Manual del Programador

SISTEMA DE GESTIÓN DE NORMAS Y PUATAS

DIRECCION GENERAL DE PLANIFICACIÓN Y COORDINACIÓN DE PROYECTOS

SISTEMA DE USO INTERNO

Autor: GILLI, Andrés Alberto

ÍNDICE TEMÁTICO

1. Introducción	**3**
1.1. Objetivos	3
1.2. Tipos de Usuario	3
1.3. Tecnologías Utilizadas	3
1.4. Librerías Utilizadas	4
2. Modelo Físico de Datos	**6**
2.1. Del Sistema	6
2.2. Del Registro de Actividad	7
3. Diccionario de Datos	**8**
3.1. Del Sistema	8
3.2. Del Registro de Actividad	10
4. Especificación de los Procesos	**11**
4.1. Registro de Usuarios	11
4.2. Ingreso al Sistema	12
4.3. Módulo de Agentes	13
4.4. Módulo de Informes	18
4.5. Módulo de Administración	22
4.6. Salir del Sistema	36
5. Reportes	**37**
5.1. Por Norma	37
5.2. Por Pauta	37
5.3. Por Usuario	38

1. Introducción

1.1. OBJETIVOS

El Sistema de Gestión de Normas y Pautas esta diseñado con el objetivo principal de ofrecer un acceso ágil, consistente, actualizado y seguro a la información necesaria para el correcto funcionamiento de la Mesa de Orientación y Servicios del Ministerio de Educación de la Provincia de Santa Fe.

Se entiende por información necesaria todas las Normas y Pautas de Trabajo definidas por las distintas instituciones gubernamentales.

- ***Normas:*** incluimos aquí Decretos, Resoluciones y Disposiciones.
- ***Pautas:*** se incluyen aquí todas las pautas de trabajo definidas por las áreas de las cuales depende en forma directa la Mesa de Orientación y Servicios.

Además se pretende registrar los Contactos relacionados con cada una de estas Normas y Pautas.

1.2. TIPOS DE USUARIO

Este Sistema presenta dos perfiles de usuario claramente diferenciados, los Agentes y los Administradores.

Los ***Agentes*** tendrán acceso total a la visualización de Normas y Pautas, así como también a las relaciones entre ellas, documentación asociada y los contactos relacionados con las mismas. Además de ello, podrán actualizar algunos de sus datos personales y cambiar su contraseña.

Los ***Administradores***, tendrán todos los privilegios que tienen los Agentes, pero además serán los responsables de las Altas, Bajas y Modificaciones de las Normas, Pautas y Contactos.
Los ***Administradores*** tendrán también la facultad de Activar, Desactivar, Modificar y Eliminar a otro Usuarios del Sistema dependiendo de las necesidades.

1.3. TECNOLOGÍAS UTILIZADAS

Este Sistema ha sido diseñado para funcionar en un entorno WEB, utilizando varias de las tecnologías disponibles para este tipo de sistemas.
Se detallan a continuación las más relevantes:

- HTML
- XML
- CSS
- JAVASCRIPT
- PHP5

- AJAX

Mediante la Librería ADOdb5 se integra a este Sistema una Base de Datos diseñada en MySQL.
Se escogió MyISAM como tipo de Tablas para la Base de Datos, además se incluyeron tipos de índice FULLTEXT.
Estos tipos de tablas e índices no son compatibles con ANSI SQL, pero se determino utilizarlos de todas formas debido a que aceleran en gran medida las consultas sobre la base, y esta será la mayor demanda del Sistema.

Si se necesita subir el Sistema en algún servidor con una base de datos que no sea MySQL solamente hay que eliminar los índices FULLTEXT y cambiar el Tipo de Tabla de MyISAM a InnoDB. Hecho esto la Base será compatible con cualquier Motor de Base de Datos.

Nota: si se desea poner en funcionamiento el Sistema en un servidor propio se necesita un Servidor HTTP. Se recomienda utilizar alguna versión de Apache, ya que fue el utilizado en todas la etapas de desarrollo. De todas maneras se podría utilizar cualquier otro disponible.

1.3. LIBRERÍAS UTILIZADAS

Todas las Librerías que se detallan a continuación se encuentran escritas en código PHP.

ADOdb5: se utiliza para integrar PHP con la base de datos, que en este caso esta diseñada en MySQL. La ventaja de utilizar esta librería es que si cambiamos la Base por otra diseñada, por ejemplo en PostgreSQL, solo habrá que cambiar una línea de código. En Secciones posteriores se verá que línea de código habrá que cambiar.

DOMPDF: se utiliza para convertir código HTML en un documento PDF, esto se necesita únicamente para generar los informes en formato PDF.

XAJAX: esta librería para PHP permite crear aplicaciones WEB basadas en la tecnología AJAX.

TEMPLATE: librería diseñada para el acceso a las plantillas HTML a través de PHP.

VALIDACION: librería diseñada para realizar varios tipos de validación.

Se detallan a continuación algunas librerías propias utilizadas en este desarrollo.

CONEXION: esta librería se desarrollo con el objetivo de mantener una conexión abierta con la Base de Datos utilizando ADOdb5. Además se configura la conexión para no tener inconvenientes con acentos y que la respuesta de las consultas sean arreglos con los mismos índices que tiene la Base de Datos.

CORTAR_PALABRA: si tenemos una cadena de texto demasiado larga, con esta librería logramos guardarla como otra cadena separada en varios renglones. Esto se utiliza para mantener las proporciones en las Tablas de las Plantillas.

EDITAR_FECHA: esta librería es utilizada con tres fines. Convertir una fecha en formato MySQL, convertir una fecha al formato utilizado en Argentina y validar la fecha ingresada por un usuario.

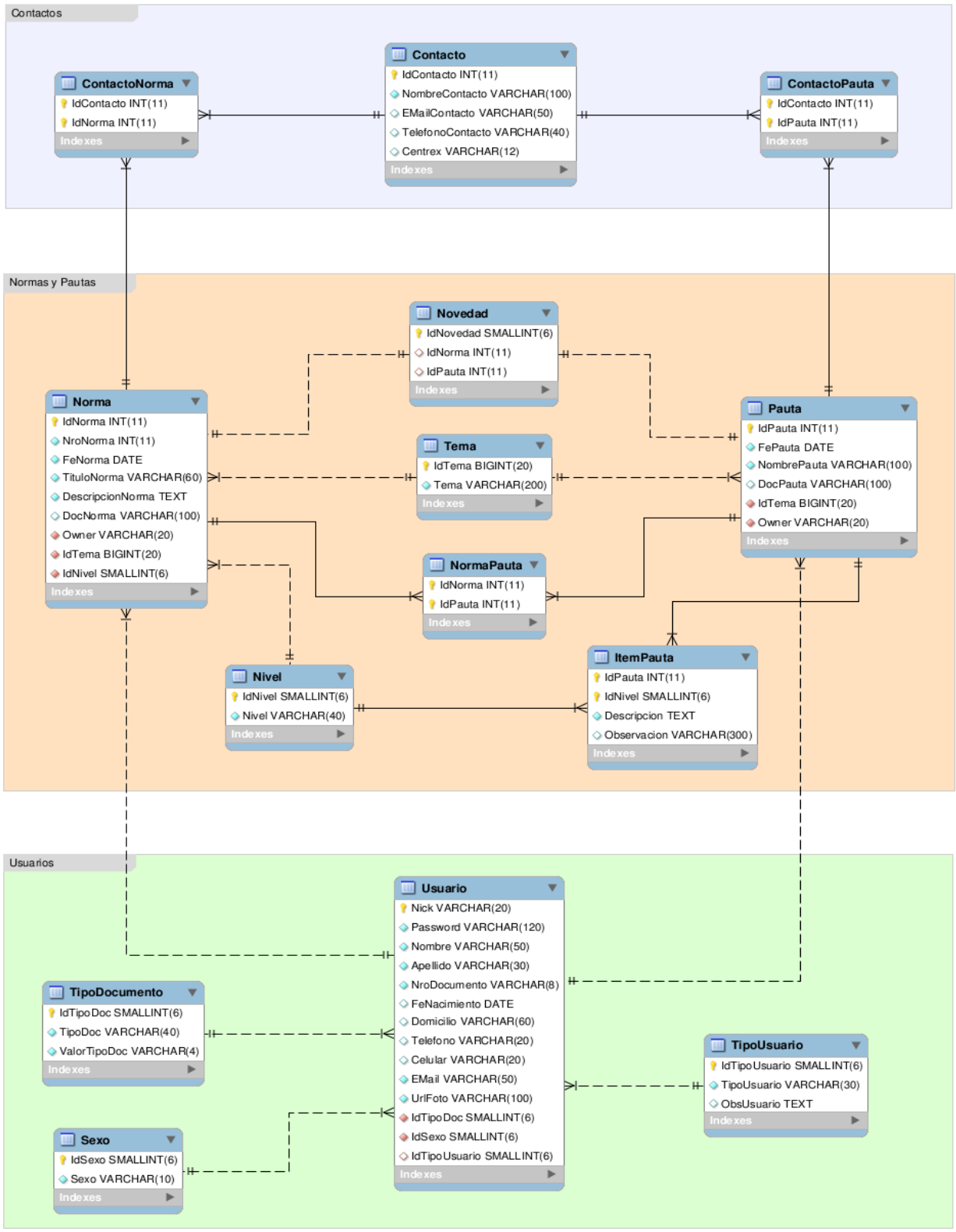
REGISTRAR_LOG: esta librería es utilizada para registrar los distintos registros de actividad. Es decir, incluye funciones para registrar la actividad de los Agentes, pero también incluye otras para registrar las Altas, Bajas y Modificaciones realizadas por los Administradores.

SEGURIDAD: esta librería controla que el usuario haya pasado por la etapa de login antes de acceder a la parte del sistema que la incluya. Si el usuario no esta logueado lo lleva a la pantalla de login. Si el usuario está logueado lo deja continuar.

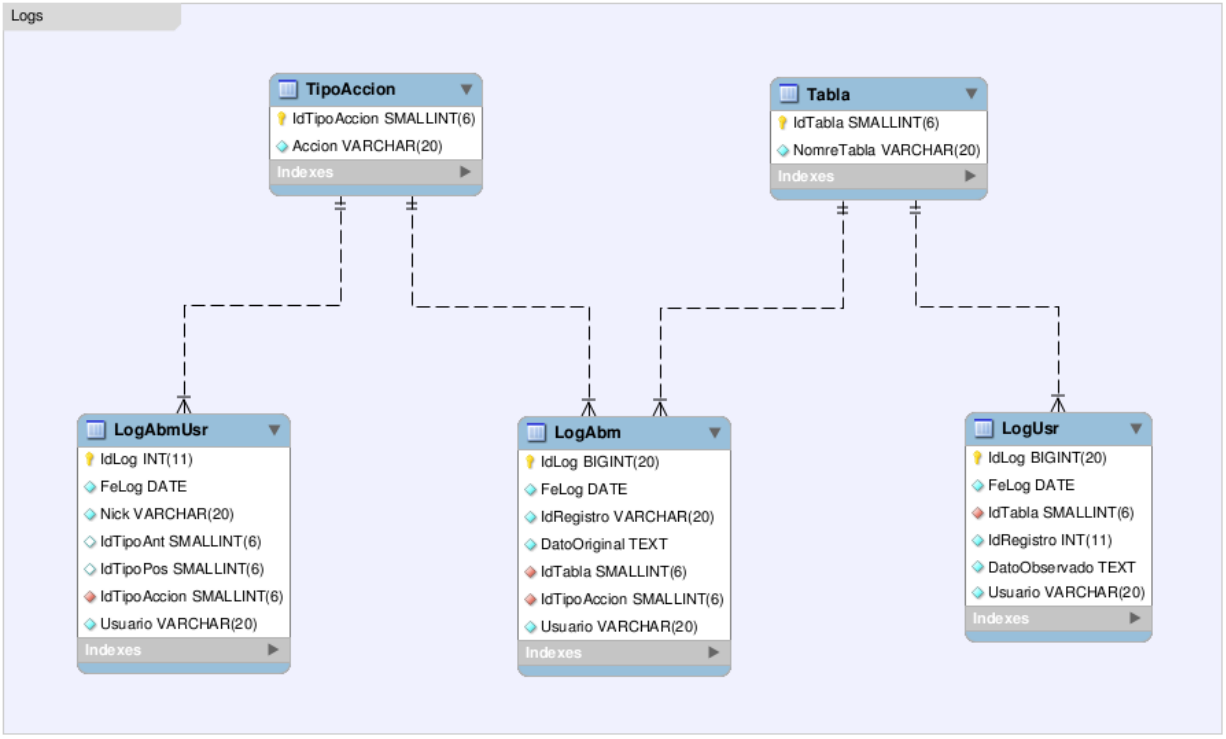
SEGURIDAD2: esta librería no solo controla que el usuario haya pasado por la etapa de login antes de acceder a la parte del sistema que la incluya, sino que también verifica que el usuario sea del tipo Administrador. Si el usuario no esta logueado lo lleva a la pantalla de login y si esta logueado pero no es Administrador lo lleva al index. Caso contrario lo deja continuar.

2. Modelo Físico de Datos

2.1. DEL SISTEMA



2.2. DEL REGISTRO DE ACTIVIDAD



Se anexan a este manual las dos imágenes anteriores en formato PDF.

3. Diccionario de Datos

3.1. DEL SISTEMA

TABLA	**ContactoNorma**				
NOMBRE	SIGNIFICADO	TIPO	LONGITUD	NATURALEZA	REGLA DE CALCULO
IdContacto	Identificador del Contacto relacionado con la Norma	Numérico	11	Usuario	No Nulo
IdNorma	Identificador de la Norma relacionada con el Contacto	Numérico	11	Usuario	No Nulo
TABLA	**ContactoPauta**				
NOMBRE	SIGNIFICADO	TIPO	LONGITUD	NATURALEZA	REGLA DE CALCULO
IdContacto	Identificador del Contacto relacionado con la Pauta	Numérico	11	Usuario	No Nulo
IdPauta	Identificador de la Pauta relacionada con el Contacto	Numérico	11	Usuario	No Nulo
TABLA	**Contacto**				
NOMBRE	SIGNIFICADO	TIPO	LONGITUD	NATURALEZA	REGLA DE CALCULO
IdContacto	Identificador del Contacto	Numérico	11	Sistema	Auto incremental en 1
NombreContacto	Nombre del Contacto	Alfanumérico	100	Usuario	No Nulo
EmailContacto	Correo electrónico del Contacto	Alfanumérico	50	Usuario	No pueden ser los tres Nulos
TelefonoContacto	Teléfono del contacto y si corresponde número de Interno	Alfanumérico	40	Usuario	
Centrex	Centrex del Contacto	Alfanumérico	12	Usuario	
TABLA	**Norma**				
NOMBRE	SIGNIFICADO	TIPO	LONGITUD	NATURALEZA	REGLA DE CALCULO
IdNorma	Identificador de la Norma	Numérico	11	Sistema	Auto incremental en 1
NroNorma	Número de la Norma	Numérico	11	Usuario	Si no ingresa nada se asigna un número negativo. Luego cuando se muestra aparece No Corresponde.
FeNorma	Fecha de entrada en Vigencia de la Norma	Fecha	N/C	Usuario	No Nulo
TituloNorma	Título de la Norma	Alfanumérico	60	Usuario	No Nulo
DescripcionNorma	Descripción de la Norma	Alfanumérico	Libre	Usuario	No Nulo
DocNorma	Url del Documento que establece la Norma	Alfanumérico	100	Sistema	Puede ser Nulo. Caso contrario se guarda el documento con nombre: Norma+IdNorma
Owner	Nombre de Usuario del Administrador que ingreso la Norma al Sistema	Alfanumérico	20	Sistema	No Nulo
IdTema	Identificador del Tema del cual se trate la Norma	Numérico	20	Usuario	No Nulo
IdNivel	Identificador del Nivel de Educación al que este referida la Norma	Numérico	6	Usuario	No Nulo
TABLA	**Pauta**				
NOMBRE	SIGNIFICADO	TIPO	LONGITUD	NATURALEZA	REGLA DE CALCULO
IdPauta	Identificador de la Pauta	Numérico	11	Sistema	Auto incremental en 1
FePauta	Fecha de entrada en Vigencia de la Pauta	Fecha	N/C	Usuario	No Nulo
NombrePauta	Nombre o Título de la Pauta	Alfanumérico	100	Usuario	No Nulo
DocPauta	Url del Documento relacionado a la Pauta	Alfanumérico	100	Sistema	Puede ser Nulo. Caso contrario se guarda el documento con nombre: Pauta+IdPauta
IdTema	Identificador del Tema del cual se trate la Pauta	Numérico	20	Usuario	No Nulo
Owner	Nombre de Usuario del Administrador que ingreso la Pauta al Sistema	Alfanumérico	20	Sistema	No Nulo
TABLA	**ItemaPauta**				
NOMBRE	SIGNIFICADO	TIPO	LONGITUD	NATURALEZA	REGLA DE CALCULO
IdPauta	Identificador de la Pauta a la cual pertenezca el ítem	Numérico	11	Usuario	No Nulo
IdNivel	Identificador del Nivel de Educación al que este referido este ítem	Numérico	6	Usuario	No Nulo
Descripcion	Descripción del ítem	Alfanumérico	Libre	Usuario	No Nulo
Observacion	Reservado para realizar alguna observación referida a este ítem	Alfanumérico	300	Usuario	Puede ser Nulo

Continua la tabla anterior.

TABLA	**Nivel**				
NOMBRE	SIGNIFICADO	TIPO	LONGITUD	NATURALEZA	REGLA DE CALCULO
IdNivel	Identificador del Nivel de Educación	Numérico	6	Usuario	Estáticos. Una vez definidos son registrados por el Administrador directamente contra la Base de
Nivel	Nombre del Nivel de Educación	Caracteres	40	Usuario	
TABLA	**NormaPauta**				
NOMBRE	SIGNIFICADO	TIPO	LONGITUD	NATURALEZA	REGLA DE CALCULO
IdNorma	Identificador de la Norma relacionada con la Pauta	Numérico	11	Usuario	No Nulo
IdPauta	Identificador de la Pauta relacionada con la Norma	Numérico	11	Usuario	No Nulo
TABLA	**Tema**				
NOMBRE	SIGNIFICADO	TIPO	LONGITUD	NATURALEZA	REGLA DE CALCULO
IdTema	Identificador del Tema	Numérico	20	Sistema	Auto incremental en 1
Tema	Nombre del Tema	Alfanumérico	200	Usuario	No Nulo
TABLA	**Novedad**				
NOMBRE	SIGNIFICADO	TIPO	LONGITUD	NATURALEZA	REGLA DE CALCULO
IdNovedad	Identificador de la Novedad, indica la posición en la cual se mostrará	Numérico	6	Usuario	La cantidad de Novedades es estática, el id variará de 1 a la cantidad definida por el Administrador
IdNorma	Identificador de la Norma relacionada con la Pauta	Numérico	11	Usuario	Puede ser Nulo
IdPauta	Identificador de la Pauta relacionada con la Norma	Numérico	11	Usuario	Puede ser Nulo
TABLA	**Usuario**				
NOMBRE	SIGNIFICADO	TIPO	LONGITUD	NATURALEZA	REGLA DE CALCULO
Nick	Nombre de Usuario	Alfanumérico	20	Usuario	Clave Primaria, por lo tanto debe ser único
Password	Contraseña del Usuario	Alfanumérico	120	Usuario	Se codifica la contraseña ingresada con la función SHA1. Por lo tanto el máximo de caracteres ingresado por el usuario será 20
Nombre	Nombre del Usuario	Caracteres	50	Usuario	No Nulo
Apellido	Apellido del Usuario	Caracteres	30	Usuario	No Nulo
NroDocumento	Número de Documento del Usuario	Numérico	8	Usuario	No Nulo
FeNacimiento	Fecha de nacimiento del Usuario	Fecha	N/C	Usuario	No Nulo
Domicilio	Domicilio actual del Usuario	Alfanumérico	60	Usuario	Puede ser Nulo
Telefono	Número de Teléfono Fijo del Usuario	Alfanumérico	20	Usuario	No pueden ser los tres Nulos
Celular	Número de Celular del Usuario	Alfanumérico	20	Usuario	
Email	Correo electrónico del Usuario	Alfanumérico	50	Usuario	
UrlFoto	Url de la Foto del Usuario	Alfanumérico	100	Usuario	No Nulo. Se reserva para futuras modificaciones del Sistema. Actualmente guarda un Avatar predefinido según el sexo del usuario
IdTipoDoc	Identificador del Tipo de Documento del Usuario	Numérico	6	Usuario	No Nulo
IdSexo	Identificador del Sexo del Usuario	Numérico	6	Usuario	No Nulo
IdTipoUsuario	Identificador del Tipo de Usuario	Numérico	6	Usuario	Nulo. Al activar el usuario un Administrador asignará el Tipo de Usuario
TABLA	**TipoDocumento**				
NOMBRE	SIGNIFICADO	TIPO	LONGITUD	NATURALEZA	REGLA DE CALCULO
IdTipoDoc	Identificador del Tipo de Documento	Numérico	6	Usuario	Los Tipos de Documento son Estáticos. Una vez definidos son registrados por el Administrador directamente contra la Base de Datos
TipoDoc	Tipo de Documento	Caracteres	40	Usuario	
ValorTipoDoc	Este campo es el que se mostrará al usuario para que seleccione el Tipo de Documento	Caracteres	4	Usuario	
TABLA	**TipoUsuario**				
NOMBRE	SIGNIFICADO	TIPO	LONGITUD	NATURALEZA	REGLA DE CALCULO
IdTipoUsuario	Identificador del Tipo de Usuario	Numérico	6	Usuario	Los Tipos de Usuario son Estáticos. Una vez definidos son registrados por el Administrador directamente contra la Base de Datos
TipoUsuario	Descripción del Tipo de Usuario	Caracteres	30	Usuario	
ObsUsuario	Campo reservado para comentar que acciones puede realizar el Tipo de Usuario	Caracteres	Libre	Usuario	

Continua la tabla anterior.

TABLA	**Sexo**				
NOMBRE	SIGNIFICADO	TIPO	LONGITUD	NATURALEZA	REGLA DE CALCULO
IdSexo	Identificador del Sexo	Numérico	6	Usuario	Los Dados del Sexo son Estáticos. Una vez definidos son registrados por el Administrador directamente contra la Base de Datos
Sexo	Sexo	Caracteres	10	Usuario	

3.2. DEL REGISTRO DE ACTIVIDAD

TABLA	**TipoAccion**				
NOMBRE	SIGNIFICADO	TIPO	LONGITUD	NATURALEZA	REGLA DE CALCULO
IdTipoAccion	Identificador del Tipo de Acción	Numérico	6	Sistema	No Nulo
Accion	Acción efectuada por el Usuario	Caracteres	20	Sistema	No Nulo
TABLA	**Tabla**				
NOMBRE	SIGNIFICADO	TIPO	LONGITUD	NATURALEZA	REGLA DE CALCULO
IdTabla	Identificador de la Tabla	Numérico	6	Sistema	No Nulo
NombreTabla	Nombre del la Tabla	Caracteres	20	Sistema	No Nulo
TABLA	**LogAbmUsr**				
NOMBRE	SIGNIFICADO	TIPO	LONGITUD	NATURALEZA	REGLA DE CALCULO
IdLog	Identificador del Log de Altas, Bajas y Modificaciones de Usuarios	Numérico	11	Sistema	Auto incremental en 1
FeLog	Fecha en que se produjo el movimiento	Fecha	N/C	Sistema	Fecha Actual
Nick	Nombre del Usuario afectado por el movimiento	Alfanumérico	20	Usuario	No Nulo
IdTipoAnt	Guarda el Tipo de Usuario del Usuario afectado antes efectuar el movimiento	Numérico	6	Sistema	No Nulo
IdTipoPos	Guarda el Tipo de Usuario del Usuario luego de efectuado el Movimiento	Numérico	6	Usuario	No Nulo
IdTipoAccion	Identificador del Tipo de Acción efectuada	Numérico	6	Usuario	No Nulo
Usuario	Nombre del Usuario que realizo este movimiento	Alfanumérico	20	Sistema	No Nulo
TABLA	**LogAbm**				
NOMBRE	SIGNIFICADO	TIPO	LONGITUD	NATURALEZA	REGLA DE CALCULO
IdLog	Identificador del Log de Altas, Bajas y Modificaciones de todas las Tablas excepto la tabla Usuario	Numérico	20	Sistema	Auto incremental en 1
FeLog	Fecha en que se produjo el movimiento	Fecha	N/C	Sistema	Fecha Actual
IdRegistro	Identificador del Registro Afectado	Alfanumérico	20	Usuario	No Nulo
DatoOriginal	Información relevante del dato afectado. Ejemplo: Si se elimina una Norma aquí quedará registrado el Título de la Norma	Alfanumérico	Libre	Usuario	No Nulo
IdTabla	Identificador de la Tabla afectada	Numérico	6	Usuario	No Nulo
IdTipoAccion	Identificador del Tipo de Acción efectuada	Numérico	6	Usuario	No Nulo
Usuario	Nombre del Usuario que realizo este movimiento	Alfanumérico	20	Sistema	No Nulo
TABLA	**LogUsr**				
NOMBRE	SIGNIFICADO	TIPO	LONGITUD	NATURALEZA	REGLA DE CALCULO
IdLog	Identificador del Log de que registra todas las Normas y Pautas visitadas por un Usuario	Numérico	20	Sistema	Auto incremental en 1
FeLog	Fecha en que se produjo el movimiento	Fecha	N/C	Sistema	Fecha Actual
IdTabla	Identificador de la Tabla observada	Numérico	6	Usuario	No Nulo
IdRegistro	Identificador del Registro observado	Numérico	6	Usuario	No Nulo
DatoObservado	Información relevante del dato observado. Ejemplo: Si se ingresa a ver una Norma aquí quedará registrado el Título de la Norma	Alfanumérico	Libre	Usuario	No Nulo
Usuario	Nombre del Usuario que ingreso a ver la Norma o la Pauta	Alfanumérico	20	Sistema	No Nulo

4. Especificación de los Procesos

Comenzaremos siguiendo la secuencia más lógica para ir explicando los distintos procesos.
Como toda página web contiene un index que se referencia cada vez que ingresamos a la misma. En nuestro caso el index utiliza la librería “seguridad.php” ya explicada en la Sección 1.4. Como el usuario al ingresar no estará logueado, la misma librería lo enviará a la página de login.
En dicha página tendrá dos opciones, una es loguearse y la otra es registrarse.

4.1. REGISTRO DE USUARIOS

registrar_usuario.php

Librerías utilizadas:
- template.php,
- conexion.php

La función del mismo es parsear la plantilla “registrar_usuario.html” con todos sus campos vacíos.

Esta plantilla permite ingresar todos los campos de la tabla Usuario, excepto el Tipo de Usuario. Este dato quedará como NULL hasta que un Administrador active el usuario. Veremos en la Sección 4.2 que si este dato es NULL el usuario no podrá loguearse.

La plantilla contiene un Formulario con varios campos de texto, dos selectores para el Tipo de Documento y Sexo y, por último, un input tipo submit. Se declara el Formulario de tal manera que envíe sus datos al proceso “hacer_registrar_usuario.php” por el método POST tras recibir un pedido submit.
Además se incluye un enlace a la pantalla de login

hacer_registrar_usuario.php

Librerías utilizadas:
- template.php,
- conexion.php,
- editar_fecha.php,
- validacion.php

Este proceso recibe todos los datos del Formulario de registro de usuarios mediante el método POST, por lo cual la primera acción que realizamos es pasar todos los datos a variables internas.
El siguiente paso es validar todos los datos. De aquí surgen dos alternativas.

1. Validación Incorrecta: en este caso se vuelve a parsear la plantilla “registrar_usuario.html” pero mostrando en pantalla todos los datos que

habían sido ingresados por el usuario en la primera instancia. Además de ello se muestra un mensaje que indica cual es el error.
2. Validación Correcta: primero se guarda en la base de datos un usuario nuevo con todos sus campos; luego se parsea la plantilla “informacion.html” enviando el mensaje de que el Usuario fue registrado con éxito.

La plantilla “informacion.html” es bastante básica. Contiene una cabecera h2 con el mensaje que se desea enviar al usuario y un enlace variable. En este caso el enlace hace referencia a “index.php”.

Nuevamente, si se accede al enlace, como el usuario no realizó el login el sistema automáticamente redirecciona el navegador hacía “login.php”.

Reglas de Gestión:
- No pueden existir nombres de usuario repetidos.
- No pueden existir documentos duplicados.

Esto se cumple además para toda la Gestión de Usuarios.

4.2. INGRESO AL SISTEMA

login.php

Librerías utilizadas:
- template.php

Su función es parsear la plantilla “login.html”.

Esta plantilla contiene un Formulario con dos campos de texto, uno para ingresar el nombre de usuario y el otro para la contraseña. Se incluye también un input tipo submit. Se declara el Formulario de manera que envíe sus datos al proceso “validar_usuario.php” mediante el método POST tras un pedido submit.
Además se incluyen en esta plantilla un mensaje de error, que en este caso no se muestra y un enlace a “registrar_usuario.php”.

validar_usuario.php

Librerías utilizadas:
- template.php,
- conexion.php

Este proceso tiene gran importancia, aquí se verifica si el usuario se encuentra habilitado o no para ingresar al sistema.
Recordemos que le campo contraseña que se aloja en la Base esta encriptado bajo la función SHA1, por tal motivo, antes de compararlos tenemos que aplicar la misma función a la contraseña ingresada por el usuario. Luego las podemos compara.

Si el usuario o contraseña son inválidos, se parsea nuevamente “login.html”, pero ahora estará visible el campo error con el mensaje correspondiente.
Si el usuario y contraseña son correctos, entonces:

1. Se inicia una nueva Sesión. Esto mantendrá persistente la variable $_SESSION.
2. Se crea un campo en $_SESSION que identifica al usuario como logueado.
3. Se crea otro campo en $_SESSION que guarda el Nombre de Usuario ingresado.
4. Se creo otro campo en $_SESSION que guarda el Tipo de Usuario.
5. Se redirecciona el navegador hacía “index.php”.

4.3. MÓDULO DE AGENTES

index.php

Librerías utilizadas:

- seguridad.php,
- template.php,
- conexion.php,
- funciones.ajax.php

Su función es parsear la plantilla “index.html”. Para poder hacerlo se necesita hacer un algunas consultas sobre la Base de Datos.
Primero se solicitan todos los datos de la tabla Novedad. Esto genera un arreglo donde cada fila contendrá tres datos, IdNovedad, IdNorma e IdPauta.
Como luego se mostrará el título de la norma o nombre de la pauta según corresponda se deben efectuar otras consultas contra la base.
Se recorre el arreglo para todas las filas:

- Si IdNorma no es NULL entonces se consulta contra la tabla Norma el campo TituloNorma de ese IdNorma.
- Si IdPauta no es NULL entonces se consulta contra la tabla Pauta el campo NombrePauta de ese IdPauta.

Obtenidos estos datos se procede a parsear “index.html”.

Esta plantilla esta compuesta por cuatro contenedores: GENERAL, MEDIO, DIVMENU y DIVCONTENIDO.

- GENERAL esta compuesto por un encabezado, el contenedor MEDIO y el pie de la página.
- MEDIO esta compuesto por dos contenedores, a la izquierda DIVMENU y a la derecha DIVCONTENIDO.
- DIVMENU contiene una tabla estática con las opciones del menú principal.
- DIVCONTENIDO contiene una tabla dinámica donde se muestran las novedades que hayan al momento del ingreso.

DIVCONTENIDO.

Cada fila de la tabla tendrá un enlace hacía una función AJAX. Esta función será distinta dependiendo de si la novedad es una norma o una pauta.

- Norma: xajax_form_norma(IdNorma),
- Pauta: xajax_form_norma(IdPauta)

Ambas funciones se explicarán más adelante.

DIVMENU.
Como ya mencionamos aquí tendremos una tabla estática con diferentes enlaces que enunciamos a continuación.
Buscar Normas: xajax_form_buscar_norma(),
Buscar Pautas: xajax_form_buscar_pauta(),
Actualizar mis datos: xajax_form_mis_datos(),
Cambiar Contraseña: xajax_form_cambiar_password(),
Informes: enlace a “informe.php” (Ver Sección 4.4),
Administración: enlace a “admin.php” (Ver Sección 4.5),
Salir: enlace a “logout.php” (Ver Sección 4.6)

Funciones AJAX utilizadas.

xajax_form_norma(IdNorma): se explicará su funcionamiento más adelante, siguiendo la secuencia de buscar norma.

xajax_form_norma(IdPauta): se explicará su funcionamiento más adelante, siguiendo la secuencia de buscar pauta.

xajax_form_buscar_norma()

Su objetivo es mostrar en pantalla un formulario que permite al usuario definir un algún criterio para buscar una determinada norma.

Para ello se parsea la plantilla “form_buscar_norma.html” en el contenedor DIVCONTENIDO. Esta plantilla incluye una tabla con tres selectores (tema, título y nivel) y varios campos de texto en blanco.
El selector de temas al recibir un cambio llama automáticamente a la función xajax_recargar_titulo($id_tema), donde $id_tema es el id del tema seleccionado.

xajax_recargar_titulo($id_tema): su finalidad es reemplazar el selector de títulos por otro que tenga solo los Títulos de las normas cuyo IdTema coincida con el id del tema elegido por el usuario.

Además, la plantilla “form_buscar_norma.html”, cuenta con un enlace al index para volver a ver la pantalla principal y un botón “Buscar” que al ser presionado envía todos los datos ingresados por el usuario como argumento a la función xajax_resultado_normas(argumentos).

xajax_resultado_normas(argumentos): esta función tiene dos objetivos, uno es hacer una búsqueda de todas las normas donde los datos sean coincidentes con los

argumentos ingresados por el usuario. El segundo objetivo es mostrar estos resultados. Los mismos se muestran tras parsear con los datos obtenidos la plantilla “form_resultado_norma.html” en el contenedor MEDIO.

Esta plantilla muestra una taba dinámica con el número de la norma, título, nivel y fecha de todas las normas que hayan coincidido con los argumentos anteriores.
Además el título de la norma será un enlace a la función xajax_form_norma($id_norma).

xajax_form_norma($id_norma)

El objetivo de esta función es mostrar en pantalla todos los datos de la norma requerida por el usuario. En primera instancia se realiza una consulta contra la tabla Norma donde se busca que el IdNorma sea igual al id enviado como parámetro. Luego se parsea la plantilla "form_norma.html" con todos los datos de la norma y se lo envía al contenedor MEDIO.

Además de campos con texto se incluyen algunos enlaces:

- xajax_resultado_contactos_norma($id_norma): tiene como objetivo mostrar los contactos relacionados a la norma. Para ello se realizan dos consultas, primero se obtienen todos los IdContacto de la tabla ContactoNorma donde IdNorma=$id_norma. Luego se consultan todos los datos de la tabla Contacto donde IdContacto coincida con los IdContacto obtenidos en la consulta anterior. Para finalizar se parsean los resultados en la plantilla “form_resultado_contactos.html” y se lo envía al contenedor MEDIO. Esta plantilla es una simple tabla dinámica donde se muestran los contactos asociados y un enlace que nos lleva nuevamente a xajax_form_norma($id_norma).
- El segundo hace referencia a la url donde se aloja el documento de la norma.
- xajax_norma_pauta($id_norma): tiene como objetivo mostrar las pautas asociadas a la norma. Para ello se realizan dos consultas, primero se obtienen todos los IdPauta de la tabla NormaPauta donde IdNorma=$id_norma. Luego se consultan todos los datos de la tabla Pauta donde IdPauta coincida con los IdPauta obtenidos en la consulta anterior. Para finalizar se parsean estos datos en la plantilla "form_resultado_pauta.html" y se lo envía al contenedor MEDIO. Esta plantilla se describirá en más detalle más adelante.
- El último enlace nos lleva nuevamente al index.

xajax_form_buscar_pauta()

Su objetivo es mostrar en pantalla un formulario que permite al usuario definir un algún criterio para buscar una determinada pauta.

Para ello se parsea la plantilla “form_buscar_pauta.html” en el contenedor DIVCONTENIDO. Esta plantilla incluye una tabla con dos selectores (tema y pauta) y varios campos de texto en blanco.

El selector de temas al recibir un cambio llama automáticamente a la función xajax_recargar_subtema($id_tema), donde $id_tema es el id del tema seleccionado.

xajax_recargar_subtema($id_tema): su finalidad es reemplazar el selector de pautas por otro que tenga solo las pautas cuyo IdTema coincida con el id del tema elegido por el usuario.

Además, la plantilla “form_buscar_pauta.html”, cuenta con un enlace al index para volver a ver la pantalla principal y un botón “Buscar” que al ser presionado envía todos los datos ingresados por el usuario como argumento a la función xajax_resultado_pautas(argumentos).

xajax_resultado_pautas(argumentos): esta función tiene dos objetivos, uno es hacer una búsqueda de todas las pautas donde los datos sean coincidentes con los argumentos ingresados por el usuario. El segundo objetivo es mostrar estos resultados. Los mismos se muestran tras parsear con los datos obtenidos la plantilla “form_resultado_pauta.html” en el contenedor MEDIO.

Esta plantilla muestra una taba dinámica con la fecha, el nombre y el tema de todas las pautas que hayan coincidido con los argumentos anteriores.
Además el nombre de la pauta será un enlace a la función xajax_form_pauta($id_pauta).

xajax_form_pauta($id_pauta)

El objetivo de esta función es mostrar en pantalla todos los datos de la pauta requerida por el usuario. En primera instancia se realiza una consulta contra la tabla Pauta donde se busca que el IdPauta sea igual al id enviado como parámetro. En segunda instancia se buscan todos los ítem de la pauta en la tabla ItemPauta. Luego se parsea la plantilla "form_pauta.html" con todos los datos de la pauta en el contenedor MEDIO.

Además de campos con texto, tendremos un el selector de nivel, donde dependiendo el nivel seleccionado se muestra uno u otro ítem. Si la pauta tiene varios ítems para los distintos niveles se mostrará el ítem cuyo IdNivel sea el menor. Este selector reacciona al cambio de nivel invocando a la función xajax_recargar_items($id_pauta,$id_nivel).

xajax_recargar_items($id_pauta,$id_nivel): el objetivo de esta función es reemplazar el ítem que se esta mostrando en pantalla por el ítem correspondiente al Nivel seleccionado por el usuario.

Se incluyen además algunos enlaces:

- xajax_resultado_contactos_pauta($id_pauta): tiene como objetivo mostrar los contactos relacionados a la pauta. Para ello se realizan dos consultas, primero se obtienen todos los IdContacto de la tabla ContactoPauta donde IdPauta=$id_pauta. Luego se consultan todos los datos de la tabla Contacto donde IdContacto coincida con los IdContacto obtenidos en la consulta

anterior. Para finalizar se parsean los resultados en la plantilla “form_resultado_contactos.html” y se lo envía al contenedor MEDIO. Esta plantilla es una simple tabla dinámica donde se muestran los contactos asociados y un enlace que nos lleva nuevamente a xajax_form_pauta($id_pauta).

- El segundo hace referencia a la url donde se aloja el documento de la pauta.
- xajax_pauta_norma($id_pauta): tiene como objetivo mostrar las normas asociadas a la pauta. Para ello se realizan dos consultas, primero se obtienen todos los IdNorma de la tabla NormaPauta donde IdPauta=$id_pauta. Luego se consultan todos los datos de la tabla Norma donde IdNorma coincida con los IdNorma obtenidos en la consulta anterior. Para finalizar se parsean estos datos en la plantilla "form_resultado_pauta.html" y se lo envía al contenedor MEDIO. Esta plantilla se describió anteriormente (Ver Página 14).
- El último enlace nos lleva nuevamente al index.

xajax_form_mis_datos()

Su objetivo es mostrar en pantalla un formulario que permite al usuario modificar algunos de sus datos personales.

Para ello primero se recupera de la variable de Sesión el identificador del usuario que está conectado. El segundo paso es consultar en la tabla Usuario de la base de datos los datos que se van a permitir editar. Por último, con estos datos se parsea la plantilla "form_mis_datos.html" en el contenedor DIVCONTENIDO. Esta plantilla incluye una tabla con varios campos de texto editables con los datos actuales del Usuario.
Además de estos campos se incluyen un botón que permite enviar los datos ingresados por el usuario al enlace xajax_modificar_mis_datos(argumentos) y un enlace al index que cumple las veces de un botón cancelar, ya que no guardará ningún cambio.

xajax_modificar_mis_datos(argumentos): su principal objetivo es guardar los cambios que haya ingresado el usuario en la base de datos. Una vez guardados los cambios se parsea la plantilla “infoajax.html” también en el contenedor DIVCONTENIDO con el mensaje correspondiente.

xajax_form_cambiar_password()

Su objetivo es mostrar en pantalla un formulario que permite al usuario cambiar su contraseña.

Para ello se parsea la plantilla "form_cambiar_password.html" en el contenedor DIVCONTENIDO. Esta plantilla incluye una tabla con dos campos de texto, de esta forma se solicita al usuario que ingrese y repita la contraseña.
Además de estos campos se incluyen un botón que permite enviar los datos ingresados por el usuario al enlace xajax_cambiar_password($pass1,$pass2) y un enlace al index que cumple las veces de un botón cancelar, ya que no guardará ningún cambio.

xajax_cambiar_password($pass1,$pass2): su principal objetivo es guardar la nueva contraseña ingresada por el usuario en la base de datos. Recordemos que antes de guardar la contraseña hay que encriptarla mediante la función SHA1. Una vez guardados los cambios se parsea la plantilla "infoajax.html" también en el contenedor DIVCONTENIDO con el mensaje correspondiente.

4.4. MÓDULO DE INFORMES

informe.php

Librerías utilizadas:

- seguridad2.php,
- template.php,
- conexion.php,
- funciones.info.ajax.php

Su única función es parsear la plantilla "informe.html".

Esta plantilla esta compuesta por cuatro contenedores: GENERAL, MEDIO, DIVMENU y DIVCONTENIDO.

- GENERAL esta compuesto por un encabezado, el contenedor MEDIO y el pie de la página.
- MEDIO esta compuesto por dos contenedores, a la izquierda DIVMENU y a la derecha DIVCONTENIDO.
- DIVMENU contiene una tabla estática con las opciones del menú principal.
- DIVCONTENIDO es el espacio reservado para mostrar los resultados según la opción del menú elegida. En principio aparecerá vacío.

DIVMENU.
Como ya mencionamos aquí tendremos una tabla estática con diferentes enlaces que enunciamos a continuación.
Por Norma: xajax_form_buscar_norma(),
Por Pauta: xajax_form_buscar_pauta(),
Por Usuario: xajax_form_mis_datos(),
Volver: enlace al index

Funciones AJAX utilizadas.

xajax_form_buscar_norma()

Su objetivo es mostrar en pantalla un formulario que permite al usuario definir un algún criterio para buscar una determinada norma.

Para ello se parsea la plantilla "form_buscar_norma.html" en el contenedor DIVCONTENIDO. Esta plantilla incluye una tabla con tres selectores (tema, título y nivel) y varios campos de texto en blanco.

El selector de temas al recibir un cambio llama automáticamente a la función xajax_recargar_titulo($id_tema), donde $id_tema es el id del tema seleccionado.

xajax_recargar_titulo($id_tema): su finalidad es reemplazar el selector de títulos por otro que tenga solo los Títulos de las normas cuyo IdTema coincida con el id del tema elegido por el usuario.

Además, la plantilla “form_buscar_norma.html”, cuenta con un enlace a “informe.php” para volver a ver la pantalla principal del módulo Informes y un botón “Buscar” que al ser presionado envía todos los datos ingresados por el usuario como argumento a la función xajax_resultado_normas(argumentos).

xajax_resultado_normas(argumentos): esta función tiene dos objetivos, uno es hacer una búsqueda de todas las normas donde los datos sean coincidentes con los argumentos ingresados por el usuario. El segundo objetivo es mostrar estos resultados. Los mismos se muestran tras parsear con los datos obtenidos la plantilla “form_resultado_norma_info.html” en el contenedor MEDIO.

Esta plantilla muestra una taba dinámica con el número de la norma, título, nivel y fecha de todas las normas que hayan coincidido con los argumentos anteriores.
Además el título de la norma será un enlace a la función xajax_informe_norma($id_norma).

xajax_informe_norma($id_norma)

El objetivo de esta función es mostrar en pantalla los registros de que usuarios ingresaron a ver la norma elegida por el usuario y en que fecha lo hicieron. En primera instancia se realiza una consulta contra la tabla LogUsr donde se buscan todos los accesos a la norma en cuestión. Con estos datos se parsea la plantilla "informe_por_tabla.html" en el contenedor MEDIO.

Esta plantilla contendrá el título de la norma elegida y una tabla dinámica con los datos de todos los usuarios que ingresaron a ver esa norma y la fecha en que lo hicieron.
Además se incluyen dos enlaces:

- “generar_pdf_norma.php?pos=$id_norma”: se explica a continuación.
- El segundo hace referencia a “informe.php”.

generar_pdf_norma.php?pos=$id_norma

Librerías utilizadas:

- seguridad2.php,
- template.php,
- conexion.php,
- dompdf_config.inc.php

Su objetivo es generar un informe en formato PDF con los registros de que usuarios ingresaron a ver la norma que se envía como parámetro y en que fecha lo hicieron.

En primera instancia se realiza una consulta contra la tabla LogUsr donde se buscan todos los accesos a la norma en cuestión. Con estos datos se parsea la plantilla "informe_por_tabla_pdf.html" sólo que ahora en vez de enviarlo a un contenedor para mostrarlo en pantalla, hay que pasarlo como parámetro a una función de DOMPDF que se encargará de convertir el código HTML en formato PDF.

xajax_form_buscar_pauta()

Su objetivo es mostrar en pantalla un formulario que permite al usuario definir un algún criterio para buscar una determinada pauta.

Para ello se parsea la plantilla “form_buscar_pauta.html” en el contenedor DIVCONTENIDO. Esta plantilla incluye una tabla con dos selectores (tema y pauta) y varios campos de texto en blanco.
El selector de temas al recibir un cambio llama automáticamente a la función xajax_recargar_subtema($id_tema), donde $id_tema es el id del tema seleccionado.

xajax_recargar_subtema($id_tema): su finalidad es reemplazar el selector de pautas por otro que tenga solo las pautas cuyo IdTema coincida con el id del tema elegido por el usuario.

Además, la plantilla “form_buscar_pauta.html”, cuenta con un enlace a “informe.php” para volver a ver la pantalla principal del módulo informes y un botón “Buscar” que al ser presionado envía todos los datos ingresados por el usuario como argumento a la función xajax_resultado_pautas(argumentos).

xajax_resultado_pautas(argumentos): esta función tiene dos objetivos, uno es hacer una búsqueda de todas las pautas donde los datos sean coincidentes con los argumentos ingresados por el usuario. El segundo objetivo es mostrar estos resultados. Los mismos se muestran tras parsear con los datos obtenidos la plantilla “form_resultado_pauta.html” en el contenedor MEDIO.

Esta plantilla muestra una taba dinámica con la fecha, el nombre y el tema de todas las pautas que hayan coincidido con los argumentos anteriores.
Además el nombre de la pauta será un enlace a la función xajax_informe_pauta($id_pauta).

xajax_informe_pauta($id_pauta)

El objetivo de esta función es mostrar en pantalla los registros de que usuarios ingresaron a ver la pauta elegida por el usuario y en que fecha lo hicieron. En primera instancia se realiza una consulta contra la tabla LogUsr donde se buscan todos los accesos a la pauta en cuestión. Con estos datos se parsea la plantilla "informe_por_tabla.html" en el contenedor MEDIO.

Esta plantilla contendrá el nombre de la pauta elegida y una tabla dinámica con los datos de todos los usuarios que ingresaron a ver esa pauta y la fecha en que lo hicieron.

Además se incluyen dos enlaces:

- “generar_pdf_pauta.php?pos=$id_pauta”: se explica a continuación.
- El segundo hace referencia a “informe.php”.

generar_pdf_pauta.php?pos=$id_pauta

Librerías utilizadas:

- seguridad2.php,
- template.php,
- conexion.php,
- dompdf_config.inc.php

Su objetivo es generar un informe en formato PDF con los registros de que usuarios ingresaron a ver la pauta que se envía como parámetro y en que fecha lo hicieron. En primera instancia se realiza una consulta contra la tabla LogUsr donde se buscan todos los accesos a la pauta en cuestión. Con estos datos se parsea la plantilla "informe_por_tabla_pdf.html" sólo que ahora en vez de enviarlo a un contenedor para mostrarlo en pantalla, hay que pasarlo como parámetro a una función de DOMPDF que se encargará de convertir el código HTML en formato PDF.

xajax_form_lista_usuario_activo(1)

El usuario ingresa a esta opción con el objetivo de generar un informe de actividad de algún usuario en particular. Por tal motivo, la finalidad de esta función es que el usuario elija otro usuario a partir del cual se generará el informe de actividad.

Para lograr este objetivo, primero se consulta contra la base de datos la tabla Usuario donde IdTipoUsuario no es NULL. Con estos datos se parsea la plantilla “form_usuarios_activos_info.html” en el contenedor MEDIO.

Esta plantilla incluye una tabla dinámica con los datos de los usuarios activos y un input tipo radio que permite al usuario elegir sobre que usuario se hará el informe.
Además, en la parte superior de la tabla se incluyen dos campos de texto y un botón que al ser presionado filtrará los usuario según los datos ingresados. Esto lo hará invocando a la función xajax_filtrar_usuarios($username,$surname).

xajax_filtrar_usuarios($username,$surname): su objetivo es consultar contra la base de datos todos los usuarios que coincidan con el nombre de usuario y/o apellido ingresados por el usuario y que además cumplan con la clausula de estar activos. Con estos datos parsea la tabla con los datos del usuario y el input tipo radio. Luego reemplaza la tabla mostrada en pantalla por la nueva que se acaba de generar.

Al final de la tabla se agregan dos campos de texto para ingresar en período a partir del cual se generará el informe, un botón que al ser presionado invoca la función xajax_informe_usuario($fe_desde,$fe_hasta,$nick) y un enlace que hace referencia nuevamente a “informe.php”.

xajax_informe_usuario($fe_desde,$fe_hasta,$nick)

El objetivo de esta función es mostrar en pantalla los registros de que normas y pautas ha visitado el usuario elegido en el período indicado y en que fecha lo hizo. Si la fecha desde y/o fecha hasta son NULL entonces se genera el informe sin tener en cuenta la fecha de ingreso. En primera instancia se realiza una consulta contra la tabla LogUsr donde se buscan todos los accesos del usuario en cuestión donde la FeLog sea menor que $fe_hasta y FeLog sea mayor que $fe_desde. Con estos datos se parsea la plantilla "informe_por_usuario.html" en el contenedor MEDIO.

Esta plantilla contendrá el nombre del usuario elegido, si fue ingresado, el período utilizado para calcular los datos y una tabla dinámica con los datos de todas las normas a pautas a las que ingreso el usuario y en que fecha ingresó.
Además se incluyen dos enlaces:

- “generar_pdf_usuario.php?pos=$nick&des=$fe_desde&has=$fe_hasta”: se explica a continuación.
- El segundo hace referencia a “informe.php”.

generar_pdf_usuario.php?pos=$nick&des=$fe_desde&has=$fe_hasta

Librerías utilizadas:

- seguridad2.php,
- template.php,
- conexion.php,
- editar_fecha.php,
- dompdf_config.inc.php

Su objetivo es generar un informe en formato PDF con los registros de actividad de el usuario enviado como parámetro en el período pasado como parámetro. En primera instancia se realiza una consulta contra la tabla LogUsr donde se buscan todos los accesos del usuario a una norma o pauta en el período, si el mismo es NULL no lo se lo tiene en cuenta para la consulta, es decir, se mostrará el registro de actividad del usuario para todas las fechas posibles. Con estos datos se parsea la plantilla "informe_por_usuario_pdf.html" sólo que ahora en vez de enviarlo a un contenedor para mostrarlo en pantalla, hay que pasarlo como parámetro a una función de DOMPDF que se encargará de convertir el código HTML en formato PDF.

4.5. MÓDULO DE ADMINISTRACIÓN

admin.php

Librerías utilizadas:

- seguridad2.php,
- template.php,
- conexion.php,
- funciones.admin.ajax.php

Su única función es parsear la plantilla “admin.html”.

Esta plantilla esta compuesta por cuatro contenedores: GENERAL, MEDIO, DIVMENU y DIVCONTENIDO.

- GENERAL esta compuesto por un encabezado, el contenedor MEDIO y el pie de la página.
- MEDIO esta compuesto por dos contenedores, a la izquierda DIVMENU y a la derecha DIVCONTENIDO.
- DIVMENU contiene una tabla estática con las opciones del menú principal.
- DIVCONTENIDO es el espacio reservado para mostrar los resultados según la opción del menú elegida. En principio aparecerá vacío.

DIVMENU.
Como ya mencionamos aquí tendremos una tabla estática con diferentes enlaces que enunciamos a continuación.
Usuarios Activos: xajax_form_lista_usuario_activo(1),
Usuarios Inactivos: xajax_form_lista_usuario_inactivo(),
Agregar Norma: xajax_form_agregar_norma(),
Buscar Norma: xajax_form_buscar_norma(),
Agregar Pauta: xajax_form_agregar_pauta(),
Buscar Pauta: xajax_form_buscar_pauta(),
Temas: xajax_form_temas(1,null),
Agregar Contacto: xajax_form_agregar_contacto(false),
Buscar Contacto: xajax_form_buscar_contacto(1),
Volver: enlace al index

Funciones AJAX utilizadas.

xajax_form_lista_usuario_activo(1)

El objetivo de esta función es mostrar en pantalla un listado con todos los usuarios activos del sistema.

Para lograr este objetivo, primero se consulta contra la base de datos la tabla Usuario donde IdTipoUsuario no es NULL. Con estos datos se parsea la plantilla “form_usuarios_activos.html” en el contenedor MEDIO.

Esta plantilla incluye una tabla dinámica con los datos de los usuarios activos y tres enlaces que permiten al administrador:

- modificar un usuario: xajax_form_modificar_usuario($nick),
- desactivar un usuario: xajax_desactivar_usuario($nick),
- eliminar un usuario: xajax_eliminar_usuario($nick)

Además, en la parte superior de la tabla se incluyen dos campos de texto y un botón que al ser presionado filtrará los usuario según los datos ingresados. Esto lo hará invocando a la función xajax_filtrar_usuarios($username,$surname).
Al final se incluye un enlace que hace referencia a “admin.php”.

xajax_filtrar_usuarios($username,$surname): su objetivo es consultar contra la base de datos todos los usuarios que coincidan con el nombre de usuario y/o apellido ingresados por el usuario y que además cumplan con la clausula de estar activos. Con estos datos parsea la tabla con los datos del usuario y el input tipo radio. Luego reemplaza la tabla mostrada en pantalla por la nueva que se acaba de generar.

xajax_form_modificar_usuario($nick)

El objetivo de esta función es mostrar en pantalla un formulario con todos los datos del usuario elegido de tal manera que el administrador pueda editar los campos. En primera instancia consulta los datos contra la tabla Usuario donde Nick sea igual a $nick. Con estos datos parsea la plantilla "form_modificar_usuario.html" en el contenedor MEDIO.

Esta plantilla cuenta con un formulario varios campos de texto editables y dos selectores que permiten modificar el sexo, tipo de documento y tipo de usuario. El formulario esta definido de tal manera que envíe todos sus campos al enlace "hacer_modificar_usuario.php" mediante el método POST tras recibir un pedido vía submit.
Se incluye por lo tanto un input tipo submit para efectuar el envío de los datos y un enlace a “admin.php”.

"***hacer_modificar_usuario.php***"

Librerías utilizadas:

- template.php,
- conexion.php,
- editar_fecha.php,
- seguridad2.php,
- validacion.php,
- registrar_log.php,
- funciones.admin.ajax.php

Este proceso recibe los datos enviados del formulario anterior, por lo tanto su finalidad será guardar los cambios que haya registrado el administrador sobre los datos del usuario. Estos datos serán guardados en la tabla Usuario.

Luego nos mostrará una pantalla de información donde se indica si los datos fueron guardados con éxito o no. Esta pantalla tendrá un enlace a “admin.php” y otro al listado de usuarios activos.

xajax_desactivar_usuario($nick)

Esta función tiene un único propósito, y es desactivar el usuario que se envía como parámetro. Para hacer efectivo el movimiento realiza un SET contra la tabla Usuario guardando en el campo TipoUsuario el valor NULL.

Luego recarga el listado de usuarios activos para mostrar los cambios.

Reglas de Gestión:

- No se podrá desactivar a si mismo.

xajax_eliminar_usuario($nick)

Esta función tiene como objetivo eliminar el usuario enviado como parámetro. Para ello realiza un DELETE contra la tabla Usuario donde Nick=$nick.

Luego recarga el listado de usuarios activos para mostrar los cambios.

Reglas de Gestión:

- No se podrá eliminar a si mismo.

xajax_form_lista_usuario_inactivo()

El objetivo de esta función es mostrar en pantalla un listado con todos los usuarios inactivos del sistema.

Para lograr este objetivo, primero se consulta contra la base de datos la tabla Usuario donde IdTipoUsuario es NULL. Con estos datos se parsea la plantilla "form_usuarios_inactivos.html" en el contenedor DIVCONTENIDO.

Esta plantilla incluye una tabla dinámica con los datos de los usuarios inactivos y dos enlaces que permiten al administrador:

- activar un usuario: xajax_form_modificar_usuario($nick),
- eliminar un usuario: xajax_eliminar_usuario_inactivo($nick)

Al final de la tabla se incluye un enlace hacía "admin.php".

xajax_form_modificar_usuario($nick)

Esta función es la misma que la explicada en esta misma sección (Ver página 23). La única diferencia es que antes de dar el control a la función aparece un mensaje que indica al administrador que para activar el usuario debe seleccionar un tipo de usuario y luego guardar los cambios.

xajax_eliminar_usuario_inactivo($nick)

Esta función tiene como objetivo eliminar el usuario enviado como parámetro. Para ello realiza un DELETE contra la tabla Usuario donde Nick=$nick.

Luego muestra parsea la plantilla "info_ajax.html" con el mensaje correspondiente en el contenedor DIVCONTENIDO.

xajax_form_agregar_norma()

El objetivo de esta función es mostrar en pantalla un formulario para completar los datos de una nueva norma.

Para hacerlo se parsea la plantilla “form_agregar_norma.html” en el contenedor MEDIO.

Esta plantilla cuenta con un formulario definido para que tras un pedido submit envíe sus campos al proceso “guardar_norma.php” mediante el método POST.
Dentro del formulario se incluyen todos los campos necesarios para crear una nueva norma y por supuesto un input tipo submit que permite enviar el formulario.

guardar_norma.php

Librerías utilizadas:

- template.php,
- editar_fecha.php,
- conexion.php,
- seguridad2.php,
- registrar_log.php,
- funciones.admin.ajax.php

Este proceso recibe todos los datos del formulario anterior, y por lo tanto, será el encargado de guardar los datos de la nueva norma. Para hacerlo realiza un INSERT contra la tabla Norma.

Entre los datos del formulario se recibirá, o no, el documento que debe quedar asociado a la Norma. El mismo se encontrará en la variable global HTML $_FILES. Aquí se utilizan funciones nativas de PHP con varios propósitos:

- is_uploaded_file($_FILES): nos indica si el navegador envió un archivo.
- copy($_FILES,$nombre): copia el archivo enviado en el destino $nombre.

En $nombre se indica la ruta y el nombre que tendrá el archivo en el servidor.

En el caso de las normas los archivos serán guardados en “./docs/normas/” y se les asignará el nombre “norma$id_norma.pdf”.

Si no se recibe ningún archivo, en el campo DocNorma de la tabla Norma se registra como NULL.

Aquí surgen dos alternativas, si la norma se pudo crear, entonces parsea la plantilla “info_opcion.html” con el mensaje correspondiente. Caso contrario vuelve a parsear la plantilla “form_agregar_norma.html” con el mensaje que indica porque no se guardo la nueva norma.

Reglas de Gestión:

- Dos Normas no pueden tener el mismo Número y Año.
- Si la Norma no tiene Número, entonces no podrá tener el mismo Título y año.

xajax_form_buscar_norma()

Su objetivo es mostrar en pantalla un formulario que permite al usuario definir un algún criterio para buscar una determinada norma.

Para ello se parsea la plantilla “form_buscar_norma.html” en el contenedor DIVCONTENIDO. Esta plantilla incluye una tabla con tres selectores (tema, título y nivel) y varios campos de texto en blanco.
El selector de temas al recibir un cambio llama automáticamente a la función xajax_recargar_titulo($id_tema), donde $id_tema es el id del tema seleccionado.

xajax_recargar_titulo($id_tema): su finalidad es reemplazar el selector de títulos por otro que tenga solo los Títulos de las normas cuyo IdTema coincida con el id del tema elegido por el usuario.

Además, la plantilla “form_buscar_norma.html”, cuenta con un enlace a “admin.php” para volver a ver la pantalla principal del módulo de administración y un botón “Buscar” que al ser presionado envía todos los datos ingresados por el usuario como argumento a la función xajax_resultado_normas(argumentos).

xajax_resultado_normas(argumentos): esta función tiene dos objetivos, uno es hacer una búsqueda de todas las normas donde los datos sean coincidentes con los argumentos ingresados por el usuario. El segundo objetivo es mostrar estos resultados. Los mismos se muestran tras parsear con los datos obtenidos la plantilla “form_resultado_norma.html” en el contenedor MEDIO.

Esta plantilla muestra una taba dinámica con el número de la norma, título, nivel y fecha de todas las normas que hayan coincidido con los argumentos anteriores.
Además el título de la norma será un enlace a la función xajax_form_norma($id_norma).

xajax_form_norma($id_norma)

El objetivo de esta función es mostrar en pantalla todos los datos de la norma requerida por el administrador. En primera instancia se realiza una consulta contra la tabla Norma donde se busca que el IdNorma sea igual al id enviado como parámetro. Luego se parsea la plantilla "form_norma_admin.html" con todos los datos de la norma y se lo envía al contenedor MEDIO.

Esta plantilla muestra un formulario con todos los datos de la norma en campos editables, de manera que el administrador pueda modificar algunos de sus datos. El formulario esta definido para que envíe sus datos al proceso “modificar_norma.php” tras un pedido submit.

modificar_norma.php: la función de este proceso es guardar los cambios que se hayan efectuado sobre los datos de la norma. Hecho esto nos muestra el mensaje que corresponda tras parsear la plantilla “info_opcion.html”. El manejo de archivos es idéntico al descripto en el proceso “guardar_norma.php”.

Además de estos campos se incluyen algunos enlaces:

- xajax_novedad_norma($id_norma),
- xajax_contacto_norma($id_norma),
- xajax_norma_pauta($id_norma),
- xajax_eliminar_norma($id_norma),
- El último enlace nos lleva nuevamente a “admin.php”.

xajax_novedad_norma($id_norma)

Su objetivo es permitir al administrador incluir esta norma en novedades.
Primero se consultan todos los datos de la tabla Novedad y con estos datos se parsea la plantilla "resultado_novedad_norma_admin.html" en el contenedor MEDIO.

Esta plantilla muestra por un lado las novedades que haya al momento de la consulta presentando un enlace para eliminar cada novedad y por otro lado tiene un enlace que permite incluir la norma a novedades.

xajax_reiniciar_novedad_norma($idnovedad, $idnorma): su principal objetivo es eliminar la novedad que coincida con $idnovedad. La novedad se elimina tras setear en la tabla Novedad IdNorma e IdPauta como NULL. El dato $idnorma lo utiliza para volver a invocar la función xajax_novedad_norma($idnorma).

xajax_asociar_novedad_norma($idnorma,$idnovedad): su objetivo es incluir la norma en el número de novedad indicado. Para ello hay que setear en la tabla Novedad IdNorma=$idnorma donde IdNovedad sea igual a $idnovedad.
Luego se vuelve a invocar la función xajax_novedad_norma($idnorma) para recargar la pantalla.

xajax_contacto_norma($id_norma)

Su objetivo es permitir al administrador asociar un contacto a una norma.
Primero se consultan contra la base los datos de todos los contactos asociados a la norma en cuestión. Luego, con los datos obtenidos se parsea la plantilla "resultado_contacto_norma_admin.html" en el contenedor MEDIO.

Esta plantilla cuenta con una tabla dinámica, donde se muestran todos los contactos asociados a la norma y un enlace que permite eliminar esta asociación.
Además de ello cuenta con un selector donde se puede elegir un contacto y un botón que permite asociar el contacto a la norma.

xajax_desasociar_contacto_norma($id_contacto,$id_norma): realiza un DELETE de todos los datos contra la tabla ContactoNorma donde IdContacto sea igual a $id_contacto y IdNorma sea igual a $id_norma. Luego vuelve a invocar la función xajax_contacto_norma($id_norma) para recargar la pantalla.

xajax_asociar_contacto_norma($id_norma,$id_contacto): realiza un INSERT sobre la tabla ContactoNorma con el contacto elegido para la norma en cuestión. Luego

vuelve a invocar la función xajax_contacto_norma($id_norma) para recargar la pantalla.

xajax_norma_pauta($id_norma)

Su objetivo es permitir al administrador asociar una pauta a la norma enviada como parámetro.
Primero se consultan contra la base los datos de todas las pautas asociadas a la norma en cuestión. Luego, con los datos obtenidos se parsea la plantilla "form_resultado_pauta_admin.html" en el contenedor MEDIO.

Esta plantilla cuenta con una tabla dinámica, donde se muestran todas las pautas asociadas a la norma y un enlace que permite eliminar esta asociación.
Además de ello cuenta con dos selectores donde se puede hacer un filtrado por tema y luego elegir una pauta, y por último, un botón que permite asociar la pauta a la norma.

xajax_desasociar_norma_pauta($id_pauta,$id_norma): realiza un DELETE de todos los datos contra la tabla NormaPauta donde IdPauta sea igual a $id_pauta y IdNorma sea igual a $id_norma. Luego vuelve a invocar la función xajax_norma_pauta($id_norma) para recargar la pantalla.

xajax_recargar_subtema($id_tema): su función es recargar el selector de pautas según el tema elegido.

xajax_asociar_norma_pauta($id_norma,$id_pauta): realiza un INSERT sobre la tabla NormaPauta con la pauta elegida para la norma en cuestión. Luego vuelve a invocar la función xajax_norma_pauta($id_norma) para recargar la pantalla.

xajax_eliminar_norma($id_norma)

Su objetivo es eliminar la norma que se esta visualizando. Antes de eliminar la norma se verifica que no tenga pautas asociadas, contactos asociados y que no este en novedades. Si esto se cumple se hace el DELETE contra la tabla Norma donde IdNorma sea igual a $id_norma.
Si se pudo eliminar se parsea “infoajax.html” con el mensaje correspondiente en el contenedor MEDIO.

xajax_form_agregar_pauta()

El objetivo de esta función es mostrar en pantalla un formulario para completar los datos de una nueva pauta.
Para hacerlo se parsea la plantilla “form_agregar_pauta.html” en el contenedor MEDIO.

Esta plantilla cuenta con un formulario definido para que tras un pedido submit envíe sus campos al proceso “guardar_pauta.php” mediante el método POST.

Dentro del formulario se incluyen todos los campos necesarios para crear una nueva pauta y por supuesto un input tipo submit que permite enviar el formulario.

guardar_pauta.php

Librerías utilizadas:

- template.php,
- editar_fecha.php,
- conexion.php,
- seguridad2.php,
- registrar_log.php,
- funciones.admin.ajax.php

Este proceso recibe todos los datos del formulario anterior, y por lo tanto, será el encargado de guardar los datos de la nueva pauta. Para hacerlo realiza un INSERT contra la tabla Pauta.
Entre los datos del formulario se recibirá, o no, el documento que debe quedar asociado a la Pauta. El mismo se encontrará en la variable global HTML $_FILES. Aquí se utilizan funciones nativas de PHP con varios propósitos:

- is_uploaded_file($_FILES): nos indica si el navegador envió un archivo.
- copy($_FILES,$nombre): copia el archivo enviado en el destino $nombre.

En $nombre se indica la ruta y el nombre que tendrá el archivo en el servidor.

En el caso de las pautas los archivos serán guardados en “./docs/pautas/” y se les asignará el nombre “pauta$id_pauta.pdf”.

Si no se recibe ningún archivo, en el campo DocNorma de la tabla Norma se registra como NULL.

Aquí surgen dos alternativas, si la pauta se pudo crear, entonces parsea la plantilla “info_opcion.html” con el mensaje correspondiente. Caso contrario vuelve a parsear la plantilla “form_agregar_pauta.html” con el mensaje que indica porque no se guardo la nueva pauta.

Reglas de Gestión:

- Dos Pautas no pueden tener el mismo Nombre y Tema.

xajax_form_buscar_pauta()

Su objetivo es mostrar en pantalla un formulario que permite al usuario definir un algún criterio para buscar una determinada pauta.

Para ello se parsea la plantilla “form_buscar_pauta.html” en el contenedor DIVCONTENIDO. Esta plantilla incluye una tabla con dos selectores (tema y pauta) y varios campos de texto en blanco.
El selector de temas al recibir un cambio llama automáticamente a la función xajax_recargar_subtema($id_tema), donde $id_tema es el id del tema seleccionado.

xajax_recargar_subtema($id_tema): su finalidad es reemplazar el selector de pautas por otro que tenga solo las pautas cuyo IdTema coincida con el id del tema elegido por el usuario.

Además, la plantilla “form_buscar_pauta.html”, cuenta con un enlace al index para volver a ver la pantalla principal y un botón “Buscar” que al ser presionado envía todos los datos ingresados por el usuario como argumento a la función xajax_resultado_pautas(argumentos).

xajax_resultado_pautas(argumentos): esta función tiene dos objetivos, uno es hacer una búsqueda de todas las pautas donde los datos sean coincidentes con los argumentos ingresados por el usuario. El segundo objetivo es mostrar estos resultados. Los mismos se muestran tras parsear con los datos obtenidos la plantilla “form_resultado_pauta.html” en el contenedor MEDIO.

Esta plantilla muestra una taba dinámica con la fecha, el nombre y el tema de todas las pautas que hayan coincidido con los argumentos anteriores.
Además el nombre de la pauta será un enlace a la función xajax_form_pauta($id_pauta).

xajax_form_pauta($id_pauta)

El objetivo de esta función es mostrar en pantalla todos los datos de la pauta requerida por el administrador. En primera instancia se realiza una consulta contra la tabla Pauta donde se busca que el IdPauta sea igual al id enviado como parámetro. Luego se parsea la plantilla "form_pauta_admin.html" con todos los datos de la pauta y se lo envía al contenedor MEDIO.

Esta plantilla muestra un formulario con todos los datos de la pauta en campos editables, de manera que el administrador pueda modificar algunos de sus datos. El formulario esta definido para que envíe sus datos al proceso “modificar_pauta.php” tras un pedido submit.

modificar_pauta.php: la función de este proceso es guardar los cambios que se hayan efectuado sobre los datos de la pauta. Hecho esto nos muestra el mensaje que corresponda tras parsear la plantilla “info_opcion.html”. El manejo de archivos es idéntico al descripto en el proceso “guardar_pauta.php”.

Además de estos campos se incluyen algunos enlaces:

- xajax_novedad_pauta($id_pauta),
- xajax_form_items_pauta($id_pauta),
- xajax_contacto_pauta($id_pauta),
- xajax_pauta_norma($id_pauta),
- xajax_eliminar_pauta($id_pauta),
- El último enlace nos lleva nuevamente a “admin.php”.

xajax_novedad_pauta($id_pauta)

Su objetivo es permitir al administrador incluir esta pauta en novedades.
Primero se consultan todos los datos de la tabla Novedad y con estos datos se parsea la plantilla "resultado_novedad_pauta_admin.html" en el contenedor MEDIO.

Esta plantilla muestra por un lado las novedades que haya al momento de la consulta presentando un enlace para eliminar cada novedad y por otro lado tiene un enlace que permite incluir la pauta a novedades.

xajax_reiniciar_novedad_pauta($id_novedad, $id_pauta): su principal objetivo es eliminar la novedad que coincida con $id_novedad. La novedad se elimina tras setear en la tabla Novedad IdNorma e IdPauta como NULL. El dato $id_pauta se utiliza para volver a invocar la función xajax_novedad_pauta($id_pauta).

xajax_asociar_novedad_pauta($id_pauta,$id_novedad): su objetivo es incluir la pauta en el número de novedad indicado. Para ello hay que setear en la tabla Novedad IdPauta=$id_pauta donde IdNovedad sea igual a $id_novedad.
Luego se vuelve a invocar la función xajax_novedad_pauta($id_pauta) para recargar la pantalla.

xajax_form_items_pauta($id_pauta)

Su objetivo es permitir al administrador agregar, modificar o eliminar un ítem asociado a esta pauta. Por lo tanto primero se consulta contra la tabla ItemPauta todos los ítems asociados a la pauta. Con estos datos se parsea la plantilla "form_itempauta_admin.html" en el contenedor MEDIO.

Esta plantilla esta dividida en dos paneles, el superior que permite modificar o eliminar un ítem existente y el inferior que permite agregar un nuevo ítem.

Panel Superior: Tiene un selector para elegir el nivel, al cambiar el nivel se recargan los datos del ítem mostrado. Los datos que se muestran son editables. Por otro lado se tiene un botón que permite modificar el ítem y otro que permite eliminarlo.

Panel Inferior: Tiene un selector para elegir el nivel del nuevo ítem y dos campos que permiten ingresar la descripción y las observaciones del ítem. Por otro lado se tiene un botón que permite guardar el nuevo ítem.

xajax_recargar_items($id_pauta,$id_nivel): su función es recargar los datos del ítem que se esta mostrando en pantalla por los datos del ítem asociado al nivel seleccionado por el administrador.

xajax_modificar_item_pauta($id_pauta,$id_nivel,$descrip,$obs): realiza un UPDATE contra la tabla ItemPauta de los datos Descripcion y Observacion por $descrip y $obs respectivamente donde IdNivel sea igual a $id_nivel e IdPauta sea igual a $id_pauta. Luego invoca a la función xajax_form_items_pauta($id_pauta) para recargar la pantalla.

xajax_eliminar_item_pauta($id_pauta,$id_nivel): su objetivo principal es hacer un DELETE contra la tabla ItemPauta donde IdPauta sea igual a $id_pauta e IdNivel sea igual a $id_nivel. Luego invoca a la función xajax_form_items_pauta($id_pauta) para recargar la pantalla.

xajax_agregar_item_pauta($id_pauta,$id_nivel,$descrip,$obs): el objetivo principal de esta función es hacer un INSERT contra la tabla ItemPauta de todos los datos enviados como parámetros.
Luego invoca a la función xajax_form_items_pauta($id_pauta) para recargar la pantalla.

xajax_contacto_pauta($id_pauta)

Su objetivo es permitir al administrador asociar un contacto a una pauta.
Primero se consultan contra la base los datos de todos los contactos asociados a la pauta en cuestión. Luego, con los datos obtenidos se parsea la plantilla "resultado_contacto_pauta_admin.html" en el contenedor MEDIO.

Esta plantilla cuenta con una tabla dinámica, donde se muestran todos los contactos asociados a la pauta y un enlace que permite eliminar esta asociación.
Además de ello cuenta con un selector donde se puede elegir un contacto y un botón que permite asociar el contacto a la pauta.

xajax_desasociar_contacto_pauta($id_contacto,$id_pauta): realiza un DELETE de todos los datos contra la tabla ContactoPauta donde IdContacto sea igual a $id_contacto y IdPauta sea igual a $id_pauta. Luego vuelve a invocar la función xajax_contacto_pauta($id_pauta) para recargar la pantalla.

xajax_asociar_contacto_pauta($id_pauta,$id_contacto): realiza un INSERT sobre la tabla ContactoPauta con el contacto elegido para la pauta en cuestión. Luego vuelve a invocar la función xajax_contacto_pauta($id_pauta) para recargar la pantalla.

xajax_pauta_norma($id_pauta)

Su objetivo es permitir al administrador asociar una norma a la pauta enviada como parámetro.
Primero se consultan contra la base los datos de todas las normas asociadas a la pauta en cuestión. Luego, con los datos obtenidos se parsea la plantilla "form_resultado_norma_admin.html" en el contenedor MEDIO.

Esta plantilla cuenta con una tabla dinámica, donde se muestran todas las normas asociadas a la pauta y un enlace que permite eliminar esta asociación.
Además de ello cuenta con dos selectores donde se puede hacer un filtrado por tema y luego elegir una norma, y por último, un botón que permite asociar la norma a la pauta.

xajax_desasociar_pauta_norma($id_pauta,$id_norma): realiza un DELETE de todos los datos contra la tabla NormaPauta donde IdPauta sea igual a $id_pauta y IdNorma sea igual a $id_norma. Luego vuelve a invocar la función xajax_pauta_norma($id_pauta) para recargar la pantalla.

xajax_recargar_subtema($id_tema): su función es recargar el selector de normas según el tema elegido.

xajax_asociar_pauta_norma($id_pauta,$id_norma): realiza un INSERT sobre la tabla NormaPauta con la norma elegida para la pauta en cuestión. Luego vuelve a invocar la función xajax_pauta_norma($id_pauta) para recargar la pantalla.

xajax_eliminar_pauta($id_pauta)

Su objetivo es eliminar la pauta que se esta visualizando. Antes de eliminar la pauta se verifica que no tenga normas asociadas, ítems asociados, contactos asociados y que no este en novedades. Si esto se cumple se hace el DELETE contra la tabla Pauta donde IdPauta sea igual a $id_pauta.
Si se pudo eliminar se parsea “infoajax.html” con el mensaje correspondiente en el contenedor MEDIO.

xajax_form_temas($pagina,$nuevo_tema)

Su objetivo es mostrar en pantalla un formulario que permitirá al administrador realizar altas y bajas de temas.
Para poder mostrar todos los temas ya registrados primero se realiza una consulta contra la tabla Tema. Con estos datos se parsea la plantilla "form_temas_admin.html" en el contenedor MEDIO.

Esta plantilla cuenta en la parte superior con dos campos de texto y dos botones, uno tiene como objetivo el alta de un tema y el otro tiene como objetivo hacer una búsqueda según la palabra ingresada por el administrador.
Luego de estos campos se muestra una tabla dinámica con los datos de los 10 primeros temas temas existentes y un enlace por cada tema que permitirá eliminarlo.
Al final de la tabla se incluye un enlace que permite cambiar de página para ver más temas. Cuando se ingresa por primera vez $pagina vale 1, entonces mostrará la primera página.
Además, la plantilla permite enviar un mensaje al usuario indicando que se creo un nuevo tema. De todas formas cuando se ingresa por primera vez $nuevo_tema es NULL, entonces no se mostrará el mensaje.

xajax_agregar_tema($nuevo_tema): realiza un INSERT contra la tabla Tema según el tema ingresado por el administrador. Luego invoca a la función xajax_form_temas(1,$nuevo_tema) para recargar la pantalla con el mensaje de que se ha creado un nuevo tema. No pueden existir dos temas con el mismo Nombre.

xajax_filtrar_temas($palabra): su objetivo es filtrar el listado de temas según la palabra ingresada por el administrador. Luego de realizada la consulta correspondiente recarga únicamente la tabla que muestra los temas en la pantalla actual con la lista de temas que devuelve la consulta.

xajax_eliminar_tema($id_tema): realiza un DELETE contra la tabla Tema donde IdTema sea igual a $id_tema. Luego invoca a la función xajax_form_temas(1,NULL) para recargar la pantalla.

xajax_form_agregar_contacto($nom_contacto)

El objetivo de esta función es mostrar en pantalla el formulario para efectuar altas de contactos. Para ello se parsea la plantilla "form_agregar_contacto.html" en el contenedor DIVCONTENIDO.

Esta plantilla cuenta con algunos campos de texto que permiten al administrador ingresar los datos necesarios para dar de alta un contacto. Luego de estos campos se incluye un botón que permite agregar el nuevo contacto.
Además de esto, la plantilla permite enviar un mensaje al administrador cuando un nuevo usuario es ingresado. Cuando se la invoca por primera vez $nom_contacto es FALSE y por tal motivo no se mostrará ningún mensaje.

xajax_guardar_contacto($nombre,$email,$tel,$ctx): esta función realiza un INSERT contra la tabla Contacto con los datos enviados como parámetros. Luego invoca a la función xajax_form_agregar_contacto($nombre) lo cual recargará la misma pantalla pero con el mensaje de que el usuario fue creado. No se pueden guardar dos contactos con el mismo nombre.

xajax_form_buscar_contacto($página)

Su objetivo es mostrar en pantalla un formulario que permitirá al administrador realizar modificaciones y bajas de temas.
Para poder mostrar todos los contactos ya registrados primero se realiza una consulta contra la tabla Contacto. Con estos datos se parsea la plantilla "form_buscar_contacto_admin.html" en el contenedor MEDIO.

Esta plantilla cuenta en la parte superior un campos de texto y un botón que tiene como objetivo hacer una búsqueda según la palabra ingresada por el administrador. Luego de estos campos se muestra una tabla dinámica con los datos de los 10 primeros temas contactos existentes y dos enlaces por cada tema que permitirán eliminar el contacto o modificarlo.
Al final de la tabla se incluye un enlace que permite cambiar de página para ver más temas. Cuando se ingresa por primera vez $pagina vale 1, entonces mostrará la primera página.

xajax_filtrar_contactos($palabra): su objetivo es filtrar el listado de contactos según la palabra ingresada por el administrador. Luego de realizada la consulta

correspondiente recarga únicamente la tabla que muestra los contactos en la pantalla actual con la lista de temas que devuelve la consulta.

xajax_eliminar_contacto($id_contacto): realiza un DELETE contra la tabla Contacto donde IdContacto sea igual a $id_contacto. Luego invoca a la función xajax_form_buscar_contacto(1) para recargar la pantalla.

xajax_form_modificar_contacto($id_contacto)

Su objetivo es parsear la plantilla "form_modificar_contacto.html". Para hacerlo primero deberá consultar contra la tabla Contacto todos los datos donde IdContacto sea igual a $id_contacto.

Esta plantilla muestra todos los datos del contacto en campos editables y contiene al final un botón que permite guardar los cambios.

modificar_contacto($id_contacto,$nombre,$email,$tel,$ctx): realiza un UPDATE de todos los datos de la tabla Contacto donde IdContacto sea igual a $id_contacto. Muestra un mensaje de que el contacto fue modificado y luego invoca a la función xajax_form_buscar_contacto(1) para volver a mostrar el listado de contactos.

4.6. SALIR DEL SISTEMA

logout.php

Librerías utilizadas:

- seguridad.php,

Su única función es cerrar la sesión del usuario, para ello limpia los datos guardados en la variable de sesión en el proceso de login. Luego redirecciona el navegador hacía “login.php”.

5. Reportes

A continuación se detallan cuales son las tablas y los campos que se consultaron al momento de realizar los reportes y que proceso los genera. Luego se muestra un ejemplo de las distintas salidas en pantalla.

5.1. POR NORMA

Listado de Agentes que visitaron una Norma

- Tablas consultadas: LogUsr, Norma y Usuario.
- Campos consultados: FeLog (LogUsr), Usuario (LogUsr), TituloNorma (Norma), Nombre (Usuario) y Apellido (Usuario).
- Proceso: "generar_pdf_norma.php?pos=$id_norma".

Sistema de Gestión de Normas y Pautas - Informes

Lista de Usuarios que visitaron esta Norma

Norma: Calendario Escolar 2011

Usuario	Apellido	Nombre	Fecha de Ingreso
root	root	root	2011-07-11
root	root	root	2011-07-11
agilli	Gilli	Andrés	2011-07-19

Dirección General de Planificación y Coordinación de Proyectos
Sistema Interno

Ilustración 1: Informe de visitas por Norma

5.2. POR PAUTA

Listado de Agentes que visitaron una Pauta

- Tablas consultadas: LogUsr, Pauta y Usuario.
- Campos consultados: FeLog (LogUsr), Usuario (LogUsr), NombrePauta (Pauta), Nombre (Usuario) y Apellido (Usuario).
- Proceso: "generar_pdf_pauta.php?pos=$id_pauta".

Sistema de Gestión de Normas y Pautas - Informes

Lista de Usuarios que visitaron esta Pauta

Pauta: Sigae - FAQs

Usuario	Apellido	Nombre	Fecha de Ingreso
agilli	Gilli	Andrés	2011-07-08
root	root	root	2011-07-16
root	root	root	2011-07-16
cgilli	Gilli	Carlos	2011-07-24
root	root	root	2011-07-26
agilli	Gilli	Andrés	2011-08-03

Dirección General de Planificación y Coordinación de Proyectos
Sistema Interno

Ilustración 2: Informe de visitas por Pauta

5.3. POR USUARIO

Listado de Normas y Pautas visitadas por un Usuario en un período

- Tablas consultadas: LogUsr, Norma, Pauta y Tema.
- Campos consultados: FeLog (LogUsr), IdTabla (LogUsr), IdRegistro (LogUsr), DatoObservado (LogUsr), IdTema (Norma o de Pauta según corresponda), Tema (Tema).
- Proceso:

"generar_pdf_usuario.php?pos=$nick&des=$fe_desde&has=$fe_hasta".

Sistema de Gestión de Normas y Pautas - Informes

Lista de Registros que visitó el Usuario agilli

Período: desde el día 01/08/2011 hasta el día 05/08/2011

Tabla	Dato	Tema	Fecha de Ingreso
Norma	Anexo I Decreto 516/10	Concurso Asistentes Escolares	2011-08-03
Norma	Anexo I Decreto 516/10	Concurso Asistentes Escolares	2011-08-03
Norma	Resolución 1470	Concurso Docentes	2011-08-03
Norma	Resolución 1470	Concurso Docentes	2011-08-03
Pauta	Examenes	Concurso Asistentes Escolares	2011-08-02
Pauta	Sigae - FAQs	Sistema de Gestión Escolar	2011-08-03
Pauta	Examenes	Concurso Asistentes Escolares	2011-08-03
Pauta	Examenes	Concurso Asistentes Escolares	2011-08-03

Dirección General de Planificación y Coordinación de Proyectos
Sistema Interno

Ilustración 3: Informe de Actividad de un Usuario

Se anexan a este manual los tres ejemplos de reportes en formato PDF.