



SISTEMA DE AUTOMATIZACIÓN DEL TRATAMIENTO DE INFORMACIÓN DE HISTORIAS CLÍNICAS Y MEDICAMENTOS EN EL SUBCENTRO DE SALUD DE SAN ANTONIO DE IBARRA

Desarrollado por: *Egdo. Jorge Aníbal IpiALES IpiALES*

Telf: 062 933310; **Cel** 097292992

Email: jipiales2006@hotmail.com; j_ipiales2005@yahoo.com

UTN-2011

RESUMEN

Para la realización de este proyecto se tuvo que modelar primeramente el diagrama entidad-relación, ayudado de una herramienta libre MysqlWorkBench, la cual facilitó este diseño, pues se trabaja en forma visual, una vez que tuvimos este modelo se tuvo que conseguir herramientas para la programación, por lo que se optó por Netbeans (versión 6.9.1) y con symfony 1.4, un framework que ayuda en gran manera la programación con PHP, la misma que ayudado con Doctrine, en la actualidad es el ORM por defecto para symfony desde la versión 1.4, aceleran el diseño proyectos web.

El conocimiento de todas estas herramientas, a más de conocimientos básicos de JavaScript, hojas de estilo CSS, ajax y html ayudaron a la culminación del presente proyecto, para facilitar el trabajo del departamento de Estadística en el Subcentro de Salud de San Antonio de Ibarra.

SUMMARY

For the realization of this project was to model first the entity-relationship diagram, a free tool helped MysqlWorkBench, which facilitated this design, because it works visually, once this model we had to get tools programming, so it was decided to Netbeans (version 6.9.1) and symfony 1.4, a framework which helps greatly with PHP programming, helped with the same Doctrine, is now the default ORM for symfony from version 1.4, accelerate web design projects.

The knowledge of these tools, more than basic knowledge of JavaScript, CSS, ajax and html helped the completion of this project to facilitate the work of the Department of Statistics in the Health Sub-center San Antonio de Ibarra.

INTRODUCCIÓN

A medida que los requerimientos de automatización de información en las entidades públicas o privadas aumentan, la tecnología en la programación debe ir a la par con estos requerimientos, pues en la actualidad la programación web ha ido acrecentando conjuntamente con la popularidad del Internet, por lo que se necesita aumentar los conocimientos de herramientas de diseño de sistemas para la web.

Uno de los inconvenientes que tienen los diseñadores, se refiere a la adquisición herramientas para lograr este objetivo, por lo que una gran alternativa que se tiene es utilizar herramientas



libres, las cuales podemos bajarnos desde la web, pues existe una diversidad de estas, las cuales las podemos utilizar sin ningún pago monetario y su rendimiento no difiere en gran medida con las herramientas pagadas.

La creación de herramientas que en la actualidad es bastante difundida como son los frameworks ha venido a agilizar el proceso de desarrollo de software para la web, este es el caso de Symfony, pues existe una buena documentación en la red y blogs de discusión los cuales a su vez ayudan para una buena comprensión de esta herramienta.

En este documento vamos a indicar los procesos necesarios para el desarrollo de aplicaciones para la web utilizando symfony y ayudado de netbeans, en este caso para aplicar al desarrollo del proyecto para la web en el subcentro de salud “San Antonio” con un servidor web local WampServer 2i.

MATERIALES Y MÉTODOS

En el presente trabajo, primeramente tuvimos que diseñar el diagrama entidad relación ayudado de MySQLWorkBench (versión 5.1.18), la misma que es gratuita y que tiene un ambiente visual ya mignable como lo mostramos en la siguiente figura.

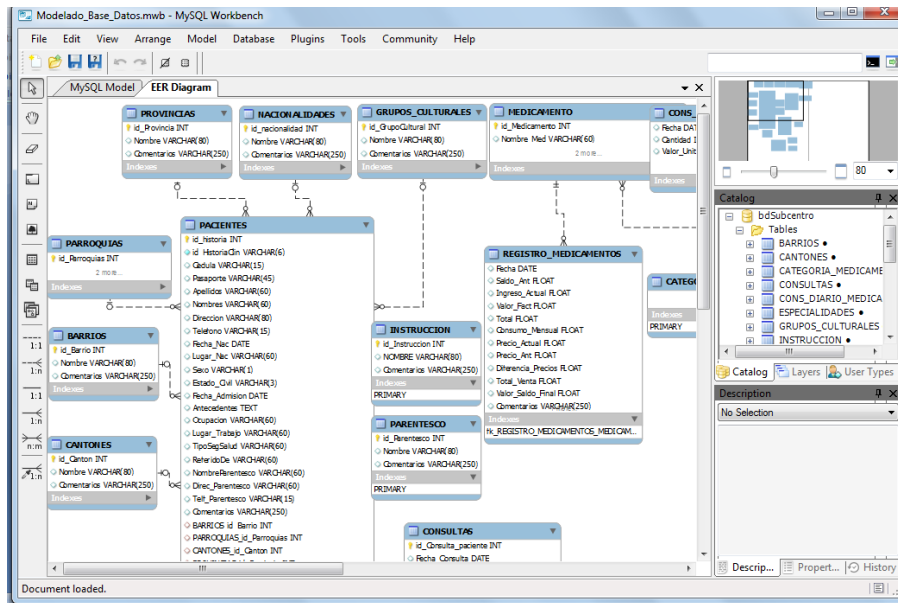


Figura1: Diseño del diagrama entidad relación con MySQLWorkBench.

Después de diseñar el esquema óptimo de la base de datos para la presente aplicación, el diagrama entidad relación óptimo, quedó de la siguiente manera:

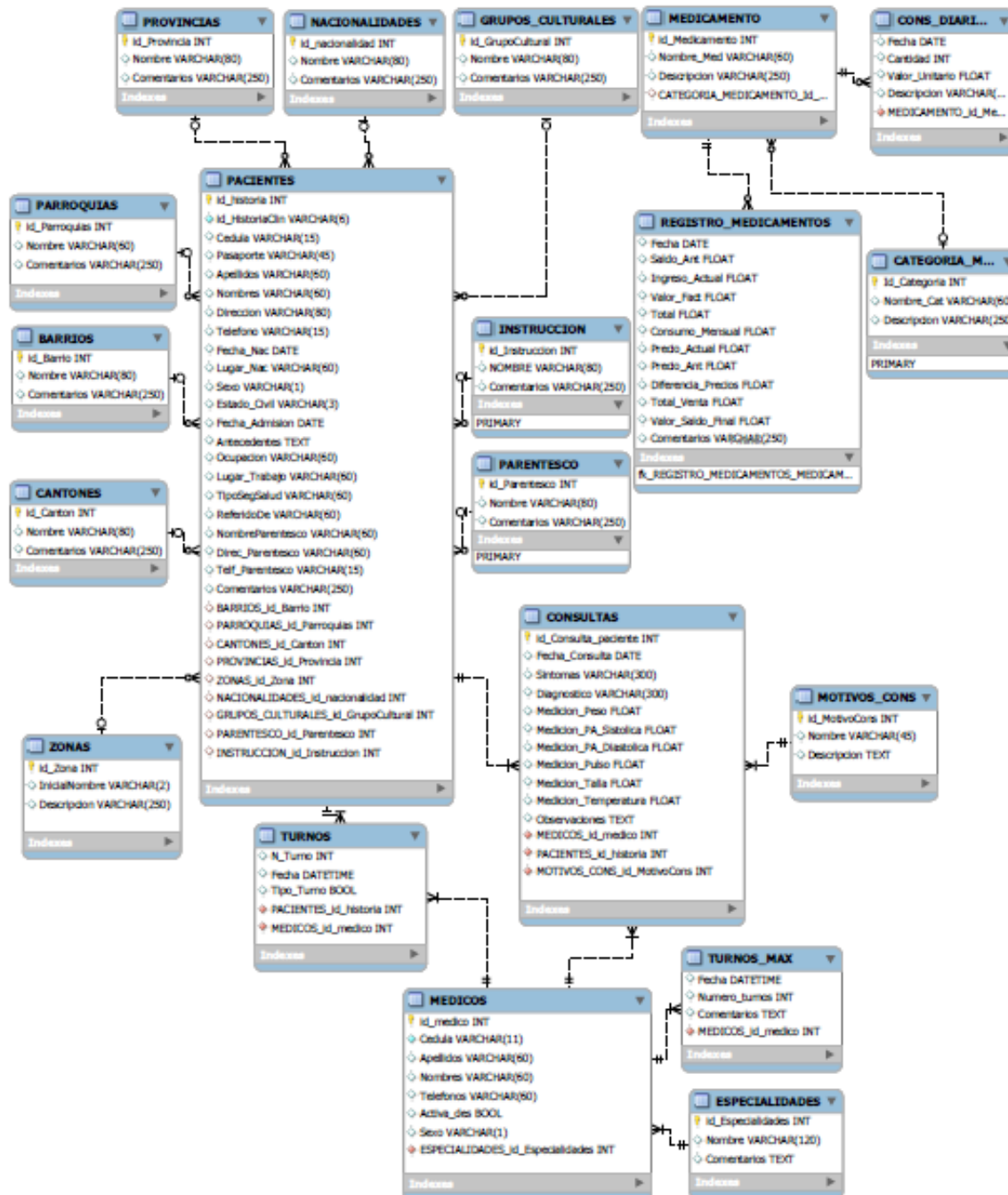


Figura2: Diagrama entidad-relación de la base de datos para el sistema del subcentro de salud San Antonio.

Las tablas que se diseñaron para el sistema del subcentro de salud san Antonio que se muestran en la figura 2, fueron diseñadas en la herramienta que citamos anteriormente.

Este diagrama fue la base para poder llevar este mismo esquema a la base de datos física al motor de Base de datos Mysql, la misma que se instala conjuntamente con la aplicación Wamp 2i.



Para que funcione la aplicación, debemos ejecutar primeramente nuestro servidor web, que en nuestro caso es el WampServer 2i, por lo que debemos instalarlo instalado, en caso contrario no funcionará, por lo que para instalarlo primeramente se debe descargar de internet (<http://www.wampserver.com/en/>), si lo tenemos instalado debemos ver un icono como el siguiente:



Figura 3: Ícono representativo de WampServer 2i

CONFIGURACIÓN DE SYMFONY EN NETBEANS.

Como ya dijimos en la introducción, nuestras herramientas de desarrollo van a ser Symfony ayudado de Netbeans para facilitar la programación con este framework y hacerlo en forma visual.

Symfony se basa en el patrón **MVC (Modelo Vista Controlador)**, es decir, la estructura lógica de los proyectos creados se divide en:

- **Aplicaciones:** es decir nuestro proyecto,
- **Clases:** las clases los tipos de funcionalidades de las que consta nuestra aplicación (gestión de usuarios, gestión de noticias, gestión de comentarios, etc.) y,
- **Módulos:** que es donde se detalla cómo hacer cada funcionalidad de la clase a la que están asignados (alta, borrado y edición de usuarios, etc.).

Ahora nos dedicaremos a configurar Netbeans para trabajar con Symfony, de la siguiente manera:

En primer lugar debemos tener un “LAMP”¹ (Linux-Apache-MySQL- PHP/Python/PERL) instalado en nuestro equipo, en nuestro caso utilizamos WAMP SERVER 2.0.

Luego descargamos la última versión estable de Symfony desde su web, para el inicio de nuestro proyecto la última versión estable fue Symfony 1.4.

Una vez descargado el paquete que normalmente viene empaquetado, lo descomprimos en el directorio de nuestra web, como nosotros trabajamos de forma local, creamos una carpeta en la unidad C:/, de la siguiente manera: “C:\SubcentroSalud\sistema\”, entonces la dirección completa donde lo descomprimos va a ser algo así: C:\SubcentroSalud\sistema\lib\vendor\symfony

En nuestro siguiente paso abrimos Netbeans, para luego crear un nuevo proyecto PHP, el asistente nos guiará paso a paso, en donde nosotros solo daremos el nombre a nuestro proyecto y localización de nuestros archivos del proyecto, pero en el paso donde debemos elegir el

¹ Conjunto de aplicaciones para configurar sitios web o servidores dinámicos con un esfuerzo reducido.



framework (PHP frameworks), obviamente elegiremos Symfony. Al seleccionarlo nos aparecerán varias opciones de configuración. Tendremos que pulsar sobre Options, a la derecha de la ventana, como se indica en el **Figura 4**:

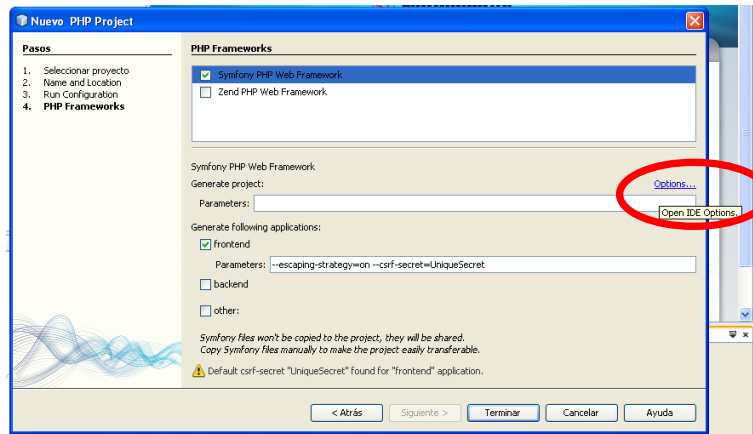


Figura 4. Localización de Options para configurar Symfony en Netbeans

Luego de hacer clic en Options..., se nos abrirá el panel de configuración de PHP, en esta ventana hacemos clic en la pestaña de PHP en donde indicaremos la ruta donde se encuentra el ejecutable de PHP del LAMP que hayamos instalado, en nuestro caso se encuentra en la siguiente ruta:

C:\wamp\bin\php\php5.3.0\php.exe

Además, en esta misma ventana tendremos que indicar la ruta de instalación de Symfony pulsando el botón **Add Folder**, que será la carpeta que hemos descomprimido en el directorio de nuestro proyecto

C:\SubcentroSalud\sistema\lib\vendor\symfony

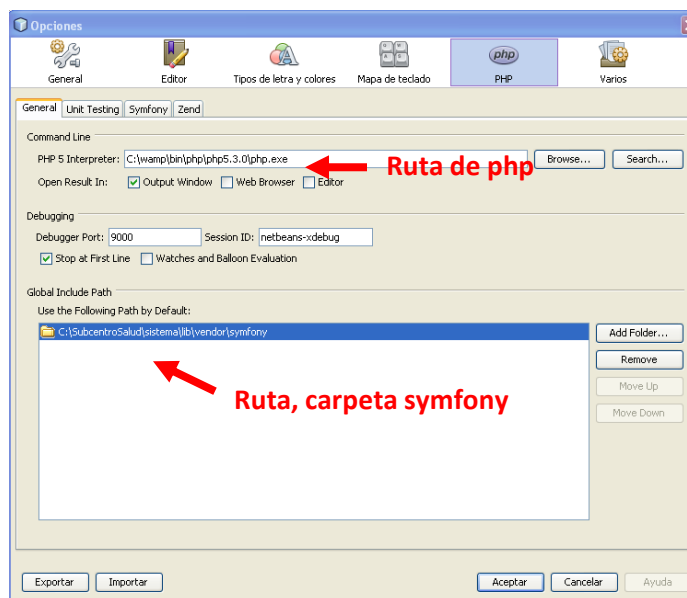


Figura 5. Indicar Ruta de Php y carpeta Symfony en el proyecto



Ahora, en la pestaña Symfony, de esta misma pantalla tendremos que indicar la ruta del fichero binario del framework, la cual en nuestro caso la encontramos en la siguiente dirección:

C:\SubcentroSalud\sistema\lib\vendor\symfony\data\bin\symfony

Esto lo indicamos en el siguiente gráfico:

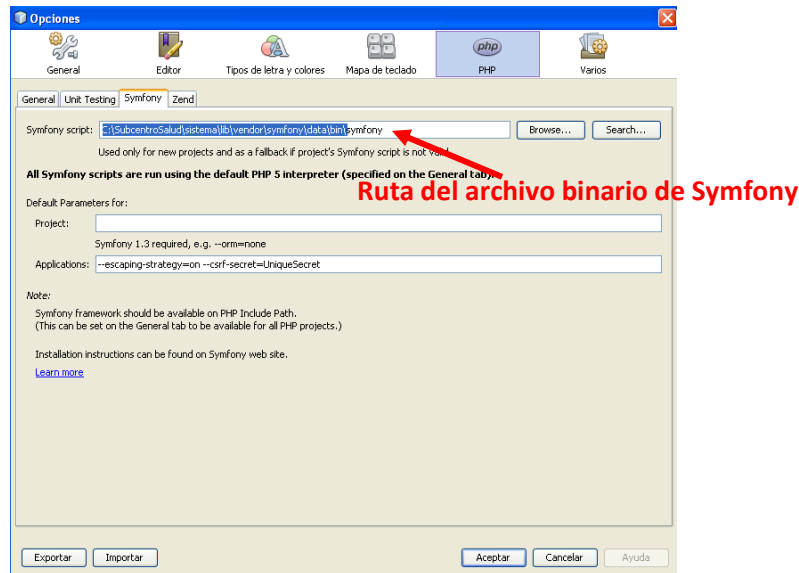


Figura 6. Localización de archivo binario de Symfony

Y por último, hacemos clic en **Aceptar**, pero antes de terminar, seleccionamos el checkbox **Backend**, en donde debe quedar con un visto como lo indicamos en el siguiente gráfico, esto con el fin de que nuestro proyecto maneje la base de datos, luego de esto si pulsamos en **Terminar**.

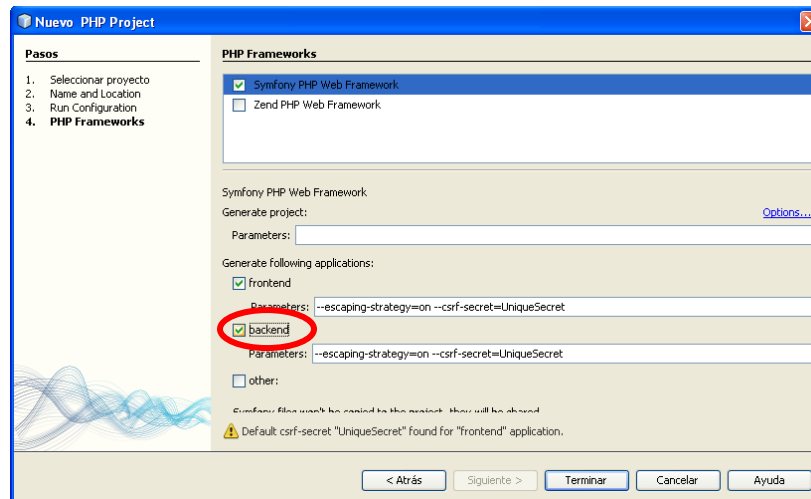


Figura 7. Localización de backend para que funcione la BBDD.

Si no hay ningún inconveniente en el transcurso de todos los pasos que nos guió el asistente, ya tendremos configurado e instalado nuestro proyecto **Symfony** en **NetBeans**.



CONFIGURANDO EL SERVIDOR WEB APACHE

Como ya solucionamos lo anterior, debemos configurar primeramente el host que en este caso nuestro host será local y virtual, de tal manera que se ejecute bajo otro nombre de host y en otro puerto (en este caso el puerto 80).

En internet encontramos varios ejemplos de cómo crear un host virtual para ejecutar el proyecto de forma local, aquí tenemos un ejemplo:

```
NameVirtualHost *:80
<VirtualHost *:80>

ServerName miApp
DocumentRoot "C:\Users\Juanjo\Documents\NetBeansProjects\miApp\web"
DirectoryIndex index.php
<Directory "C:\Users\Juanjo\Documents\NetBeansProjects\miApp\web">
AllowOverride All
Allow from All
</Directory>
```

#solo para la barra de depuración:

```
Alias /sf "C:\symfony-1.4.11\data\web\sف"
<Directory "C:\symfony-1.4.11\data\web\sف">
AllowOverride All
Allow from All
</Directory>
</VirtualHost>
```

Lo que vamos a hacer nosotros es coger cualquiera de estos códigos y editarlo para acoplarlo a nuestro proyecto, hay que tomar en consideración que nosotros vamos a utilizar Symfony como framework para programar con Php, es por eso que esta configuración hace referencia a esta carpeta en donde tendremos nuestro Symfony, lo que nos quedaría de la siguiente manera:

```
# Esta es la configuración de Sistema para Subcentro de Salud
#Listen 127.0.0.1:80

<VirtualHost 127.0.0.1:80>
ServerName sistema
DocumentRoot "C:\SubcentroSalud\sistema\web"
DirectoryIndex index.php
Alias /sf "C:\SubcentroSalud\sistema\lib\vendor\symfony\data\web\sف"
<Directory "C:\SubcentroSalud\sistema\lib\vendor\symfony\data\web\sف">
AllowOverride All
Allow from All
</Directory>
<Directory "C:\SubcentroSalud\sistema\web">
```



```
AllowOverride All
Allow from All
</Directory>
</VirtualHost>
```

Como ya están establecidas por Symfony las rutas de la carpeta del proyecto, nos queda seguir los siguientes pasos:

1. Copiar el contenido (Ctrl+C) del código anterior (del VirtualHost),
2. Abrir el archivo `httpd.conf` que se encuentra en la siguiente ruta: `C:\wamp\bin\apache\Apache2.2.11\conf`,
3. Lo pegamos (Ctrl+V), normalmente en la parte final del contenido de este archivo,
4. Guardamos,
5. Y por último reiniciamos los servicios de Apache.

Si todo ha salido bien, y si hemos configurado correctamente Apache, abrimos el navegador y anotamos la siguiente dirección: <http://localhost:80>, pero también lo podemos hacer desde Netbeans, digitando la tecla F6 para ejecutar el proyecto. Y de esta manera nos debe salir una pantalla similar a la siguiente:

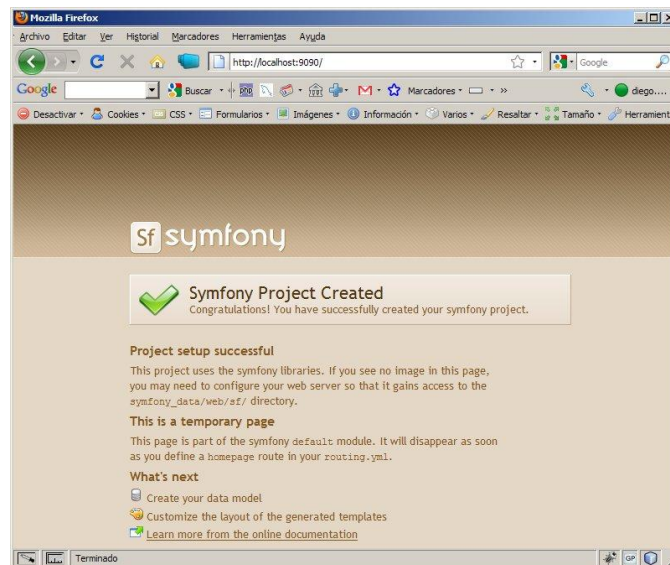


Figura 8. Pantalla de Symfony al concluir configuraciones básicas

Todo lo anterior es para correr la aplicación como un servidor virtual, por lo tanto se debe haber agregado el dominio virtual al archivo **hosts** de nuestro sistema operativo, para esto voy a la dirección `C:\Windows\System32\drivers\etc\hosts` y agrego:

```
127.0.0.1 localhost
127.0.0.1 sistema
```

Por lo que para entrar a nuestro proyecto solo digitaremos **`http://sistema`** en nuestro navegador (Mozilla Firefox 4 o superior de preferencia).





EL SISTEMA

Después de haber realizado las configuraciones necesarias para que nuestra aplicación pueda funcionar, vamos a analizar la estructura de nuestro proyecto, el mismo que fue realizado con la ayuda de Netbeans 6.9.1., la última versión estable hasta la fecha de iniciación de nuestro proyecto.

Para empezar a desarrollar proyectos con Symfony, hay que tener un conocimiento por lo menos básico de HTML, CSS, con esta forma de programación aprenderemos a ser un poco más ordenados en relación a lo que tiene que ver al lugar donde van a ir código de programación.

Pues hay veces que utilizaremos solo código HTML, en otros casos deberíamos utilizar solo PHP o una mezcla de ellos, además hacer referencia a archivos CSS para poder dar estilos a nuestras páginas.

Nos parece importante entender como es la estructura del código Symfony. Todo proyecto creado por Symfony consta de una serie de carpetas y archivos que contienen el código base del framework. El árbol de carpetas de un proyecto es:

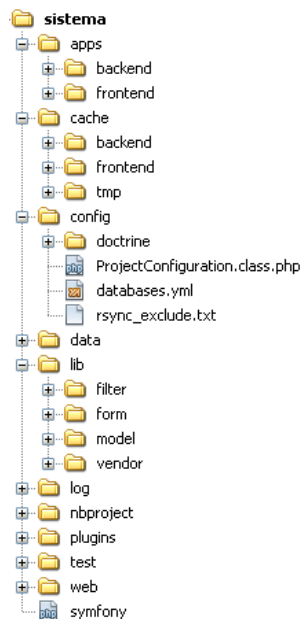


Figura 9. Estructura de archivos del proyecto symfony

Además, un proyecto creado con symfony están divididos en:

- **Aplicaciones:** los mismos que pueden tener varios módulos, normalmente pueden haber dos aplicaciones Backend y Frontend.
- **Módulos:** Estos a su vez suelen coincidir con las tablas de la base de datos creada según sea el propósito del proyecto.

Después que tengamos la base de datos y tengamos claro como es la estructura de un proyecto symfony, podremos insertar nuestro código donde crea necesario.



Para esto tenemos que crear una carpeta en donde vamos a crear todas las carpetas y también copiar la carpeta Symfony descargada desde su sitio oficial, en nuestro caso vamos a ayudarnos de una herramienta "Netbeans":

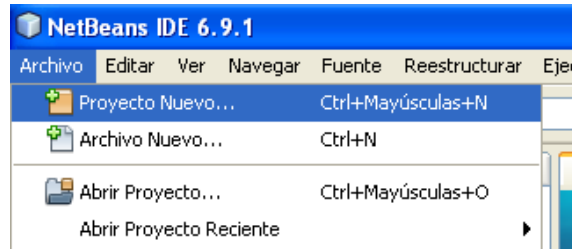


Figura 10. Crear un proyecto nuevo en symfony con Netbeans.

Esta forma de crear el proyecto reemplaza al siguiente código:

```
symfony generate:project sistema
```

Cualquiera sea la forma en la que creamos nuestro proyecto, nos habrá creado todo el árbol de directorios y archivos necesarios para poner en marcha nuestra web con este framework.

Luego creamos nuestros módulos, pero si utilizamos Netbeans como es en nuestro caso al crear nuestro proyecto también ya se crea estas dos aplicaciones por defecto.

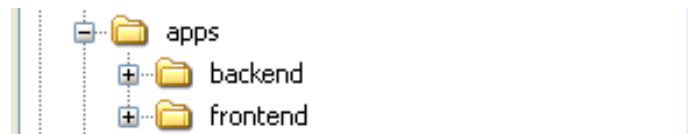


Figura 11. Estructura de carpetas de la aplicación.

En cambio si no utiliza Netbeans, tendrá que crearlos manualmente, aplicando un código similar al siguiente:

- `symfony generate:app --escaping-strategy=on --csrf-secret=UniqueSecret frontend`
- `symfony generate:app --escaping-strategy=on --csrf-secret=UniqueSecret backend`

Al llamar a la tarea `generate:app`, también hemos pasado dos opciones relacionadas con la seguridad:

- `-escaping-strategy`: Permite escapar la salida para evitar ataques XSS
- `-csrf-secret`: Permite tokens de sesión en los formularios para prevenir los ataques CSRF

El siguiente paso, será conectar nuestra aplicación con la base de datos, para ello, Symfony dispone de un archivo de configuración **databases.yml** en el que le indicaremos donde está localizada, a que base de datos y con qué usuario y contraseña lo deberá hacer.

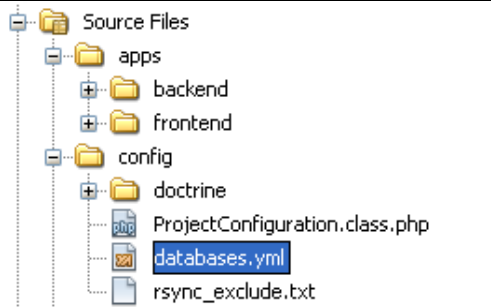


Figura 12. Localización del archivo de configuración de la base de datos para el proyecto.

Este archivo normalmente contiene el siguiente un código similar al siguiente:

```
all:
doctrine:
class: sfDoctrineDatabase
param:
dsn: 'mysql:host=localhost;dbname=bdbblog'
username: root
password: null
```

Con esto, ya tenemos configurado nuestro proyecto con la base de datos. Ahora le decimos a Doctrine que genere el archivo *schema.yml* a partir de la base de datos:

```
symfony doctrine:build-schema
```

Luego creamos el modelo, para que se genere clases para manejar la base de datos mediante objetos, de la siguiente manera:

```
symfony doctrine:build-model
```

De una manera similar crearemos con doctrine, los filtros, los formularios, etc. Para no realizar de manera separada todas estas acciones, utilizaremos un solo comando, de la siguiente manera:

```
Symfony doctrine:build -all
```

Solo que con el comando anterior hay que tener cuidado si es que ya tiene creado la base de datos anteriormente, ya que esta es una forma de resetear la base de datos y dejarla en blanco, o los vuelve a reconstruir.

Como hicimos notar anteriormente, nosotros utilizamos Netbeans, por lo que estas líneas de código lo podemos ejecutar de forma visual.

Si todo salió bien, ya podemos irnos con nuestro navegador a ver la pantalla de Symfony en donde nos dice que el proyecto ha sido creado, para eso visitamos la dirección **http:// sistema/**, de esta forma:



Figura 13. Pantalla inicial de un proyecto symfony creado.

De aquí en adelante no es más que crear módulos en nuestras aplicaciones como es el caso de Backend y Frontend, además de insertar código para manipular y controlar nuestro objetivo en el sistema.

NOTA: La forma de ejecutar un comando de symfony es sencillo, solo hacemos clic derecho sobre el nombre de la aplicación en la que estamos trabajando y nos aparecerá una ventana como se indica en la **figura 14:**

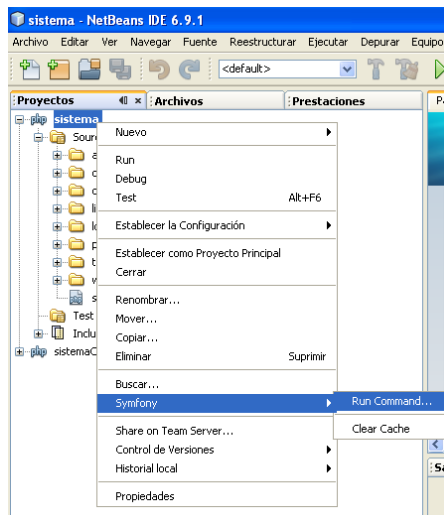


Figura 14: Ejecutar comandos Symfony en Netbeans.

Por lo que hacemos clic izquierdo en la opción *Run Command...* del menú desplegable como se indica en la **figura 14.**

ESTRUCTURA DE ARCHIVOS DEL PROYECTO SYMFONY

En el presente proyecto tenemos dos aplicaciones, Backend y Frontend:

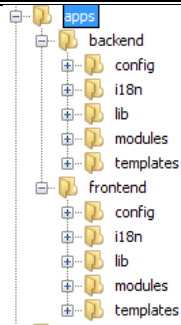


Figura 15: Estructura de archivos symfony (aplicación).

FRONTEND: En la **Figura 15** podemos observar visualmente que esta aplicación está formada por otros directorios que cuando se crea la aplicación Symfony con Netbeans las crea.

BACKEND: Esta aplicación tiene una estructura similar a la anterior como lo podemos observar en la **Figura 15**, solo que más adelante podremos ver que en esta aplicación es donde se invirtió mayor esfuerzo en programación y dedicación de tiempo.

ESTRUCTURA DE LA APLICACIÓN

La aplicación contiene varios subdirectorios como podemos visualizar en la **figura 15**, estas son:

Subdirectorio config:

La carpeta que contiene archivos yaml para configurar la aplicación, estas la podemos observar en la siguiente figura:

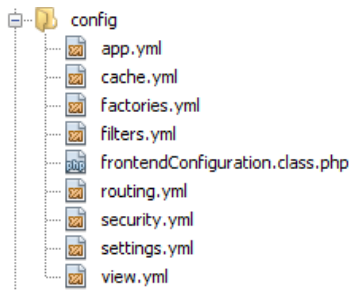


Figura 16. Contenido de la carpeta config.

Como podemos observar, existen archivos yaml, para configurar la aplicación (app.yml), para configurar la cache (cache.yml), los objetos básicos que necesita el marco de trabajo durante la vida de cualquier solicitud (factories.yml), para configurar los filtros (filters.yml), para configurar las formas de ingresar a las diferentes páginas llamadas rutas (routing.yml), para asegurar o no la aplicación y que solo personas autorizadas puedan ingresar (security.yml), para configurar los diferentes entornos de ejecución del sistema (settings.yml) y por supuesto para configurar las vistas (view.yml). Todos estos archivos ya vienen con su propio código que fue generado en la creación de la aplicación.





Subdirectorio l18n:

En esta carpeta guardaremos archivos de configuración de idiomas para las aplicaciones, en el caso de la aplicación frontend, no tiene ningún archivo de configuración para esta situación, pero para el caso de la aplicación Backend si la tenemos, este archivo se localiza en la dirección que se muestra en la **Figura 17**.



Figura 17. Contenido de la carpeta l18n.

Como podemos observar, son archivos xml, en donde si lo abrimos existen traducciones al español para los mensajes más comunes que normalmente se muestran en inglés.

Subdirectorio lib:

Esta carpeta contiene las librerías útiles para la aplicación, que pueden ser creadas por el mismo framework o por el programador para crear o utilizar sus propias librerías, normalmente viene con un solo archivo que se crea por defecto al crear nuestra aplicación, como se muestra en la siguiente figura:



Figura 18. Contenido de la carpeta lib.

Subdirectorio modules:

Este directorio es el que contiene los módulos que se van a crear para el sistema, los mismos que pueden ser creados visualmente ayudado de netbeans, como se muestra en la siguiente figura:

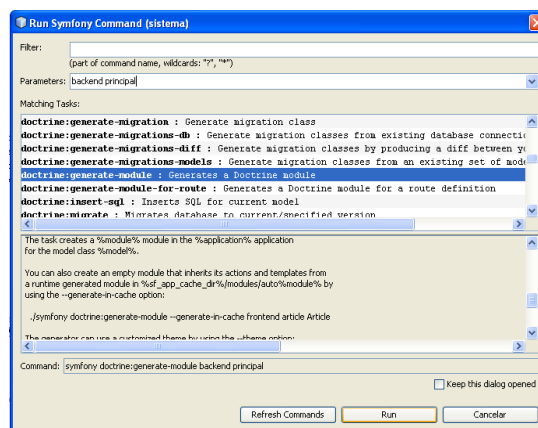




Figura 19. Pantalla para crear módulos con Netbeans

En la aplicación **FrontEnd**, solo tenemos un solo módulo en donde ingresamos información básica del Subcentro (Nosotros) y de la aplicación (Acerca de..), el mismo que se llama **principal**.

ESTRUCTURA DE LOS MÓDULOS

En cada módulo symfony crea por defecto subdirectorios al ejecutar el comando para crear módulos que en forma visual ayudado de netbeans se lo puede localizar en la siguiente figura:

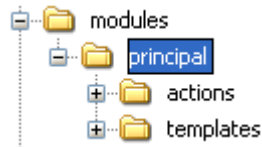


Figura 20. Estructura de módulos.

Subdirectorio actions:

En donde van a tener las funciones necesarias para ejecutar alguna acción, al ejecutar estas acciones devuelven el resultado al layout respectivo, por lo que para cada acción tenemos un archivo de visualización (layout) en el subdirectorio templates del subdirectorio del módulo como podemos observar en la figura 15, la sintaxis del nombre de la acción es como sigue:

public o protected execute + [nombre_acción]+([argumentos])

Por ejemplo *public executeIndex(sfWebRequest \$request)*, todas estas funciones están en el archivo *actions.class.php* las mismas que pueden ser públicas (public) o privadas (protected) dentro del directorio del módulo que estemos viendo.

Para el caso del sistema del Subcentro de Salud, generamos los módulos con el comando de symfony doctrine:admin-generator, por lo que se crea automáticamente otros subdirectorios a más de los que visualizamos en la figura 19, esto lo vamos a mostrar en la figura 20.

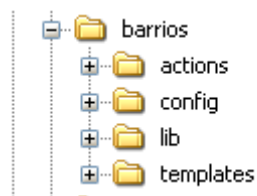


Figura 21: Estructura de módulos creados con admin_generator.

Como podemos observar en la **figura 21**, a más de las dos carpetas que se pudo visualizar en la **figura 20**, tenemos una carpeta en donde contendrá archivos de configuración (config) y una carpeta en donde se pueda guardar las librerías propias del módulo (lib).



Pero cada carpeta contiene archivos, que pueden ser con extensión `.yml` para configuraciones normalmente, o `.class.php` en donde podemos introducir nuestras propias funciones o manipular las funciones que por defecto crea symfony y archivos `.php`, para parciales o para las vistas del módulo, algunas de estas las podemos observar en la siguiente figura:

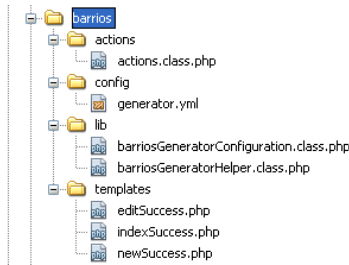


Figura 22. Estructura de directorios y subdirectorios de módulos creados con `admin-generator`.

Subdirectorio `config`:

El subdirectorio `config` de un módulo, inicialmente los módulos no tienen ninguna configuración específica, en cualquier caso, es posible modificar las opciones de la aplicación en cualquier módulo que así lo requiera. Algunos de los usos típicos son los de cambiar la descripción HTML en todas las acciones de un módulo, también se pueden añadir nuevos parámetros exclusivamente para un módulo concreto.

El archivo que más se ha manipulado en nuestro caso se llama el `generator.yml`, que podemos observar visualmente en la **Figura 22**.

El archivo **`generator.yml`** se utiliza para los módulos generados automáticamente a partir de una tabla de la base de datos, es decir, para los módulos utilizados en el *scaffolding* y para las partes de administración creadas de forma automática, contiene las opciones que definen como se muestran las filas y los registros en las páginas generadas y también define las interacciones con el usuario: filtros, ordenación, botones, etc.

Además podemos crear manualmente otros archivos para configurar nuestro módulo entre ellas pueden ser: para configurar la seguridad **`security.yml`**. para configurar la vista del módulo **`view.yml`**, etc.

Subdirectorio `lib`:

El directorio `/lib` debería contener sólo las bibliotecas (libraries) necesarias para ejecutar llamadas a funciones creadas por defecto del framework o creadas por el programador.

En el presente caso solo utilizamos las creadas por el framework, estas librerías se encuentran en dos archivos **`nombreGeneratorConfiguration.class.php`**, en donde nombre se refiere al nombre del módulo, y el **`nombreGeneratorHelp.class.php`**, de la misma manera, nombre se refiere al nombre del módulo.

Estos dos archivos contienen funciones públicas que son llamadas por el sistema, para que realice alguna actividad, pero estas solo se encuentran en el directorio cache en forma temporal,



mientras este módulo se lo utilice, ya que si lo mandamos a limpiar el cache, todas estas desaparecen, aquí vamos a ver el contenido de la caché.

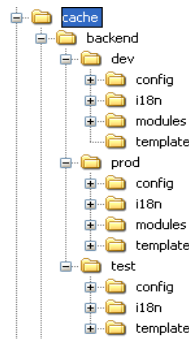


Figura 23. Estructura de archivos de la carpeta cache.

Como hemos podido observar en la **figura 23**, tres subdirectorios dev, prod y test, estos son los llamados entornos: El **entorno de desarrollo (dev)**: Este es el ambiente utilizado por desarrolladores web para añadir nuevas funciones, corregir los errores, etc., el **entorno de producción (prod)**: Este es el entorno donde un usuario final interactúa y el **entorno de prueba (test)**: Este entorno se utiliza para probar automáticamente la aplicación. Depende cómo se esté trabajando para que se muestren los archivos temporales en la caché, es decir que si estamos trabajando en el entorno de desarrollo los directorios *prod* y *test* van a estar vacíos, y así en los otros entornos solo mostrarán el contenido de estos directorios si están trabajando en ellos.

NOTA: Como se muestra en la **figura 23**, el contenido de la caché en cada uno de los entornos, también contienen los mismos directorios que mostramos en el directorio apps de la aplicación y de los respectivos módulos, es decir que en este directorio se crea como una especie de espejo en la caché, solo que en esta se muestra el código de las funciones creadas automáticamente por el framework, en las cuales podemos introducir nuestro propio código para personalizarlos o nuestras propias funciones.

Subdirectorio templates:

Este directorio contiene todas las plantillas globales para una aplicación cuando es el caso de la aplicación y para el módulo si este es el caso, es decir que este directorio se encarga de la vista en Symfony, la vista está formada por dos partes:

- La presentación HTML del resultado de la acción (que se guarda en la plantilla, en el layout y en los fragmentos de plantilla).
- El resto, que incluye entre otros los siguientes elementos:
 - Declaraciones <meta>: palabras clave (keywords), descripción (description), duración de la cache, etc.
 - El título de la página: no solo es útil para los usuarios que tienen abiertas varias ventanas del navegador, sino que también es muy importante para que los buscadores indexen bien la página.
 - Inclusión de archivos: de JavaScript y de hojas de estilos.



- Layout: algunas acciones necesitan un layout personalizado (ventanas emergentes, anuncios, etc.) o puede que no necesiten cargar ningún layout (por ejemplo en las acciones relacionadas con Ajax).

En la vista, todo lo que no es HTML se considera configuración de la propia vista y Symfony permite 2 formas de manipular esa configuración. La forma habitual es mediante el archivo de configuración view.yml. Se utiliza cuando los valores de configuración no dependen del contexto o de alguna consulta a la base de datos. Cuando se trabaja con valores dinámicos que cambian con cada acción, se recurre al segundo método para establecer la configuración de la vista: añadir los atributos directamente en el objeto sfResponse durante la acción.

El layout de una aplicación se llama layout.php y se puede encontrar en el directorio apps/frontend/templates/, en el caso de la aplicación, como se muestra en la siguiente figura:

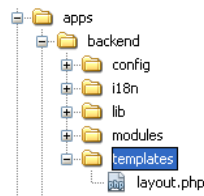


Figura 24. Localización de archivo layout de la aplicación.

Y archivos .php que pueden ser fragmentos de código que se les conoce como parciales y su nombre empiezan con dos rayas bajas seguidas y los archivos de vistas que siempre terminan con Success, como por ejemplo indexSuccess.php todos los mismos sirven como vistas en cada módulo, estos se muestran en la siguiente figura:

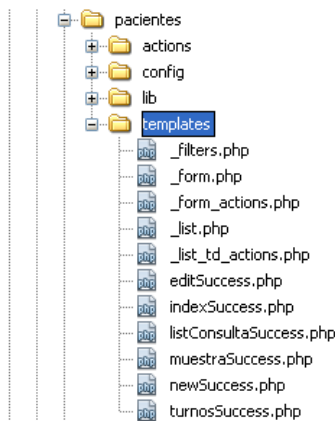


Figura 25: Estructura de archivos de la carpeta templates.

En resumen, la **figura 26**, muestra un esquema de cómo se organizan los archivos y subdirectorios de nuestro proyecto, siguiendo la estructura de proyecto / aplicación / módulo / acción.

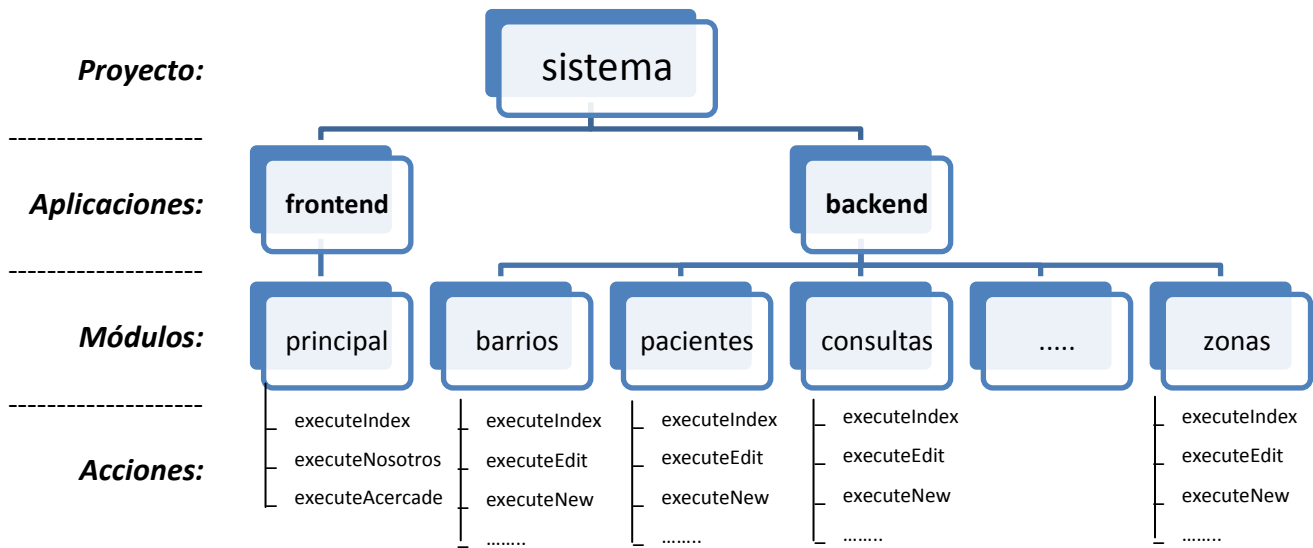


Figura 26: Estructura de un proyecto symfony.

RESULTADOS Y DISCUSIÓN

El presente tema de trabajo se lo realizó con un framework relativamente nuevo, por lo que en primer lugar se tuvo que realizar el estudio a fondo del mismo antes de empezar a diseñarlo en su totalidad.

Tanto para el diseño de la base de datos, servidor web y el desarrollo del sistema se utilizó herramientas libre en su totalidad, por lo que no tuvimos que pagar ninguna licencia, esta es una ventaja significativa que ayudó en gran medida al desarrollo del mismo.

La facilidad de acceso a la información necesaria para la continuación del desarrollo del proyecto, permitió seguir avanzando sin ninguna restricción en cuanto a este aspecto.

El sistema está orientado a las enfermeras y médicos que prestan los servicios profesionales en el Subcentro de Salud de San Antonio de Ibarra, los mismos que tendrán acceso a la Base de Datos mediante una intranet en el edificio de esta casa de salud.

En cuanto a la base de datos, en esta podremos registrar información necesaria de:

- Historias Clínicas de Pacientes.
- Datos Personales de Médicos que prestan sus servicios en esta casa de salud.
- Datos necesarios de medicamentos categorizados y el registro de las entradas y salidas de los mismos.
- Datos básicos de turnos que se dan a diario a los pacientes.



Todo esto, más la posibilidad de restaurar o crear copias de seguridad de los datos en un respectivo tiempo.

CON RESPECTO A SEGURIDADES:

El sistema está diseñado en su mayor parte para las personas que trabajan en esta casa de salud, ya que la información que se maneja solo las conciernen a ellos, es decir en su mayoría son las historias clínicas de pacientes, medicamentos y datos personales de los médicos que prestan su servicio aquí.

Un paciente, o una persona particular a esta casa de salud, solo podrá ingresar a ver la presentación, aspectos del sistema, y datos básicos del Subcentro de Salud san Antonio, pues para ingresar a la administración de información de la base de datos deberá tener una clave de acceso, el mismo que solo deben tenerlos personas autorizadas a usar el sistema, con este sistema mantenemos seguridad e integridad los datos muy importantes para el trabajo diario del Subcentro.

CON RESPECTO AL SOFTWARE:

En cuanto al sistema va a tener sus limitaciones con respecto al alcance que brinda a los usuarios, pues en su mayoría se pone más énfasis en las historias clínicas de los pacientes más no en turnos, medicinas, tipos de atención que se pueden dar y datos específicos para ellos, aunque el sistema si acoge estos temas pero de una manera superficial.

En cuanto a Reportes, este es otro aspecto que está limitado, pues tenemos pocos tipos de reportes, los más necesarios actualmente para el Subcentro, pero podríamos diversificarlo más aún.

Pero el tipo de programación que brinda Symfony, permite crear software escalable, por lo que podemos aumentar módulos para cubrir con la mayoría de requerimientos, es una de las grandes ventajas que se puede encontrar en esta herramienta.

CON RESPECTO A LAS TABLAS DE LA BASE DE DATOS

Las tablas de las bases de datos que se diseñó para el sistema están orientadas exclusivamente para el Subcentro y cubrir las necesidades más urgentes de esta casa de salud.

Para esto se define el modelo de datos, por lo que es necesario un ORM (Doctrine o Propel) para interactuar con la base de datos y crear el primer módulo de la aplicación.

La ventaja de utilizar Symfony es que esta herramienta se encarga de la mayor parte del trabajo, por lo que es fácil crear un módulo web completamente funcional sin tener que escribir mucho código PHP.

Todos los datos de Historias Clínicas, Médicos, Medicamentos, Turnos, Consultas, se lo guardan en una base de datos relacional.



Ahora, como Symfony es un framework orientado a objetos, esta herramienta nos da la posibilidad de trabajar con objetos. Por ejemplo, preferimos utilizar objetos a tener que escribir sentencias SQL para obtener los registros de la base de datos, ya que de esta manera estaremos más apegados a la programación orientada a objetos.

Por lo que para poder trabajar con objetos en una base de datos relacional, es necesario realizar un “mapeo”² o conversión entre la información de la base de datos y los objetos PHP, para esto lo podemos realizarlas con unas herramientas llamadas ORM, pues Symfony incluye por defecto dos de las más utilizadas: Propel y Doctrine, para la versión de Symfony (Versión 1.4) que se utilizó, viene por defecto Doctrine.

El ORM procede a partir de la descripción de cada tabla y de las relaciones entre tablas a crear clases PHP las mismas que son necesarias para trabajar con objetos.

PRESENTACIÓN DEL SISTEMA:

Quedan vacíos en algunas opciones que se podría corregir, como es lo que se refiere a la presentación de las páginas, en los que se puede perfeccionarlos incrementando animaciones flash para que se vea aún más vistoso, pero se utilizó conocimientos básicos de hojas de estilo y javascript para lograr una presentación lo más apropiado para llegar a nuestro objetivo, por lo que tenemos el resultado en la **figura 27**:



Figura 27: Pantallas de Presentación del sistema del Subcentro de Salud san Antonio.

INCONVENIENTES:

Un inconveniente que se tuvo en el desarrollo del proyecto fue a la hora de su instalación y configuración en el equipo donde tendría que trabajar como servidor, y el temor que tenían las personas, usuarios de este equipo de infectar su máquina, ya que en este equipo se encontraba

² Técnica que crea una base de datos orientada a objetos virtual, sobre la base de datos relacional.



instalado una aplicación (congelador) para retornar al estado anterior y para lograr la instalación de este sistema en el equipo se debía descongelarlo.

CONCLUSIONES

- ✓ El Subcentro de Salud San Antonio es el eje principal de salud que tiene la parroquia de San Antonio de Ibarra, es por eso que el número de historias han crecido, pues acá acuden pobladores de sus barrios aledaños.
- ✓ El crecimiento de la población hace también que los servicios de salud requieran una urgente actualización en cuanto a herramientas para el tratamiento de información para no generar aglomeraciones.
- ✓ Existen herramientas libres en la actualidad que brindan grandes ventajas para el diseñador de sitios web como es el caso de Symfony.
- ✓ Un framework automatiza las tareas más comunes por lo que el desarrollo de aplicaciones se simplifica en gran medida.
- ✓ Un framework cambia el criterio de un programador pues proporciona una estructura al código fuente, logrando así crear código más legible y más fácil de mantener.

LITERATURA CITADA

Para encontrar mayor información acerca del framework utilizado, y específicamente de admin generator de Doctrine, lo pueden encontrar en el capítulo 12 del libro "Practical Symfony" en español (Día 12), en la dirección web http://www.symfony-project.org/jobeeet/1_2/Doctrine/es/.

Otras fuentes citadas:

- POTENCIER, Fabien & ZANINOTTO, François, SYMFONY 1.0, LA GUÍA DEFINITIVA, Creative Commons, Año 2008.
- POTENCIER, Fabien & ZANINOTTO, François, SYMFONY 1.2, LA GUÍA DEFINITIVA, Creative Commons, Año 2008.
- PÉREZ, Javier Eguíluz; INTRODUCCIÓN A AJAX, Creative Commons, Año 2008.



AUTOMATION SYSTEM INFORMATION MANAGEMENT AND DRUG HISTORY IN THE HEALTH SUBCENTRES SAN ANTONIO DE IBARRA

Developed by: Egdo. Jorge Aníbal IpiALES IpiALES

Telf: 062 933310; **Cel** 097292992

Email: jipiales2006@hotmail.com; j_ipiales2005@yahoo.com

UTN-2011

SUMMARY

For the realization of this project was to model first the entity-relationship diagram, a free tool helped MysqlWorkBench, which facilitated this design, because it works visually, once this model we had to get tools programming, so it was decided to Netbeans (version 6.9.1) and symfony 1.4, a framework which helps greatly with PHP programming, helped with the same Doctrine, is now the default ORM for symfony from version 1.4, accelerate web design projects.

The knowledge of these tools, more than basic knowledge of JavaScript, CSS, ajax and html helped the completion of this project to facilitate the work of the Department of Statistics in the Health Sub-center San Antonio de Ibarra.

INTRODUCTION

As the automation requirements of information in public or private entities increase, programming technology should go hand in hand with these requirements, because at present web programming has been increasing along with the popularity of the Internet, so the need to increase the knowledge of system design tools for the web.

One of the drawbacks with the designers, refers to the acquisition tools to achieve this goal, so a great alternative you have is to use free tools, which we can get off from the web, as there is a variety of these, the which we can use it without any monetary payment and performance does not differ greatly with paid tools.

The creation of tools that is now quite widespread as are the frameworks has been to expedite the process of software development for the web, this is the case Symfony, as there is good documentation on the network and which discussions blogs in turn help to a good understanding of this tool.

In this paper we will indicate the processes necessary to develop web applications using netbeans, symfony and helped in this case to apply for project development for the web in the health sub-center "San Antonio" with a local web server WampServer 2i.

MATERIALS AND METHODS

In this paper, first we had to design entity relationship diagram helped MysqlWorkBench (version1.5.18), the same which is free and has a visual environment and migable as shown in the figure below.

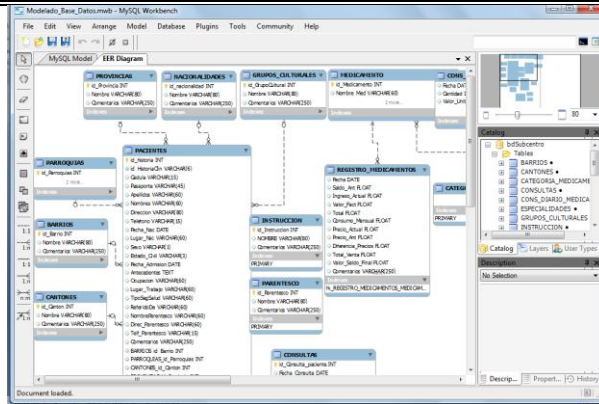


Figure 1. Design with MySQL Workbench entity relationship diagram.

After designing the optimal scheme of the database for this application, the entity relationship diagram best, was as follows:

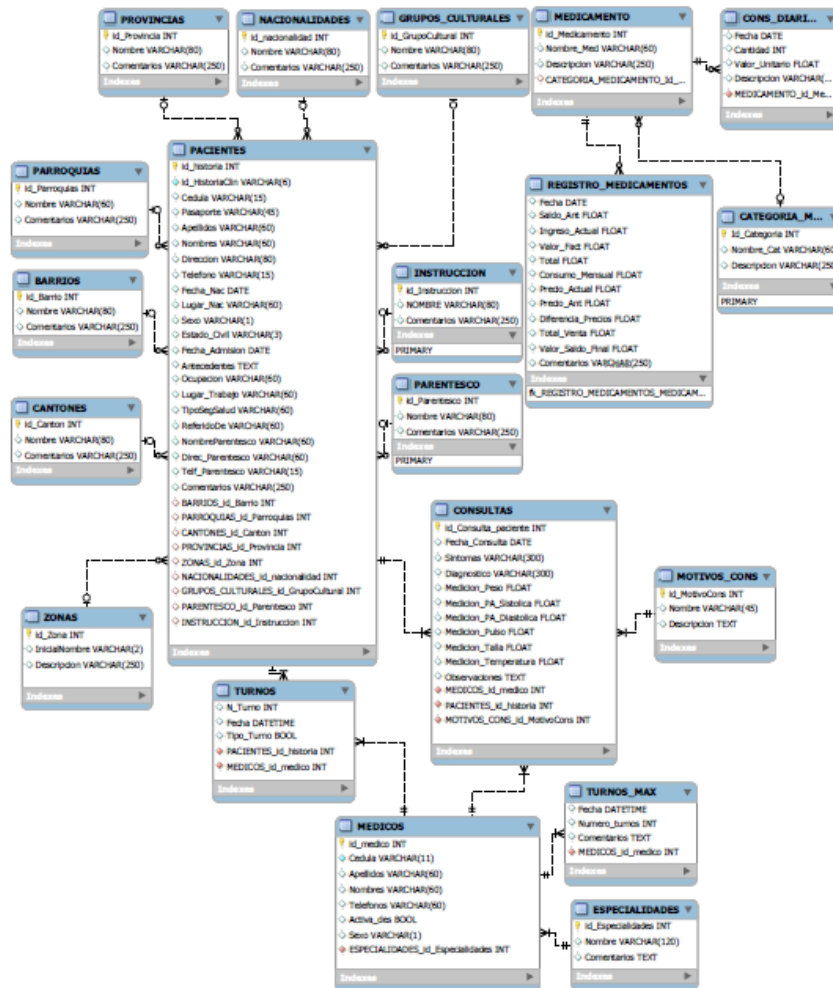


Figure 2: Entity-Relationship Diagram of the database for the health subcenter system of San Antonio.



The tables were designed for the system of health subcenter San Antonio that are shown in Figure 2, were designed in the tool you quoted above.

This diagram was the basis to bring this scheme to the physical database engine to MySQL database, it is installed together with the application Wamp2i.

To run the application, we first run our web server, which in our case is the WampServer 2i, so we install it installed, otherwise it will not work, so to install it first must be downloaded from the Internet (<http://www.wampserver.com/en/>), if you have installed should see an icon like this:



Figure 3: WampServer icon representing 2i

SYMFONY SETTINGS IN NETBEANS.

As mentioned in the introduction, our development tools will be Netbeans Symfony helped to facilitate programming with this framework and make it visually.

As mentioned in the introduction, our development tools will be Netbeans Symfony helped to facilitate programming with this framework and make it visually.

Symfony is based on MVC (Model View Controller), ie, the logical structure of the projects created are divided into:

- **Applications:** that is our project,
- **Classes:** Classes types of functionality that includes our application (user management, news management, management of comments, etc..) And
- **Modules:** which is which details how to make each feature of the class to which they are assigned (high, delete and edit users, etc..).

Now we turn to configure Netbeans to work with symfony, as follows:

First we have a "LAMP" (Linux-Apache-MySQL-PHP / Python / PERL) installed on your computer, in our case we use WAMP SERVER 2.0.

Then download the latest stable version of symfony from its website, for the start of our project was the latest stable Symfony 1.4.

After downloading the package usually comes packaged, unzip in the directory on the web, as we work locally, create a folder on drive C: /, as follows: "C: \ SubcentroSalud \ system \", then the full address where the decompressed will be something like: C: \ SubcentroSalud \ system \ lib \ vendor \ symfony



In our next step we opened Netbeans, and then create a new PHP project, the wizard will guide us step by step, where we only give our project name and location of our project files, but in the step where we choose the framework (PHP frameworks) obviously choose Symphony. When we select various configuration options appear. We'll have to click on Options on the right of the window, as shown in Figure 4:

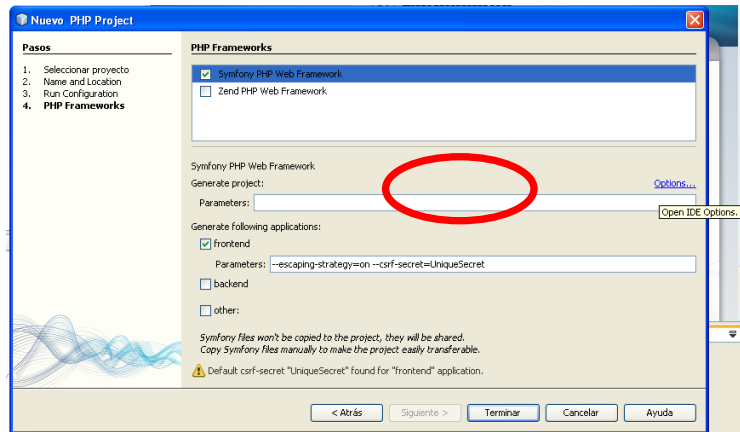


Figure 4. Location of Options to configure symfony in Netbeans

After clicking on Options ..., we will open the PHP configuration panel in this window click on the tab where PHP will indicate the path where the PHP executable we have installed LAMP, in our case in the following path:

C: \ wamp \ bin \ php \ php5.3.0 \ php.exe

In addition, this window will indicate the Symfony installation path by clicking the Add Folder, which will have decompressed folder in the directory of our project

C: \ SubcentroSalud \ system \ lib \ vendor \ symfony

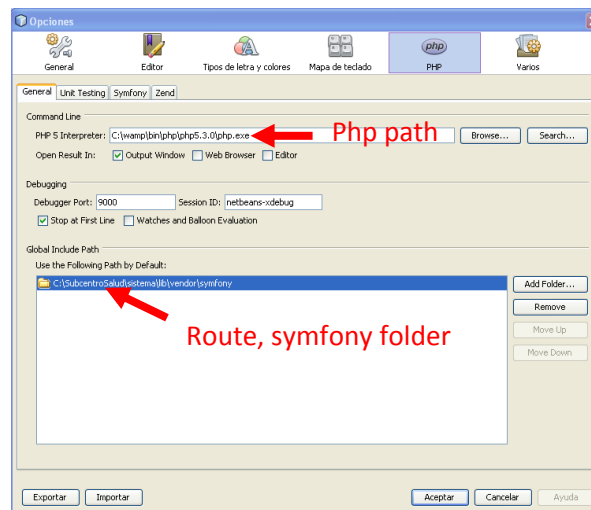


Figure 5. Indicate Php path and folder in the project Symfony



Now, Symfony tab of this screen will indicate the path to the binary framework, which in our case is found at the following address:

C: \ SubcentroSalud \ system \ lib \ vendor \ symfony \ data \ bin \ symfony

This is indicated in the figure below:

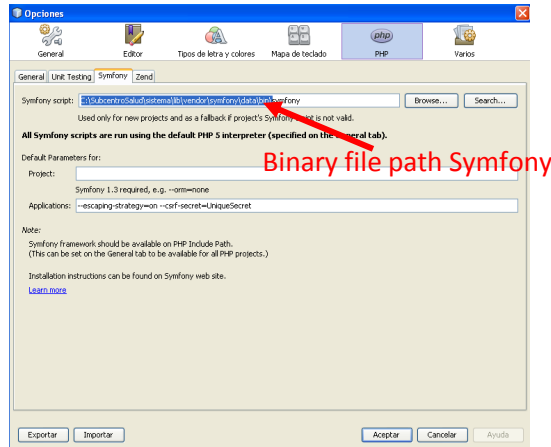


Figure 6. Location Symfony binary file

Finally, we click OK, but before finishing, select the checkbox Backend, where it should be seen as we indicated in the graph below, this in order that our project manage the database after that if you click on Finish.

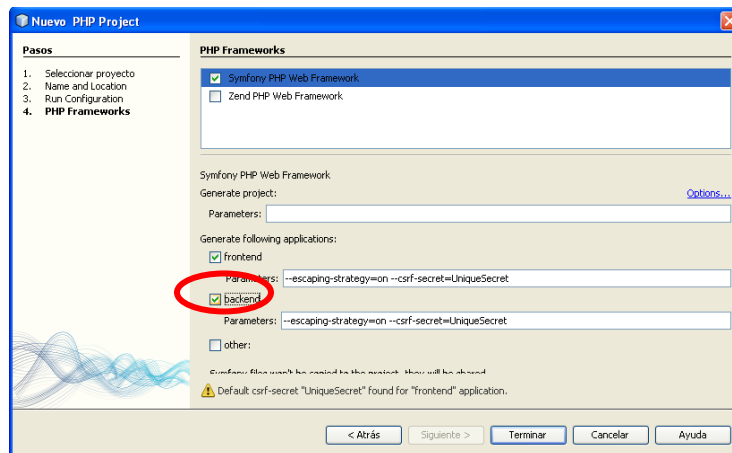


Figure 7. Location backend to run the DB.

If there is any problem during all the steps that the wizard guided us, we will have configured and installed our symfony project in NetBeans.

CONFIGURING APACHE WEB SERVER

As we solve this, we must first set the host in this case our local host will be virtual, so that it runs under a different name in another host and port (in this case port 80).



On the Internet we found several examples of how to create a virtual host to run the project locally, here's an example:

```
NameVirtualHost *:80
<VirtualHost *:80>
```

```
ServerName miApp
DocumentRoot "C:\Users\Juanjo\Documents\NetBeansProjects\miApp\web"
DirectoryIndex index.php
<Directory "C:\Users\Juanjo\Documents\NetBeansProjects\miApp\web">
AllowOverride All
Allow from All
</Directory>
```

#solo para la barra de depuración:

```
Alias /sf "C:\symfony-1.4.11\data\web\sf"
<Directory "C:\symfony-1.4.11\data\web\sf">
AllowOverride All
Allow from All
</Directory>
</VirtualHost>
```

What we do is we take any of these codes and edit it to attach to your project, take into consideration that we will use Symfony as a framework for programming with PHP, which is why this configuration refers to this folder where we will symfony, so we would be as follows:

```
# Esta es la configuración de Sistema para Subcentro de Salud
#Listen 127.0.0.1:80

<VirtualHost 127.0.0.1:80>
ServerName sistema
DocumentRoot "C:\SubcentroSalud\sistema\web"
DirectoryIndex index.php
Alias /sf "C:\SubcentroSalud\sistema\lib\vendor\symfony\data\web\sf"
<Directory "C:\SubcentroSalud\sistema\lib\vendor\symfony\data\web\sf">
AllowOverride All
Allow from All
</Directory>
<Directory "C:\SubcentroSalud\sistema\web">
AllowOverride All
Allow from All
</Directory>
</VirtualHost>
```

As symfony already established routes in the project folder, we have to follow these steps:



1. Copy the contents (Ctrl + C) the code above (the VirtualHost)
2. Open the httpd.conf file located in the following path: C: \ wamp \ bin \ apache \ Apache2.2.11 \ conf,
3. Paste it (Ctrl + V), usually at the end of the contents of this file,
4. Save,
5. And finally restart Apache services.

If all went well, and if you have configured Apache correctly, open the browser and note the following address: http://localhost:80, but we can also do it from Netbeans, typing the F6 key to execute the project. And so we should get a screen similar to the following:

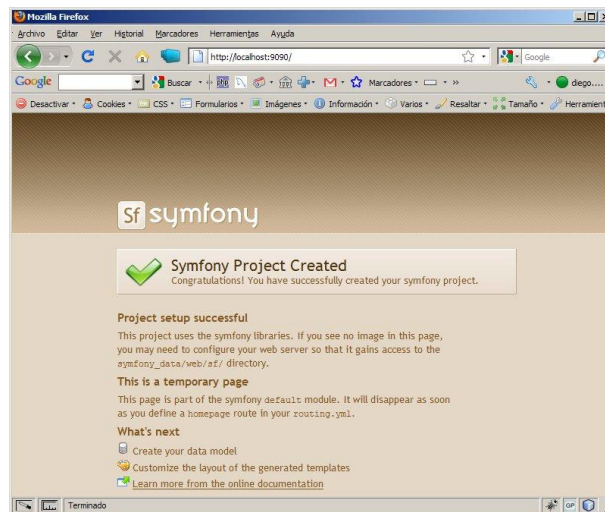


Figure 8. Symfony screen at the end of basic configurations

All this is to run the application as a virtual server, so you must have added the virtual domain hosts file of your operating system to go to this address C: \ Windows \ System32 \ drivers \ etc \ hosts and added:

```
127.0.0.1    localhost
127.0.0.1    sistema
```

So to enter our project http://sistema single digits in your browser (Mozilla Firefox 4 or higher preferred).

SYSTEM

After making the necessary settings for your application to work, we analyze the structure of our project, it was done with the help of Netbeans 6.9.1., The latest stable version to date of initiation of our project.

To begin to develop projects with symfony, you must have a knowledge of at least basic HTML, CSS, programming this way learn to be a little more orderly in relation to what you have to see the place where they will go code programming.



Well, sometimes you use only HTML, in other cases we should use PHP alone or a mixture of them also refer to files CSS for styling our pages.

It seems important to understand how the code structure Symfony. Any project created by Symfony is a series of folders and files containing the framework code base. The tree of folders in a project is:

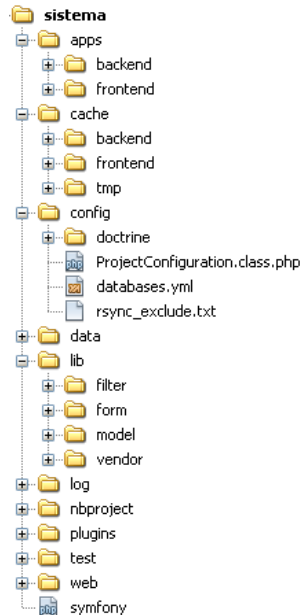


Figure 9. File structure of symfony project

In addition, a project created with symfony are divided into:

- Applications: that can have the same modules can usually be two backend and frontend applications.
- Modules: These in turn tend to coincide with the tables in the database created as the project purpose.

Then we have the database and we are clear as is the structure of a symfony project, we can insert our code where deemed necessary.

For this we need to create a folder where we will create all the folders and copy the folder Symfony downloaded from their official website, in our case we will help with a tool "Netbeans"

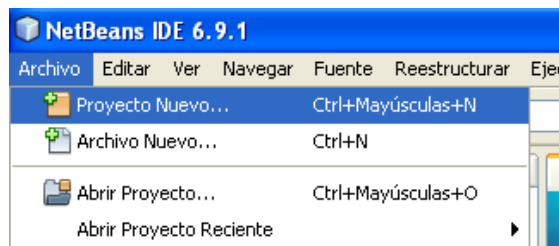


Figure 10. Create a new project in symfony with Netbeans.



This way of creating the project replaces the following code:

```
symfony generate: project system
```

Whichever way we create our project, we have created the entire tree of directories and files needed to launch our website with this framework.

Then we create our modules, but if we use Netbeans as it is in our case to create our project and also creates two default applications.

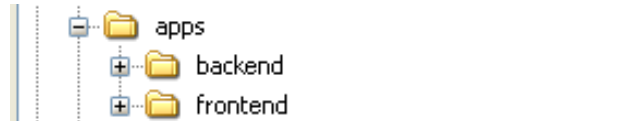


Figure 11. Folder structure of the application.

However if you do not use Netbeans, you must create them manually, using a code like the following:

- `symfony generate: app - escaping-strategy = on - csrf-secret = UniqueSecret frontend`
- `symfony generate: app - escaping-strategy = on - csrf-secret = UniqueSecret backend`

When calling the `generate: app` task, we have also been two security related options:

- `escaping-strategy`: Enables output escaping to prevent XSS attacks
- `csrf-secret`: Enables session tokens in forms to prevent CSRF attacks

The next step is to connect our application to the database, for this, Symfony has a `databases.yml` configuration file in which we will indicate where it is located, to which database and which user and password is to be made.

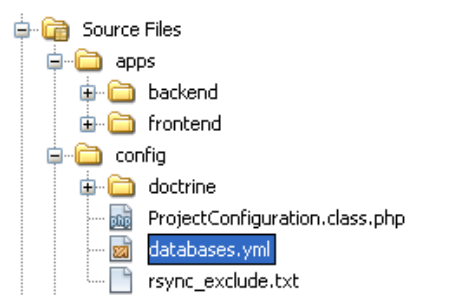


Figure 12. Location of the configuration file of the database for the project.

This file usually contains the following code similar to the following:

```
all:
  doctrine:
    class: sfDoctrineDatabase
    param:
      dsn: 'mysql:host = localhost, dbname = bdblog'
```



```
username: root  
password: null
```

With this, we have configured our project database. Now we tell Doctrine to generate schema.yml from the database:

```
symfony doctrine: build-schema
```

Then we create the model classes to be generated to manage the database through objects, as follows:

```
symfony doctrine: build-model
```

In a similar way to create doctrine, filters, forms, etc. To not perform all these actions separately, using a single command, as follows:

```
Symfony doctrine: build-all
```

Only with the above command must be careful if you already have the database created above, as this is a way to reset the database and leave it blank, or rebuilds.

As we noted above, we use Netbeans, so that these lines of code can be run visually.

If all went well, we can go with your browser to see the screen Symfony where we said that the project has been created for that visit <http://sistema/>, as follows:



Figure 13. Initial screen of a symfony project created.

From now on no more than create modules in our applications such as Backend and Frontend, in addition to insert code to manipulate and control our vision for the system.

NOTE: The way to run a symfony command is simple, just right click on the name of the application in which we are working and a window will appear as shown in Figure 14:

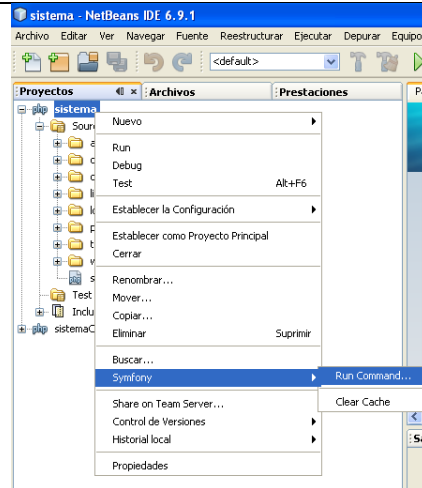


Figure 14: Execute commands in Netbeans Symfony.

As we left click on Run Command ... option from the dropdown menu as shown in Figure 14. File structure of symfony project

In this project we have two applications, Backend and Frontend:

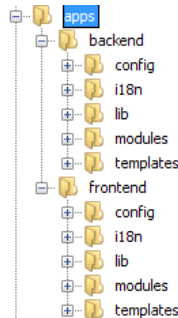


Figure 15: symfony file structure (application).

FRONTEND: In Figure 15 we can see visually that this application consists of other directories that when the application is created with Netbeans Symfony creates them.

BACKEND: This application has a structure similar to the above as we can see in Figure 15, only later we see that in this application is where more effort was invested in programming and time commitment.

STRUCTURE OF THE APPLICATION

The application contains several subdirectories as we can see in Figure 15, these are: Config subdirectory:

The yaml folder containing files to configure the application, these can be seen in the figure below:

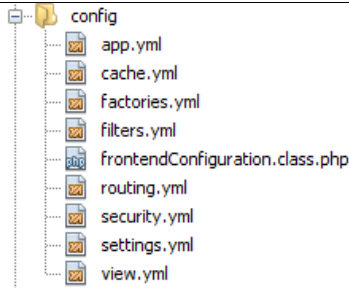


Figure 16. Contenido the config folder.

As we can see, there yml files to configure the application (app.yml) to configure the cache (cache.yml), the basic objects needed by the framework during the life of any application (factories.yml) to configure the filters (filters.yml) to configure the ways to get to different pages called routes (routing.yml) to ensure the application or not and that only authorized persons can enter (security.yml) to configure the different system execution environments (settings.yml) and of course to set the view (view.yml). All these files come with their own code that was generated in the creation of the application.

I18N SUBDIRECTORY:

Save files in this folder language settings for applications, in the case of the frontend application, has no configuration file for this, but in the case of the application backend if we have, this file is located in the direction shown in Figure 17.



Figure 17. I18n folder contents.

As we can see, are xml files, where if there are open Spanish translations for the most common messages that normally appear in English.

Lib subdirectory:

This folder contains useful libraries for the application, which can be created by the framework itself or by the programmer to create or use your own libraries, usually comes with a single file that is created by default when creating your application, as shown in the following figure:



Figure 18. Lib folder contents.

Modules subdirectory:



This is the directory containing the modules that will be created for the system, the same that can be created visually netbeans helped, as shown in the following figure:

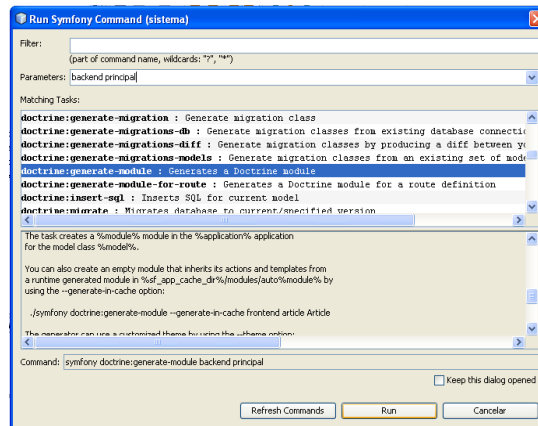


Figure 19. Screen to create Netbeans modules

In the frontend application, we only have a single module where basic information entered Subcentro (We) and application (About ..), it is called principal.

STRUCTURE MODULES

In each module, symfony creates a default subdirectories by executing the command to create modules that visually helped netbeans it can be found in the following figure:

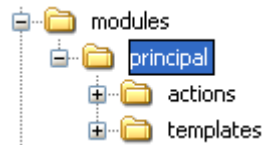


Figure 20. Module structure.

Subdirectory actions:

Where they will have the functions necessary to perform some action, to implement these actions return the result to the respective layout, so that for every action we have a display file (layout) in the templates subdirectory subdirectory of the module as seen in figure 15, the syntax of the name of the action is as follows:

public or protected execute + [nombre_acción] + ([arguments])

For example public executeIndex (sfWebRequest \$ request), all these functions are in the same actions.class.php file which can be public (public) or private (protected) within the module directory we are watching.

In the case of health sub-system, generate the module with the command symfony doctrine:admin-generator, so it automatically creates subdirectories over that seen in Figure 19, that we will show in the figure 20.

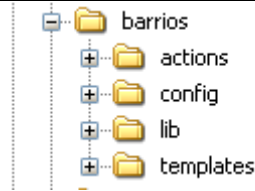


Figure 21: Structure of Modules built admin_generator.

As shown in Figure 21, more than two folders that was visualized in Figure 20, we have a folder where files contain configuration (config) and a folder in which to save the module libs (lib). But each folder contains files that may be with. Yml normal configurations, or. Class.php where we can bring our own functions or functions that manipulate the default symfony creates and files. Php, for partial or views of the module, some of these we can see in the following figure:

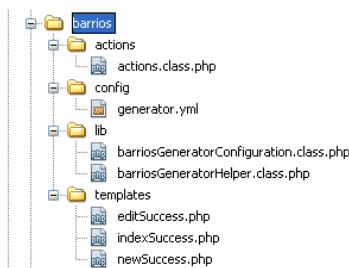


Figure 22. Structure of directories and subdirectories of modules created with admin-generator.

Config subdirectory:

The config subdirectory of a module, modules initially have no specific configuration, in any case, it is possible to modify the application settings in any module that requires it. Some typical applications are changing the HTML description for all actions of a module, you can also add new parameters only for a particular module.

The file has been manipulated more in our case is called generator.yml, we can see visually in **Figure 22**.

Generator.yml file is used for modules automatically generated from a table in the database, ie to the modules used in the scaffolding and administration parts automatically created, containing features that define it as are displayed in rows and records generated pages and also defines the user interaction: filters, sorting, buttons, etc.

We may also create other files manually to configure our module among them may be: to set up security security.yml. to configure the module view.yml view, etc..

Lib subdirectory:

The /lib directory should contain only those libraries (libraries) required to implement function calls the framework created by default or created by the programmer.



In this case we use only those created by the framework, these libraries are located in two files `nombreGeneratorConfiguration.class.php`, where `nombre` refers to the module name and the `nombreGeneratorHelp.class.php`, in the same way, `nombre` refers to the name of the module. These two files contain public functions that are called by the system to perform some activity, but these are only found in the cache directory temporarily, while this module is in use, because if we sent him to clean the cache, all they disappear, here we will see the contents of the cache.

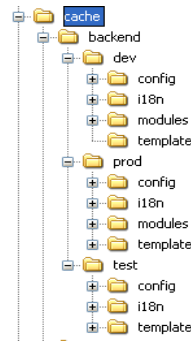


Figure 23. Structure files in the cache.

As we have seen in Figure 23, three subdirectories `dev`, `prod` and `test`, these are called environments: The development environment (`dev`): This is the environment used by web developers to add features, fix bugs, etc.. the production environment (`prod`): This is the environment where an end user interacts and test environment (`test`): This environment is used to automatically test the application. It depends how it is working to display the temporary files in the cache, ie if we are working on the development environment and test `prod` directories will be empty, and so in other environments only display the contents of these directories if they are working on them.

NOTE: As shown in Figure 23, the contents of the cache in each of the environments, they also contain the same directories as shown in the apps from the application and the respective modules, ie in this directory is created as a kind of mirror in the cache, only this is the code of the functions created automatically by the framework, in which we introduce our own code to customize or our own functions.

Templates subdirectory:

This directory contains all the global templates for an application when the application for the module and if this is the case, ie that this directory is responsible for the symfony view, the view consists of two parts:

- The HTML presentation of the results of the action (which is saved in the template, the layout and template fragments)
- The remainder, which includes among others the following elements:
 - Meta declarations: keywords, description, duration of cache, etc..
 - The title of the page: not only useful for users who have multiple browser windows open, but is also very important for search engines to index page properly.
 - Include files: JavaScript and style sheets.



- Layout: Some actions require a custom layout (pop-ups, ads, etc.) Or may not need to load any layout (eg actions related to Ajax).

At the hearing, all HTML is not considered setup symfony own view and allows 2 ways to handle that setting. The usual way is by view.yml configuration file. It is used when the settings are not dependent on the context of a query to the database. When working with dynamic values that change with each action, the second method is used to set the view configuration: add attributes directly in the object sfResponse during the action.

The layout of an application is called layout.php and can be found in the apps/frontend/templates/, in the case of the application, as shown in the figure below:

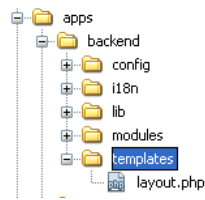


Figure 24. Localización layout file of the application.

And files .php that can be fragments of code that are known as partial and name start with two dashes followed by low and view files that always end with Success, like all the same indexSuccess.php serve as seen in each module, these are shown in the figure below:

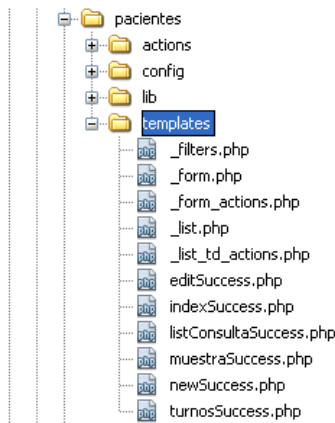


Figure 25: Structure of files in the templates.

In summary, Figure 26 shows a diagram of how to organize files and subdirectories of our project, following the structure of the project / application / module / action.

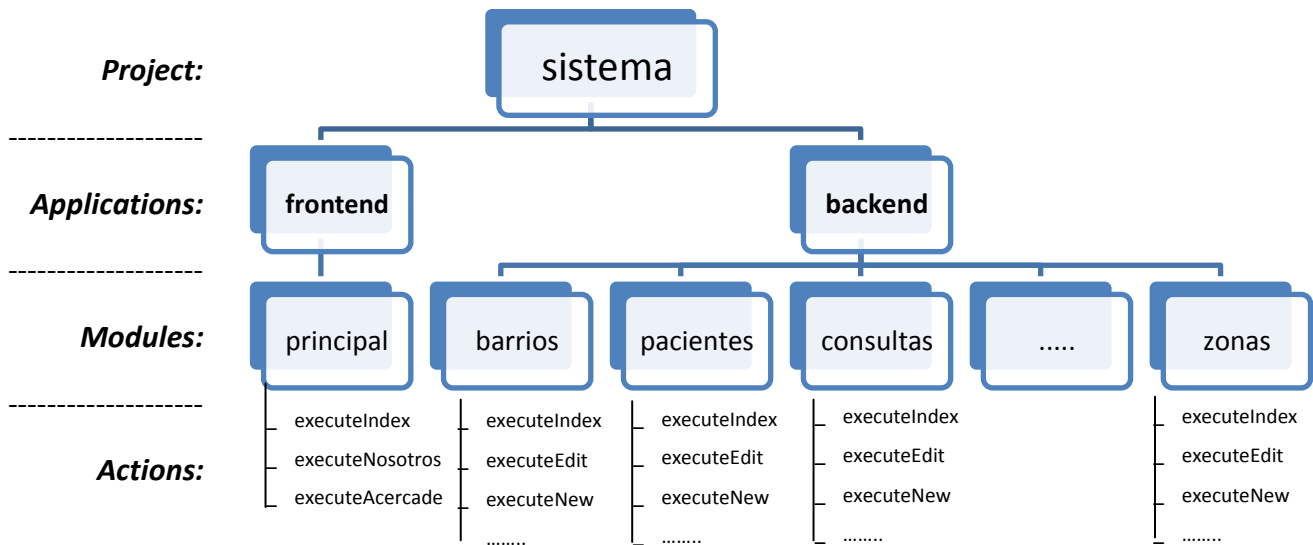


Figure 26: Structure of a symfony project.

RESULTS AND DISCUSSION

This item of work it performed with a relatively new framework, so we first had to make a thorough study of it before you start designing a whole.

Both for the design of the database, web server and development tools free system was used in its entirety, so we did not have to pay any license, this is a significant advantage that greatly helped the development.

The ease of access to information necessary for the further development of the project, allowed to move forward without any restriction on this.

The system is aimed at nurses and doctors who provide professional services in the Health Sub-center San Antonio de Ibarra, they will have access to the database through an intranet in the building of this house of health.

Regarding the database, we record this information to:

- Patient Medical Records.
- Personal Data Doctors serving in the nursing home.
- Data needed drugs categorized and recording inputs and outputs of the same.
- Basic data shifts that occur daily to the patients.

All this, plus the ability to restore or create backup copies of data in a respective time.



WITH RESPECT TO SECURITY:

The system is designed mostly for people working in the nursing home, because the information is handled only the concern to them, ie they are mostly the case histories of patients, medications and personal data physicians who serve here.

A patient, or a particular person at this nursing home, you can only enter to see the presentation aspects of the system, basic data and health sub-san Antonio, then to enter the information management database should be a password, it must take them only persons authorized to use the system, this system maintain data integrity and security very importantes for the daily work of the Sub-center.

WITH RESPECT TO THE SOFTWARE:

For the system will have its limitations on the scope it provides to users, since most of them put more emphasis on the medical records of patients not on shift, medicine, types of care that can be taken and data specific to them, while the host system if these issues but in a superficial way. As for reports, this is another aspect that is limited because we have few types of reports, the more necessary at present for the subcenters, but we could diversify even more. But the type of programming that provides symfony, software to create scalable, so that we can increase with modules to cover most requirements, it is one of the great advantages which can be found in this tool.

WITH RESPECT TO THE TABLES OF THE DATABASE

The tables in the database that was designed for the system is provided solely for the sub-centers and urgent needs of the nursing home.

For this, define the data model, so you need an ORM (Doctrine or Propel) to interact with the database and create the first module of the application.

The advantage of using Symfony is that this tool is responsible for most of the work, making it easy to create a fully functional web module without writing much code PHP.

All data Medical Records, Doctors, Drugs, Shifts, Consultations, put it in a relational database. Now, as symfony is an object-oriented framework, this tool gives us the opportunity to work with objects. For example, we prefer to use objects to having to write SQL statements to obtain records from the database, because this way we will be more attached to the object-oriented programming.

As for working with objects in a relational database, you must perform a "mapping" or conversion between the information in the database and PHP objects to this we can perform them with some tools called ORM, Symfony as default includes two of the most used: Propel and Doctrine, for symfony version (Version 1.4) was used, it comes standard Doctrine.

The ORM proceeds from the description of each table and the relationships between tables to create the same PHP classes that are needed to work with objects.



PRESENTATION OF THE SYSTEM:

Gaps are some options that could be corrected, as regards the presentation of pages, which can be improved by increasing flash animations to make it look even more beautiful, but basic knowledge is used style sheets and javascript to achieve a more appropriate presentation to reach our goal, we have the result in Figure 27:



Figure 27: Presentation Screens health sub-system of San Antonio.

DISADVANTAGES:

A disadvantage was taken into the development of the project was at the time of installation and configuration on the computer would have to work as a server, and the fear that people have, users of this computer to infect your machine, since this equipment was installed an application (freezer) to return to the previous state and to achieve the installation of this system on the computer should be thawing.

CONCLUSIONS




- The Health Sub-center San Antonio is the main health is the parish of San Antonio de Ibarra, is why the number of stories have grown, as people come here of their surrounding neighborhoods.
- The population growth also makes health services require urgent update in terms of tools for information processing to avoid creating crowds.
- Free tools exist today that offer great advantages for the designer of web sites such as Symfony.
- A framework automates common tasks so that application development is greatly simplified.
- A framework changes the program criteria as it provides a structure to the source code, thus creating code more readable and easier to maintain.



LITERATURE CITED

To find more information about the framework used, and specifically Doctrine admin generator, can be found in Chapter 12 of the book "Practical symfony" in Spanish (Day 12) in the web address http://www.symfony-project.org/jobeeet/1_2/Doctrine/es/.

Other sources cited:

-  POTENCIER, Fabien & Zaninotto, François, Symfony 1.0 DEFINITIVE GUIDE, Creative Commons, Year 2008.
-  POTENCIER, Fabien & Zaninotto, François, symfony 1.2, The Definitive Guide, Creative Commons, Year 2008.
-  Perez, Javier Eguiluz, Introduction to AJAX, Creative Commons, Year 2008.