálculo. La estructura de esta tabla es la siguiente:

```
-- Table: hcalculos_rta

-- DROP TABLE IF EXISTS `hcalculos_rta`;

CREATE TABLE `hcalculos_rta` (
  `id_hcalculo`    char(20) NOT NULL,
  `id_usuario`     char(20) NOT NULL,
  `fechacalculo`   timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
  `usrIP`          varchar(12) NOT NULL,
  `geneidOps`      varchar(10) NOT NULL,
  `modo`           varchar(10) NOT NULL,
  `grafico`        varchar(10) NOT NULL,
  `TiempoProceso`  float NOT NULL,
  `size`           int(20),
  /* Keys */
  PRIMARY KEY (`id_hcalculo`)
) ENGINE = MyISAM;
```

La denominación de los campos es explicita (se ve claramente que guarda cada uno). El campo id_hcalculo será la clave primaria como ya comente en el apartado anterior.

La fecha del cálculo se guarda automáticamente con current_timestamp.

El campo id_usuario esta definido para poder guardar el usuario que ha realizado el cálculo, aunque esta funcionalidad no la he implementado. Este campo mantiene integridad referencial con su homologo de la tabla usuarios:

```
-- Table: usuarios_rta

-- DROP TABLE IF EXISTS `usuarios_rta`;

CREATE TABLE `usuarios_rta` (
  `id`      char(20) NOT NULL,
  `nombre`  varchar(10) NOT NULL,
  `pwd`     varchar(10) NOT NULL,
  `level`   char(20) DEFAULT 'user',
  /* Keys */
  PRIMARY KEY (`id`)
) ENGINE = MyISAM
  COMMENT = 'Tabla de usuarios';
```

```
267.    ##########################################################################################
    ########
268.    #Fin del calculo
269.    $fin=microtime(get_as_float);
270.    $TiempoProceso=$fin-$inicio;
271.    echo "Tiempo de Calculo : " . $TiempoProceso;
272.    ##########################################################################################
    ########
273.    #Insertamos el registro en MySQL
274.    $bd= mysql_connect("localhost", "admin", "admin");
275.    mysql_select_db("anotagen", $bd);
276.
277.    $sql  = "INSERT INTO hcalculos_rta ";
278.
279.    $sql .= " (id_hcalculo,id_usuario,usrIP,geneidOps,modo,grafico,TiempoProceso,size)
    ";
280.
281.    $sql .= " VALUES ('$fastatemp', '00001','$usrIP','$opcionesgeneid
    ','$mododecalculo','$mostrarimagen','$TiempoProceso','$tamano')";
282.
283.    $result = mysql_query($sql);
284.
285.    #echo "resultado : ".$result . "<br>\n" ;
286.    ##########################################################################################
    ########
287.
288.
```

Seguidamente tratamos los resultados del fichero gff y los traspasamos a otra tabla para facilitar su posterior análisis.

- La tabla hgff_rta contendrá los datos de los resultados. La estructura de esta tabla es la siguiente:

```
-- Table: hgff_rta

-- DROP TABLE IF EXISTS `hgff_rta`;

CREATE TABLE `hgff_rta` (
  `id_gff`        char(30) NOT NULL,
  `id_hcalculos`  char(20),
  `linea`         int,
  `seqname`       char(50) COLLATE `latin1_swedish_ci`,
  `source`        char(50) COLLATE `latin1_swedish_ci`,
  `feature`       char(50) COLLATE `latin1_swedish_ci`,
  `start`         int(20),
  `end`           int(20),
  `score`         float,
  `strand`        char COLLATE `latin1_swedish_ci`,
  `frame`         char(2) COLLATE `latin1_swedish_ci`,
  `comments`      varchar(200) COLLATE `latin1_swedish_ci`,
  /* Keys */
  PRIMARY KEY (`id_gff`)
) ENGINE = InnoDB;
```

La denominación de los campos respeta la definición de gff. El campo Id_gff que será el índice se calcula añadiendo el numero de línea formateado al id_calculo (ver líneas 314 y 315). Adicionalmente guardo también el id_calculo y el número de línea.

El campo id_hcalculos guarda integridad referencial con su homologo de la tabla hcalculos_rta.

```
289.    ##############################################################################################
    #########
290.
291.
292.    #almacenar el resultado en Mysql para luego hacer estadisticas y explorar los
    resultados
293.
294.    $bd= mysql_connect("localhost", "admin", "admin");
295.    mysql_select_db("anotagen", $bd);
296.
297.        $file=fopen($resultados,"rb");
298.        $contents=fread($file,filesize($resultados));
299.        fclose($file);
300.
301.            ##separamos por lineas
302.        $contents=str_replace("\r","\n",$contents);
303.        $contents=str_replace("\n\n","\n",$contents);
304.        $lineas=explode("\n",$contents);
305.            #contador para las lineas
306.            $ll=0;
307.
308.        foreach ($lineas as $linea){
309.                //Separamos los campos de gff si no son comentarios y quitamos las
    lineas en blanco
310.                    if (!(ereg("^#",$linea)) && ($linea != ""))
311.                    {
312.                            $campos=explode("\t",$linea);
313.                            $ll=$ll+1;
314.                            ##formateamos el indice añadiendo ceros...para que se
    muestre ordenada la tabla
315.                            $id_gffloc=$fastatemp . str_pad($ll, 5, "0", STR_PAD_LEFT);
316.                            $sql = "INSERT INTO hgff_rta ";
317.                            $sql .= " VALUES ('$id_gffloc',
    '$fastatemp','$ll','$campos[0]','$campos[1]','$campos[2]','$campos[3]','$campos[4]','$ca
    mpos[5]','$campos[6]','$campos[7]','$campos[8]');
318.                            ";
319.                #echo $sql .  "<br>";
320.                            $result = mysql_query($sql);
321.                            #echo $result .  "<br>";
322.                    }
323.
324.
325.        }
326.    ##############################################################################################
    #########
327.
328.
329.
330.    ?>
```

9 Tratamiento de los datos peticiones de la PEC

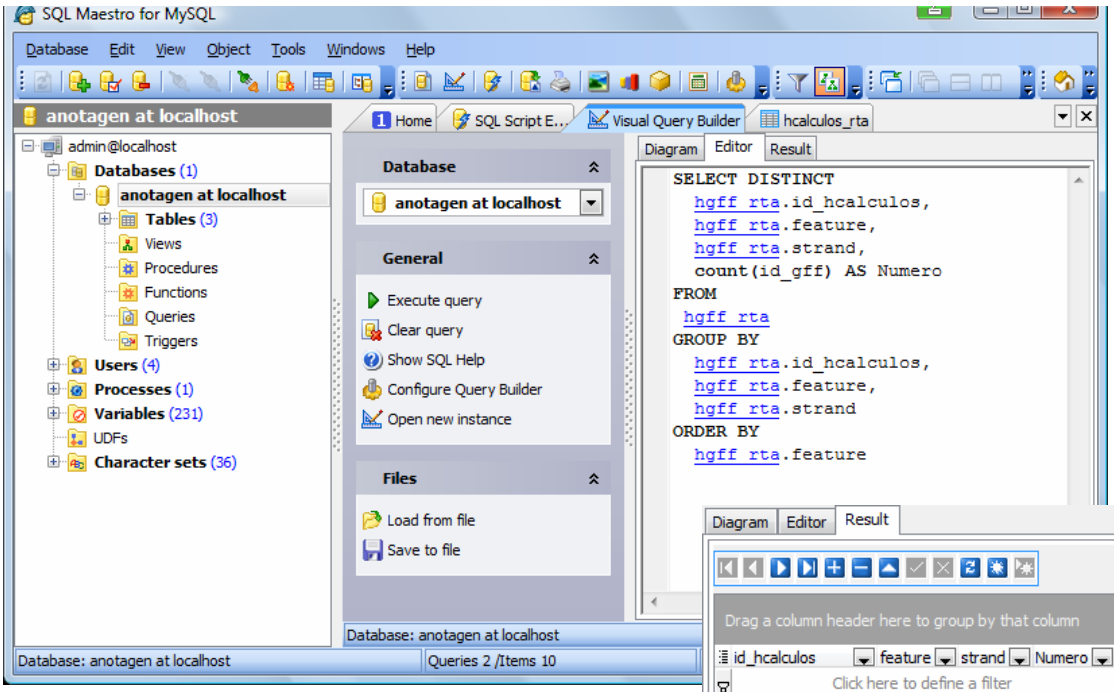
Lo primero aclarar una de las peticiones de la PEC no implementada. Se pide que guardemos

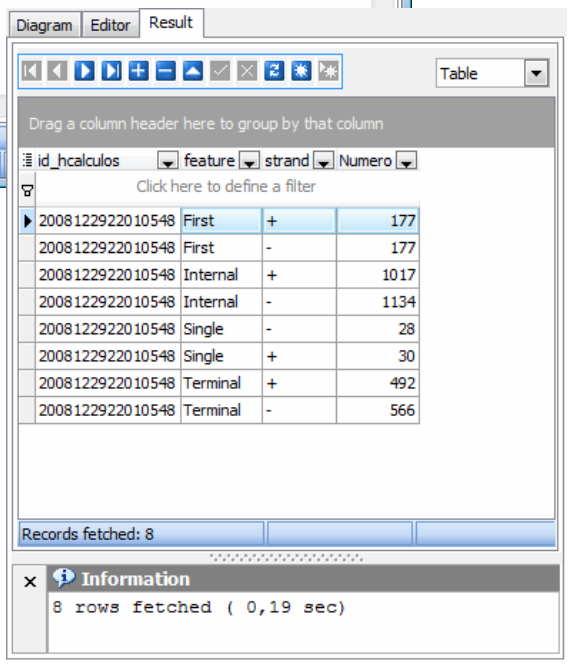
- numero de genes en la predicción
- numero de exones en la predicción

Tanto el número de genes como el de exones se pueden calcular con una consulta sobre la tabla hgff_rta. Dado que este es un análisis de tabla, lo podemos guardar o no, ya que se puede calcular fácilmente después y mostrarlo al usuario.

Esta consulta no la he implementado en la última versión operativa, pero sería la siguiente (muestro como montarla en MySQL maestro):

Como se puede apreciar, desarrollar con SQL Maestro es espectacularmente rápido, esta consulta la he montado mientras escribía este resumen de la PEC el 31 de diciembre, me ha costado 3 minutos, incluido el tiempo de testearla.



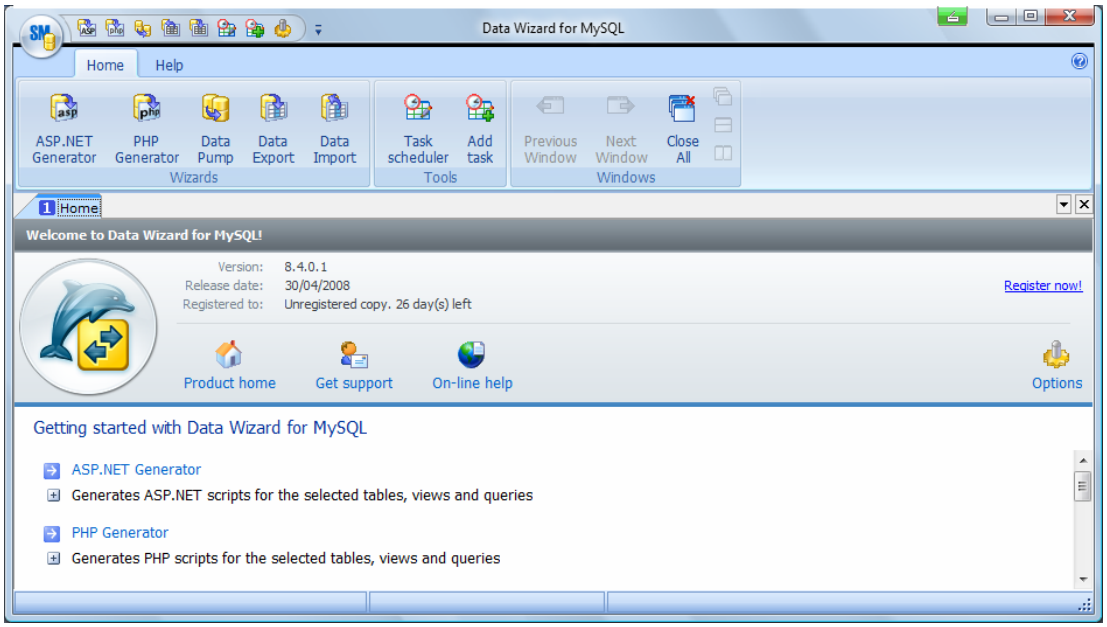


El resultado de la consulta es:

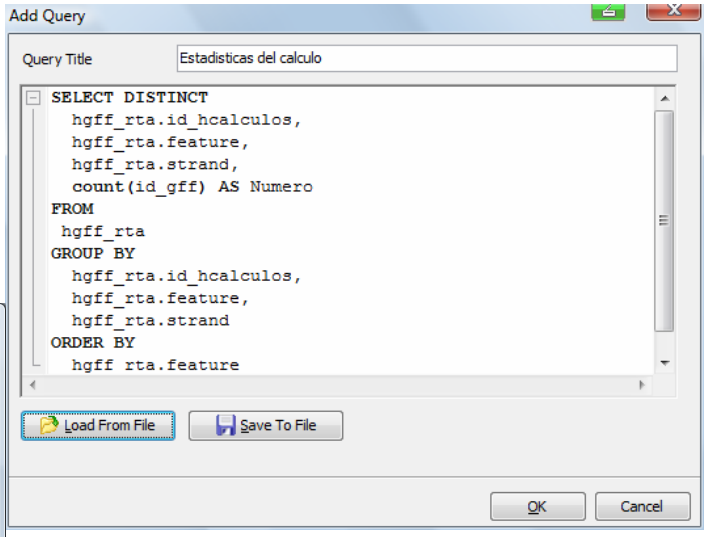
Como vemos, el resultado tiene una relación varios a uno con tabla maestra cálculos_rta, así que la mejor opción es ejecutar la consulta cada vez que la necesitemos, y no guardarla (al fin y al cabo al sistema le cuesta 0,19 segundos).

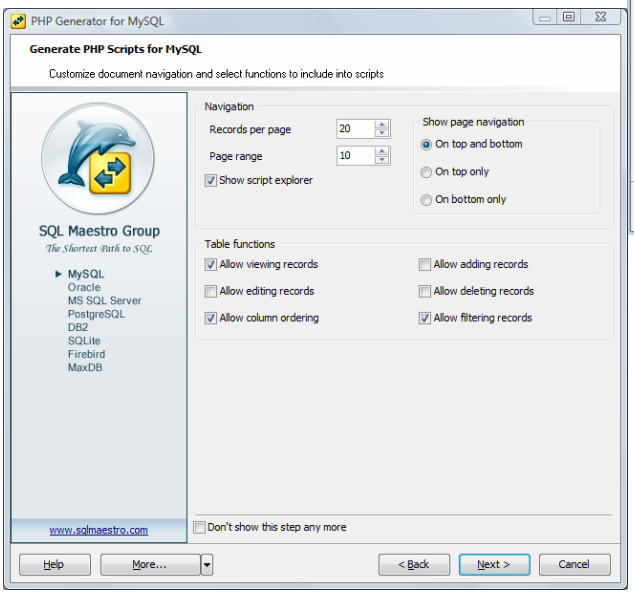
Si quisiéramos implementarla en la aplicación haría:

- Guardo la consulta. En el servidor, he dejado una carpeta para ir guardando las consultas.
- Lanzo el data wizard y selecciono generar código php



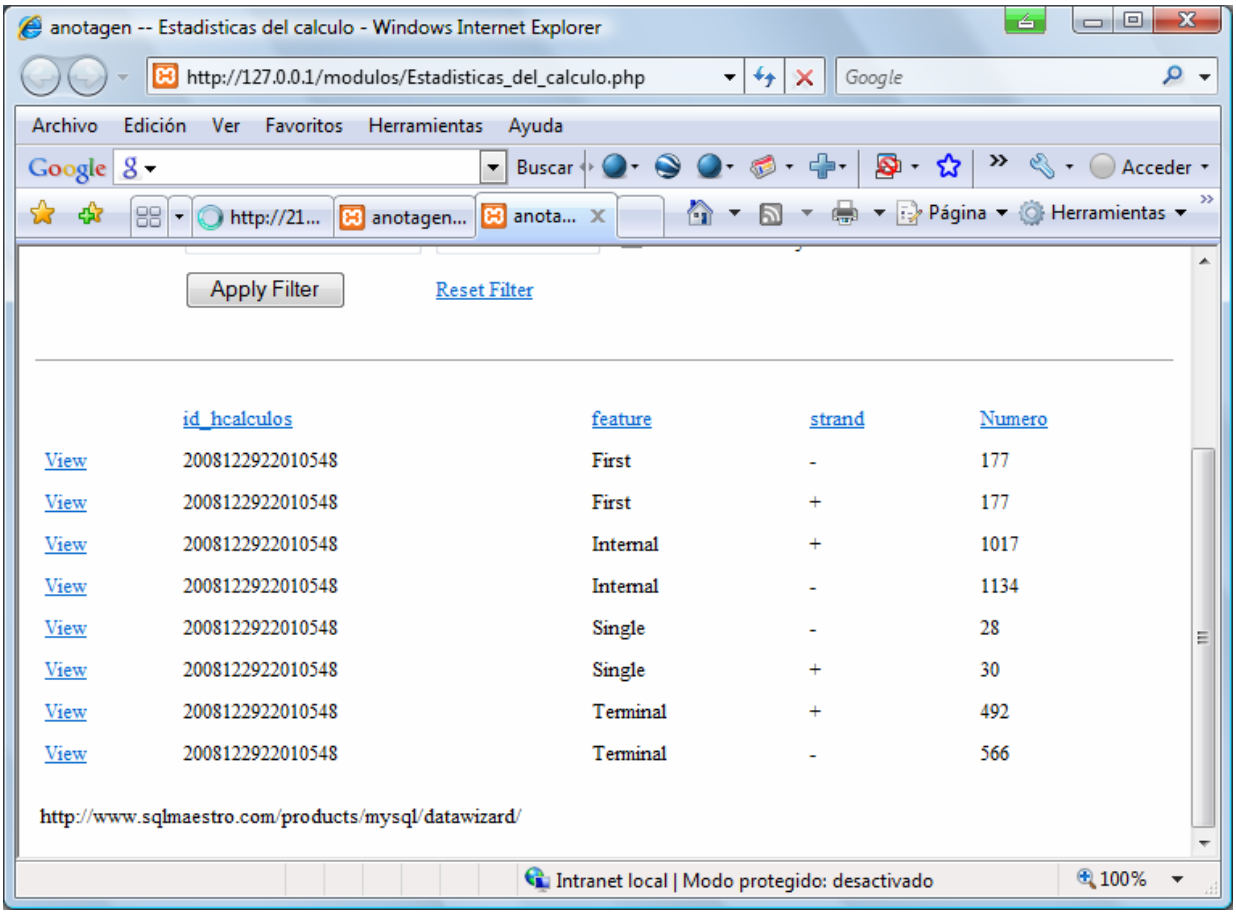
- Selecciono la base de datos y la consulta guardada:
- En las páginas siguientes selecciono distintas opciones del comportamiento y formato. El formato no me importa mucho, ya que posteriormente solo usaré la parte de los scripts de php.



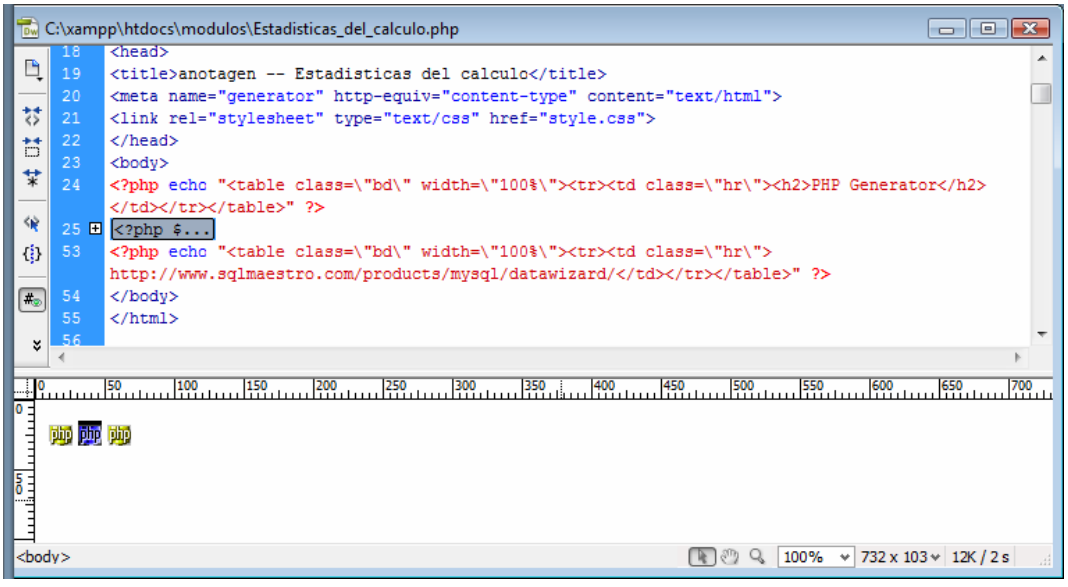


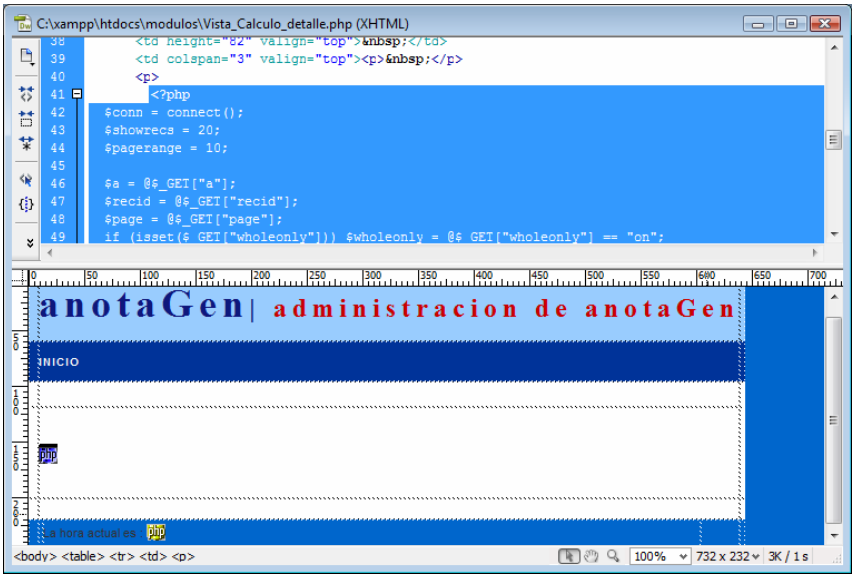
Después de varias pantallas más de configuración le doy a generar y ya lo tengo.

El siguiente paso es un tanto más manual, y pasa por cortar y pegar las partes del script que deseemos en nuestra plantilla. Y luego establecer los vínculos en el resto de páginas.

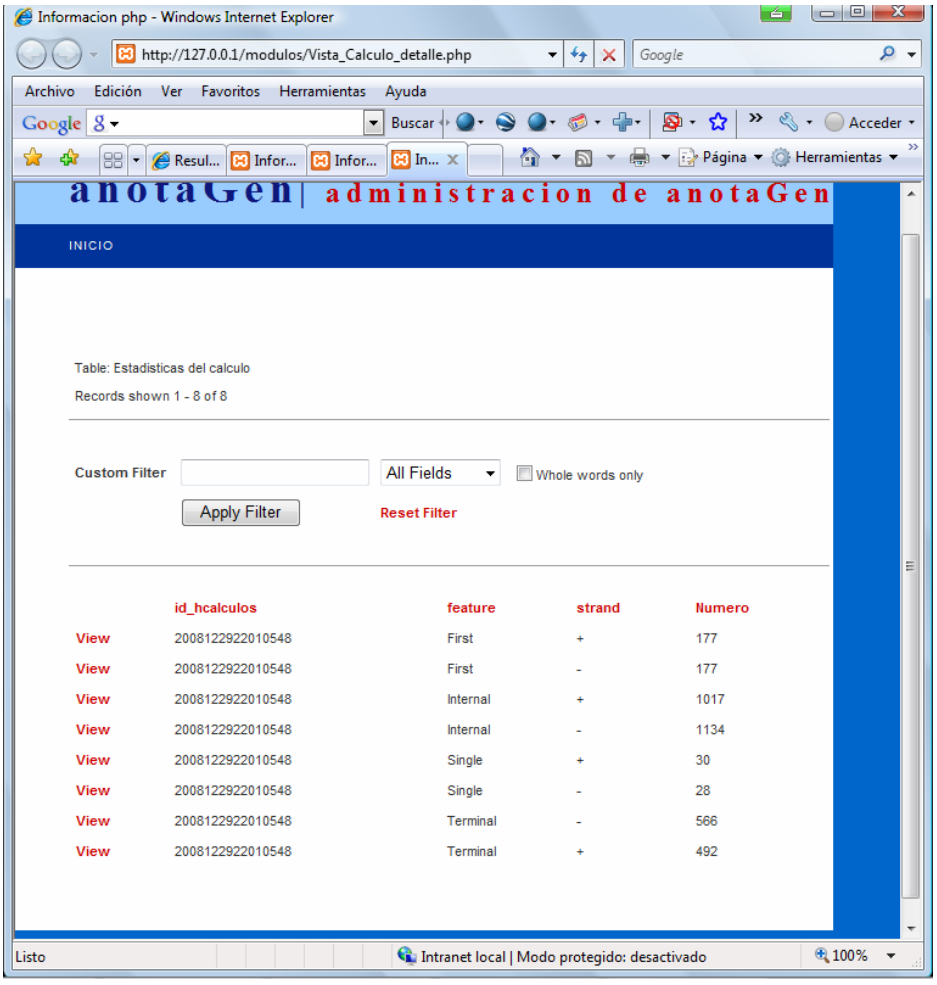


- Si editamos el fichero estadisticas_del_calculo.php en dreamweaver, vemos que son tres scripts, y que nos interesa el segundoy el tercero, , el primero es el que se encarga de los encabezados y pies. Lo cortamos y lo pegamos en nuestra plantilla.



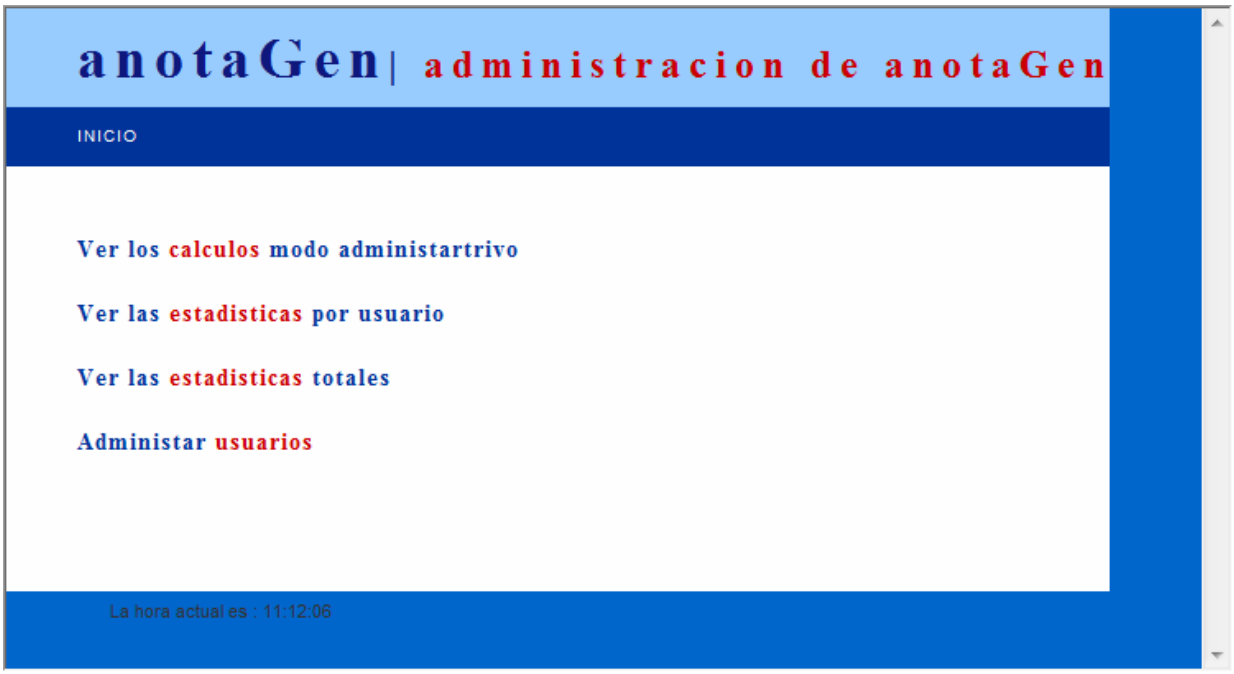


Y a funcionar. Todos los módulos que se pueden encontrar el servidor se han desarrollado de la misma forma (bueno casi, hay algunos que tienen unos cuantos retoques más)



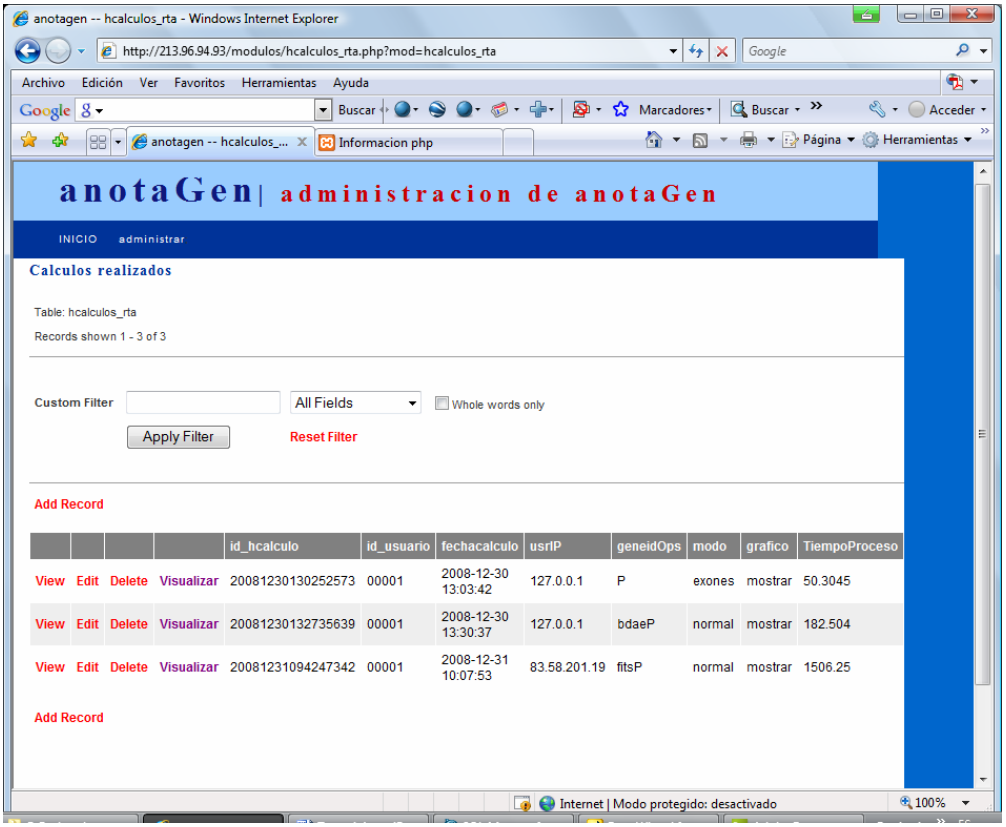
¡¡¡Creo que evidente que es cada cosa!!!

10 Módulos administrativos disponibles:



Desde el menú administrar se puede acceder a una serie de estadísticas de los cálculos realizados por los usuarios. (Se podrían poner algunas más, sobre todo relativas a los cálculos entre fechas):

Cálculos en modo administrativo:



La vista de cálculos en modo administrativo permite editar manualmente la tabla hcalculos_rta. Desde aquí se deberían implementar la llamada a las rutinas que borran los archivos de un cálculo al pulsar “Delete”, esta parte no esta implementada. Este formulario aún no tiene implementada la rutina de seguridad (falta de tiempo). Igualmente, he añadido la posibilidad de visualizar el resultado de un cálculo ya realizado desde el link “visualizar”.

Estadísticas de usuario

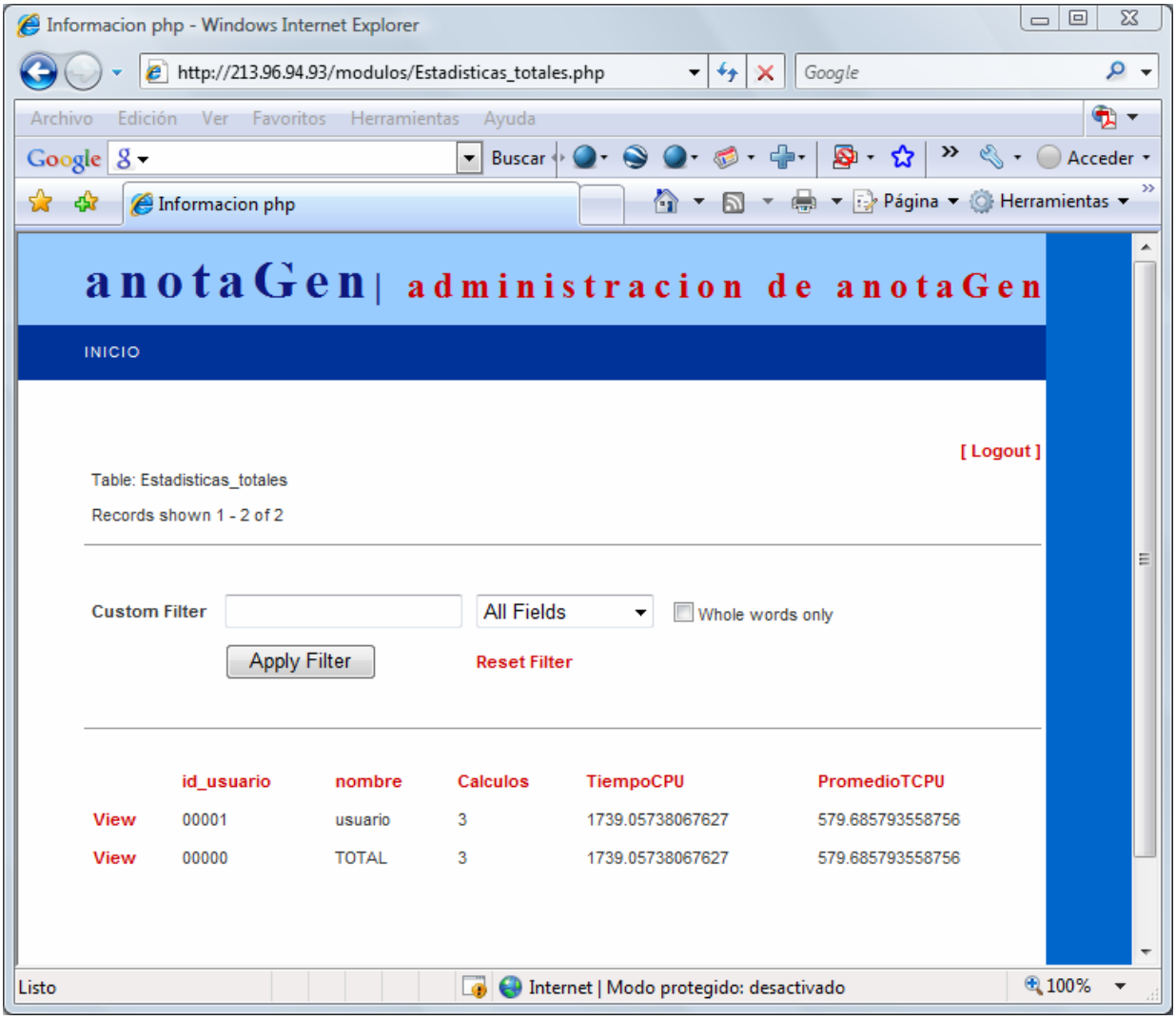
Este modulo se muestran las estadísticas relativas a los cálculos de cada usuario:

- Numero de cálculos realizado
- El tiempo total de CPU gastado
- El promedio de tiempo de CPU

Estadísticas totales

Muestra las estadísticas totales de los cálculos. Tiene la seguridad implementada.

La consulta que ejecuta esta página es un poco curiosa. Esta forma de consulta se usa para unir datos aparentemente no coherentes, es un poco chapuza pero muuuuy útil. (su tiempo de ejecución es menor que usando group by).



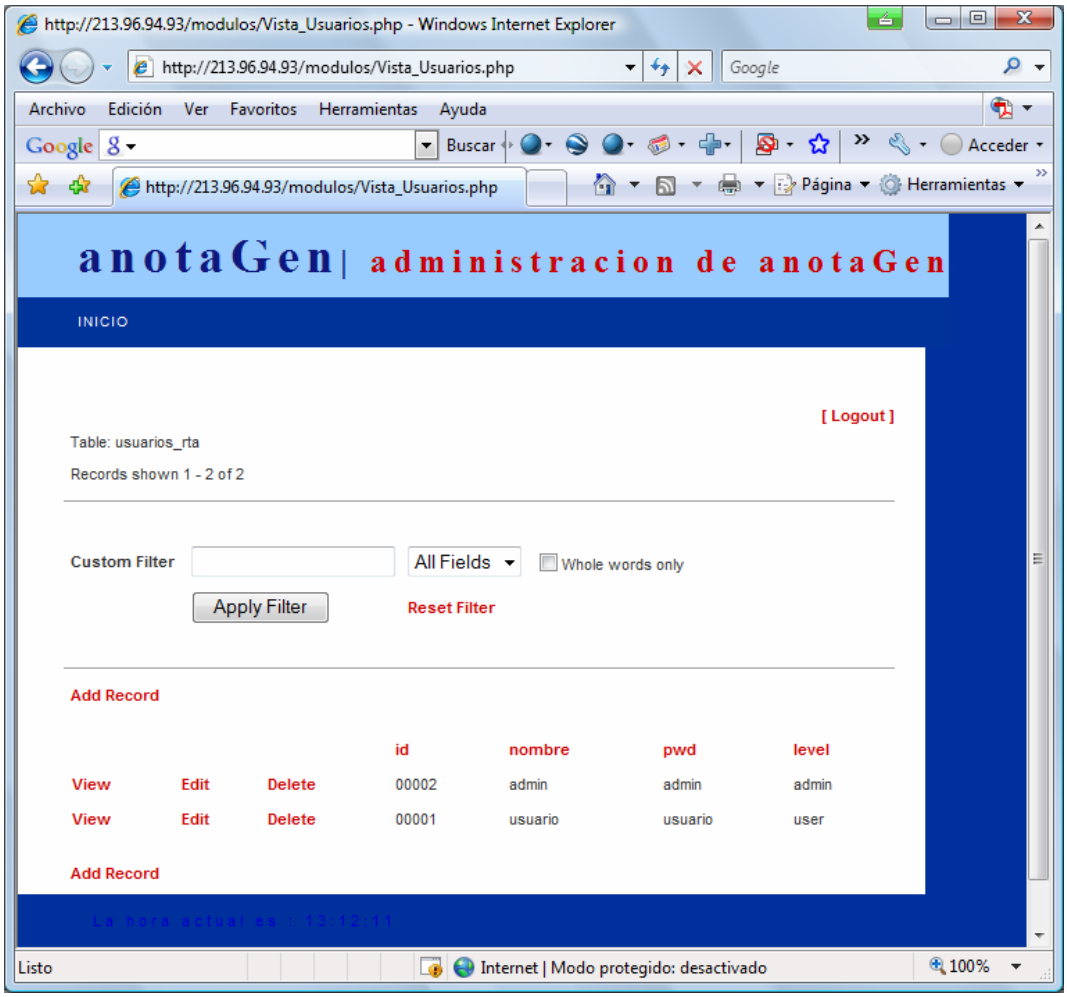

```
SELECT
  hcalculos_rta.id_usuario,
  usuarios_rta.nombre,
  count(id_hcalculo) AS Calculos,
  sum(TiempoProceso) AS TiempoCPU,
  avg(TiempoProceso) AS PromedioTCPU
FROM
 hcalculos_rta
 INNER JOIN usuarios_rta ON (hcalculos_rta.id_usuario=usuarios_rta.id)
GROUP BY
  hcalculos_rta.id_usuario,
  usuarios_rta.nombre

  UNION

  SELECT
 '00000' as id_usuario,
 'TOTAL' as nombre,
  count(id_hcalculo) AS Calculos,
  sum(TiempoProceso) AS TiempoCPU,
  avg(TiempoProceso) AS PromedioTCPU
FROM
 hcalculos_rta
```

Administrar usuarios

Para añadir o quitar usuarios, este módulo esta a mitad de acabar. Falta ponerle las plantillas correctamente. Tiene la seguridad implementada.



Modulo de autentificación de usuarios.

Este lo explico por encima. Es una subrutina que comprueba en la tabla usuarios si existe el usuario (falta configurar el level). Si el usuario y pwd coinciden pinta en pantalla la pagina y si no, le pone el mensaje de error. También pone en el borde superior un link a si mismo con el parámetro “logout” para finalizar la sesión.

Y ya esta ¡!!!!!. Ahora a preparar la nochevieja¡¡¡¡¡¡ …

El esfuerzo ha merecido la pena, casi he aprendido php¡¡¡¡¡