En el documento capaVistaI.docx vimos como crear nuestro primer template.php para nuestra página, ahora debemos empezar a crear todas las vistas que va a contener la página. Las vistas contienen toda la información que se visualiza en la página, no pueden contener casi ninguna lógica y deben estar conenidas dentro del directorio view de la arquitectura COCOAS.

Retomando un poco lo que habíamos hecho en el pasado documento vamos a recordar la estructura de nuestro archivo template.php. Si recuerdan bien, el unico archivo que no creamos en ese momento fue el archivo content.php. Aqui tampoco lo vamos a hacer, lo que si vamos a hacer es crear multiples archivos “content” que serán las vistas de nuestra pagina web. Para eso debemos entender un poco mas quién es el templateManager.php y como se integra con el archivo binding.xml.

Cuando una pagina web dentro de COCOAS es solicitada se utiliza el archivo binding.xml para obtener toda la logica de renderizado neCesaria para mostrar la pagina, para entender esto mejor podemos partir de un ejemplo de archivo binding.xml:

<?xml version="1.0"?>

<vistas>

<vista nombre=’inicio’ url='inicio.php' clase=’Inicio’>

<consulta destino=’bienvenida’>getBienvenida</consulta>

</vista>

</vistas>

En el archivo mostrado se puede apreciar las definiciones ejemplo de vistas de un modulo desarrollado con COCOAS, en particular en este ejemplo, se esta definiendo la vista de cuyo URL es “inicio.php”, a esa vista se le ha dado el nombre de “inicio” dentro del enrutamiento logico del proyecto. Ademas se le ha asignado la clase Inicio.php (Delegate) para las operaciones CRUD dentro de la vista y se ha definido especificamente un metodo de consulta llamado “getBienvenida” cuyo resultado es esperado dentro de una variable llamada $bienvenida. Entonces, ¿Que quiere decir todo esto? Bueno ahora vamos a ir explicando con mas detalle todas y cada una de estas propiedades.

* Nombre:

Representa un String unico entre todas las vistas del proyecto que permitira realizar el enrutamiento de las paginas sin la necesidad de verse atados al nombre fisico de la vista, esto nos trae como principal ventaja que podemos cambiar el nombre del archivo de la vista sin afectar casi en ninguna manera los vinculos de la pagina.

* URL

Es el nombre fisico de la vista, en nuestro caso “inicio.php”.

* Clase

Es la clase encargada de realizar las peticiones, inserciones y modificaciones de datos contra el servidor, se utiliza esta capa del framework para independizar las vistas de la arquitectura utilizada para manejar los datos. Este tipo de planteamiento tambien brindar una mayor organizacion del proyecto al amuentar el nivel de granularidad de la informacion y realizar un enfoque transaccional.

* Consulta: Esta parte del archivo se refiere a todos los posibles métodos a invocar a la hora de realizar peticiones (consultas de datos). Es necesaria una marcada separación entre las operaciones de consulta y las operaciones de modificaciones de datos.

Si vamos a utilizar este archivo bingin.xml entonces debemos crear las clases que complementen el código planteado en el archivo binding.xml. Para lograr eso vamos a necesitar crear 2 archivos, inicio.php e Inicio.delegate.php:

¿Qué es inicio.php?

Es nuestra vista, si recordamos nuestro template.php seria la sección azul que llamamos “content” dentro de nuestra plantilla por supuesto que no siempre nuestra sección content va a ser el archivo inicio.php, ya que la sección “content” es por naturaleza la que mas varia según cada sección de la pagina.

La pagina inicio.php va a contener la información relevante al inicio de la pagina, un mensaje de bienvenida, resumen de todo el portal, etc. Siempre debemos recordar que esta pagina no puede contener ninguna lógica avanzada de la aplicación. Vamos a crear un ejemplo de pagina inicio.php que sea acorde a lo que colocamos en el archivo binding.xml.

<div id="content">

!Esta es tu página de inicio! Y tu variable es <?php echo $bienvenida; ?>

</div>

Listo! Esa sería nuestra página de inicio, recuerda que debe estar embebida dentro del contenedor “content” porque así lo definimos en el template.php.

Para evitar establecer nuestra página de inicio dentro del proyecto debemos ir al archivo globals.php y setear en la variable DEFAULT\_VIEW el nombre lógico de la vista que queremos utilizar por defecto cuando entren en nuestra página. En nuestro caso vamos colocarle el valor “inicio” ya que esa es la única vista con la que contamos (debemos colocar como valor un nombre que coincida con alguno de los nombres colocados a las vistas en el archivo binding.xml).

$GLOBALS["DEFAULT\_VIEW"] = "inicio";

Ahora solo faltaría realizar dos cosas

Asignar a todos los roles los permisos para ver o no ver esta vista, y, crear el delegate que vamos a utilizar en esta vista y ya estaríamos corriendo nuestro primero hola mundo, para asignar roles debemos ir al documento capaVistaIII.docx, para crear el delegate debemos ir al documento capaDelegate.docx.