# 4. Documentos de arquitectura

### 4.1 Estándares de Codificación

Este apartado describe reglas y recomendaciones para el desarrollo de aplicaciones usando el lenguaje de la plataforma Microsoft.Net: C#, y su objetivo es definir líneas guías para enmarcar la consistencia de estilo y de formato para ayudar a los desarrolladores a evitar errores comunes. También se han incluido lineamientos y recomendaciones para el desarrollo de aplicaciones escalables y en capas. Específicamente este documento cubre convenciones de nombres, estilos de codificación, uso del lenguaje, mejores prácticas para aplicaciones, etc. Todos estos temas son cubiertos detallando reglas y recomendaciones las cuales son aplicables para C#.

Las fuentes de información utilizadas para el desarrollo de este documento, son las siguientes:

|  |  |
| --- | --- |
| **Design Guidelines for Class Library Developers** | http://msdn.microsoft.com/en-us/library/czefa0ke.aspx |
| **Guidelines and Best Practices** | http://msdn.microsoft.com/en-us/library/ms184412.aspx |
| **Best Practices** | http://msdn.microsoft.com/en-us/library/ms184411.aspx |
| **El Guille** | http://www.elguille.info/colabora/NET2005/giovannyfernandez\_EstandarCodificacionNET.htm |

**Estructura del Sistema**

En todo módulo habrá 2 carpetas que contengan:

* Entities 🡪 Clases que contienen definiciones de datos. Ej.: *Musico*, *GrupoMusical*.
* Factories 🡪 Clases que contienen la lógica de negocios (Manejo de los datos). Ej.: *MusicoFactory*

*Métodos requeridos en cada Factory*

* *Devolver():* Para obtener un solo objeto de ese tipo pasando como parámetro la clave primaria. Ej.: *MusicoFactory.Devolver(int idMusico)*
* *DevolverTodos():* Para obtener colección de objetos de ese tipo, no tiene parámetros. Ej.: *MusicoFactory.DevolverTodos ()*
* *DevolverTodosPor():* Para obtener colección de objetos de ese tipo, pasando como parámetro el nombre del campo o campos claves que no son clave primaria. Ej.: *MusicoFactory.LoadAllByGrupo\_Codigo(int idGrupoInteres)*
* *Guardar():* Para crear un registro nuevo en la tabla de ese objeto.
* *Borrar():* Para eliminar un registro de forma física o lógica del objeto.
* *Actualizar():* Actualización del objeto.

**Convenciones de Nombres**

Esta sección se concentra completamente en el trabajo de nombrar proyectos, archivos de código fuente, e identificadores incluyendo campos, variables, propiedades, métodos, parámetros, clases, interfaces y espacios de nombres (Namespaces).

* Parámetros: Deben llevar el mismo nombre que los campos de la tabla, sin prefijos.
* Colecciones: tipo *Array* (*string*[] ,*Musico*[] ) y llevan el nombre colNombreDelObjeto. Ej.: *colMusico*, *colGrupoDeInteres*.
* Formularios: Nombre del formulario identificativo de su función y comenzando con frm. Ej.: *frmListadoMusicos*
* Controles de Formulario (*textbox*, *button*, etc.) Identificador del control seguido del nombre de campo. Ej.: *txtNombre*, *txtApellido*.
* Siempre usar nombres con estilo de escritura CamelCase o PascalCase.

CamelCase

Forma de escribir una palabra donde su primera letra esta en minúsculas, y la primera letra de las subsiguientes palabras en mayúsculas. Ejemplo: *cantidadCuentas*.

PascalCase

Forma de escribir una palabra donde la primera letra esta en mayúscula, y la primera letra de las subsiguientes palabras igualmente en mayúscula. Ejemplo: *CantidadCuentas*.

* Evitar nombres totalmente en MAYÚSCULAS o en minúsculas. Nombres de una sola palabra serán totalmente en minúsculas si se usa con el estilo de escritura camelcase. Solo las constantes podran ir escritas en mayuscula.
* Nunca usar nombres que comiencen con caracteres numéricos.
* Variables y propiedades deben describir la entidad que representan no el tipo o tamaño.
* Evitar el uso de abreviaturas a menos que el nombre completo sea excesivo.
* Nunca incluir el nombre de la clase a los nombres de las propiedades. Ejemplo: *Cliente.Nombre* NO *Cliente.NombreCliente*
* Tratar de utilizar los siguientes prefijos en las variables y propiedades booleanas, “Puede”, “Es” o “Tiene”.
* Los nombres de todas las estructuras (namespaces, procedimientos, clases, interfaces y propiedades) de código deben ser en español.
* Utilice la técnica verbo-sustantivo para nombrar procedimientos que ejecuten alguna operación en un determinado objeto, como por ejemplo *CalcularDesperdicio().*
* Un nombre debe más bien expresar el "qué" que el "cómo". Si se utiliza un nombre que evite referirse a la implementación se estará conservando la abstracción de la estructura ya que la implementación está sujeta a cambios, de esta manera se describe que hace la estructura y no como lo hace. Por ejemplo es mas claro nombrar un procedimiento de acceso a datos *SeleccionarRegistro()* que *RealizarConsultaSelect(),* porque lo que importa (para que otra persona entienda el código) es que se supone que hace el método y no como lo hace.
* No utilice números o cadenas literales, como por ejemplo For i = 1 To 7. En su lugar, emplee constantes con nombre, del tipo For i = 1 To Enumeracion.length para que resulten fáciles de mantener y comprender.
* Los archivos y los nombres de carpetas, al igual que los nombres de procedimientos, deben describir claramente su finalidad.

|  |  |
| --- | --- |
| Archivo de Proyecto | - Utilizar estilo de Escritura Pascal - Siempre coincidir el nombre del ensamblado y el espacio de nombres raíz. Ejemplo:Empresa.SistemaOnline.csproj -> Empresa.SistemaOnline.dll -> namespace: Empresa.SistemaOnline |
| Archivo Fuente | * - Utilizar estilo de Escritura Pascal   Siempre coincidir el nombre de la clase y el nombre del archivo.  - Evitar incluir más de una clase, enumerador (global) o delegado (global) por archivo. Ejemplo: MiClase.cs => public class MiClase |
| Espacio de Nombre (Namespace) | - Estilo de Escritura Pascal - Tratar de coincidir el nombre ensamblado/proyecto. Ejemplo: namespace Empresa.SistemaWeb |
| Clase | Utilizar estilo de Escritura Pascal |
| Interfase | - Utilizar estilo de Escritura Pascal - Siempre agregar el prefijo "I" Ejemplo:  interface ICliente  { … } |
| Método | - Utilizar estilo de Escritura Pascal - Tratar de usar un verbo o par verbo-objeto.Ejemplo:  public void Ejecutar() { … }  private string ObtenerVersionEnsamblado(Assembly  libreria) { … } |
| Propiedad | - Utilizar estilo de Escritura Pascal - El nombre de la propiedad debe representar la entidad que devuelve. - Nunca utilizar los prefijos "Get" o "Set". Ejemplo:  public string Nombre  {  get{ … }  set{ … }  } |
| Campo (Público, Protegido o Interno) | - Utilizar estilo de Escritura Camel - Evitar el uso de campos no privados. - Usar propiedades en lugar de estos campos. Ejemplo:  public string nombre;  protected IList listaInterna; |
| Campo (Privado) | - Utilizar estilo de escritura tipo CamelCase  Ejemplo:  private string nombre; |
| Enumerador | - Utilizar estilo de Escritura Pascal Ejemplo:  public enum TiposCliente  {  Normal,  Frecuente  } |
| Variable (Interna) | - Utilizar estilo Camel. - Evitar el uso de nombres de un solo carácter tales como "x" o "y" a excepción de aquellos usados en la sentencia for. - Evitar enumerar nombre de variables como cadena1, cadena2, cadena3, etc. |
| Parámetro | - Utilizar estilo de escritura Camel Ejemplo:  public void Ejecutar(string texto, int  iteraciones)  { … } |

#### Formato

#### Declare una sola variable por línea

#### Evite albergar múltiples clases en un solo archivo.

#### Una clase debe estar definida en orden descendente de la siguiente manera: Variables Miembro, Constructores, Enumeraciones, Estructuras o Clases anidadas, Propiedades y por ultimo los Métodos.

#### Aísle la interfaz de implementación utilizando bloques *Region*. Recordar que escribiendo *#region* y apretando la tecla tab se crea una región.

* Utilice espacios antes y después de los operadores siempre que eso no altere la sangría aplicada al código:

**miVariable = 3; en vez de miVariable=3;**

#### Cuando escriba instrucciones SQL utilice mayúsculas para las palabras clave: SELECT, UPDATE, WHERE, FROM, etc.

#### No compare strings con “” o con string.empty, use la propiedad length del string:

**If(String.length == 0)**

**Tablas**

#### Cuando ponga nombres a tablas, hágalo en singular. Por ejemplo, use Empleado en lugar de Empleados.

#### Cuando ponga nombre a las columnas de las tablas, no repita el nombre de la tabla; por ejemplo, evite un campo llamado EstudianteApellido de una tabla llamada Estudiante. (Igual que con las propiedades de una clase).

#### No incorpore el tipo de datos en el nombre de una columna.

#### No ponga prefijos *sp* a los procedimientos almacenados, ya que se trata de un prefijo reservado para la identificación de procedimientos almacenados de sistemas.

#### Los nombres de los campos deben empezar con minuscula.

**Comentarios**

#### Los comentarios deben ser en español.

#### Use los comentarios para explicar el propósito del código. No los use como si fueran traducciones literales.

#### Utilizar /// para comentarios XML de las funciones y métodos. No olvidar que cuando se presiona tres veces slash (“/”), el editor generará los bloques de comentarios respectivos.

#### Evite comentarios que expliquen cosas obvias, en la mayoría de las cosas el código debe ser auto explicativo. Un buen código con buenas prácticas de nombrado no debe ser comentado.

### 4.2 Estándares de Diseño

Los estándares de diseño son ciertas pautas que se establecen para conseguir uniformidad en el desarrollo de páginas Web.

**Recomendaciones al utilizar Flash**

* Evitar el uso de Flash en la página inicial, ya que impide el acceso de robots para ser indexada por los buscadores.
* Colocar una imagen en el caso de que la presentación Flash no cargue

**Uso de Metatags**

Los **metatags** son marcas en lenguaje HTML que van en la parte superior del código fuente de cada pagina, a través de las cuales se entrega a los sistemas de indexación y búsqueda, la información minima para hacer una correcta interpretación del contenido que incluye.

Ej:

<title> Portal Musical </title>

<meta name=”title”> Portal Musical </meta>

**Normas para Incorporar Elementos Gráficos y Multimediales**

* Optimizar el peso de las imágenes: se debe bajar al máximo posible el peso de las imágenes; cuando esto no sea posible hacerlo por su tamaño, se debe reducir el número de colores disponibles y la resolución (72 dpi es la norma).
* El formato de las imágenes debe ser del tipo .jpg, jpeg, gif, png.
* Las imágenes no deben tener ningún tipo de borde.
* Usar el atributo ALT en imágenes: en el código HTML se debe usar el atributo ALT (texto alterno) en las imágenes para que éste se despliegue antes que las imágenes y facilite de esta forma la comprensión del contenido a los usuarios.

**Interoperabilidad**

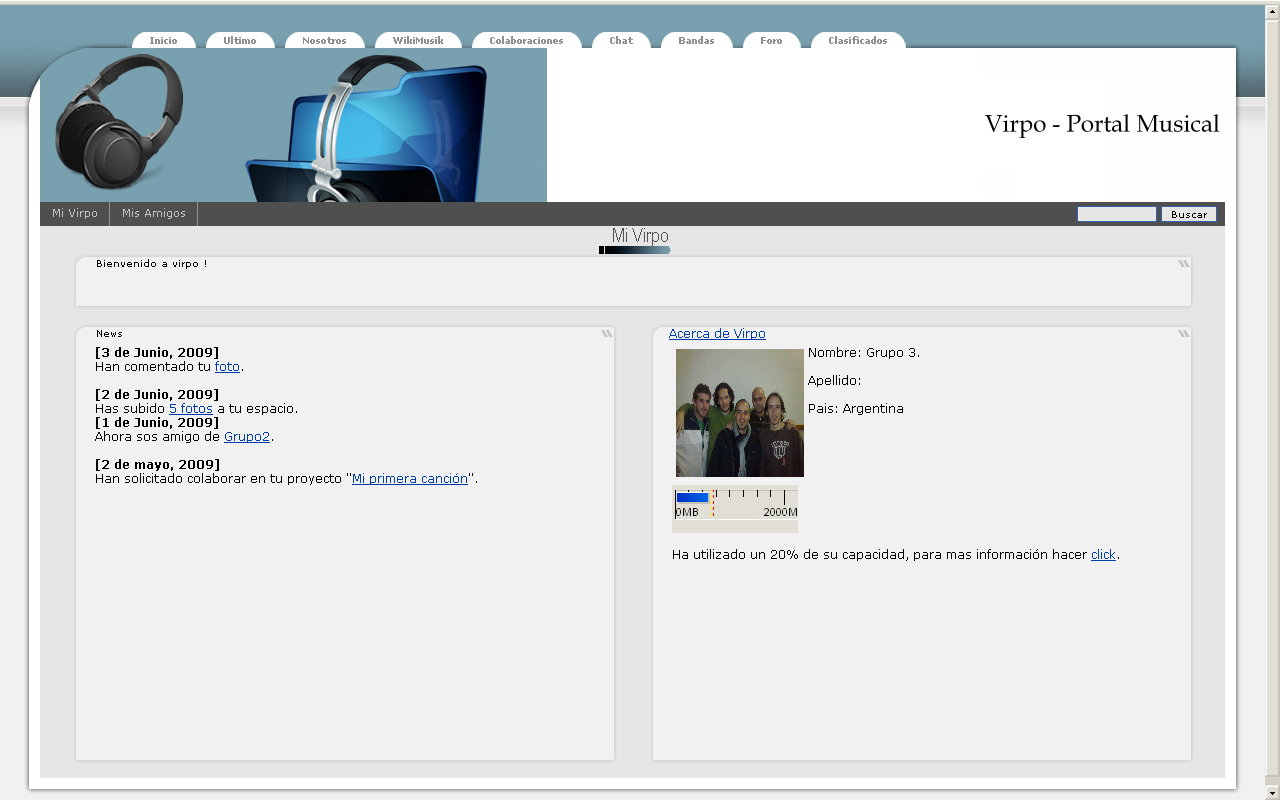
* Probar el sitio en los navegadores Microsoft Internet Explorer 7 y 8 y Mozilla Firefox 3.0

**Formato**

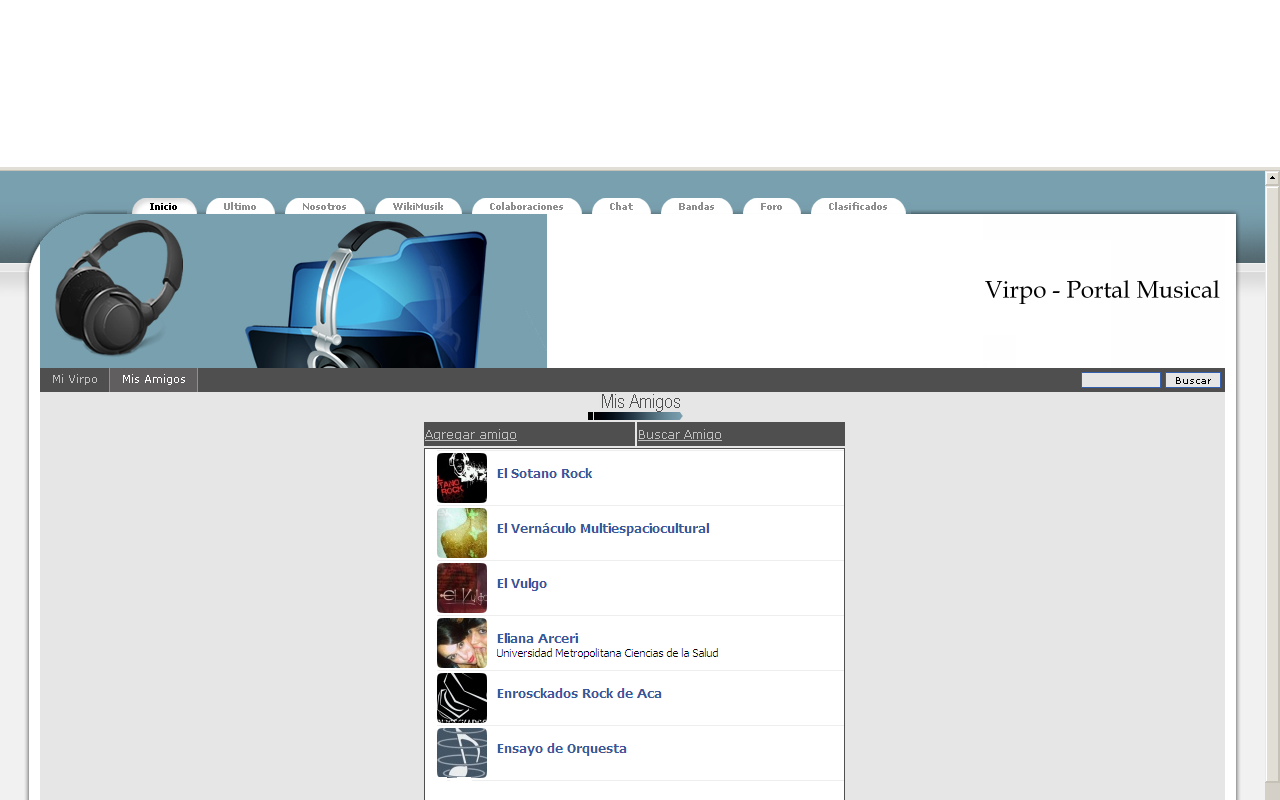
* Definir títulos de página con el estilo llamado *Title.*
* Definir subtítulos con el estilo *Subtitle.*
* Definir opción de menú con estilo *MenuOption.*
* Definir opción de submenú con estilo *SubMenuOption.*
* El nombre de la hoja de estilo principal será *Principal.css.*
* Clase de estilo para tablas: *BlueTable, BlackTable, DefaultTable.*
* El tipo de letra a utilizar deber ser Helvética para Menúes y Verdana para el cuerpo de la página.

### 4.3 Vistas de cada interfaz

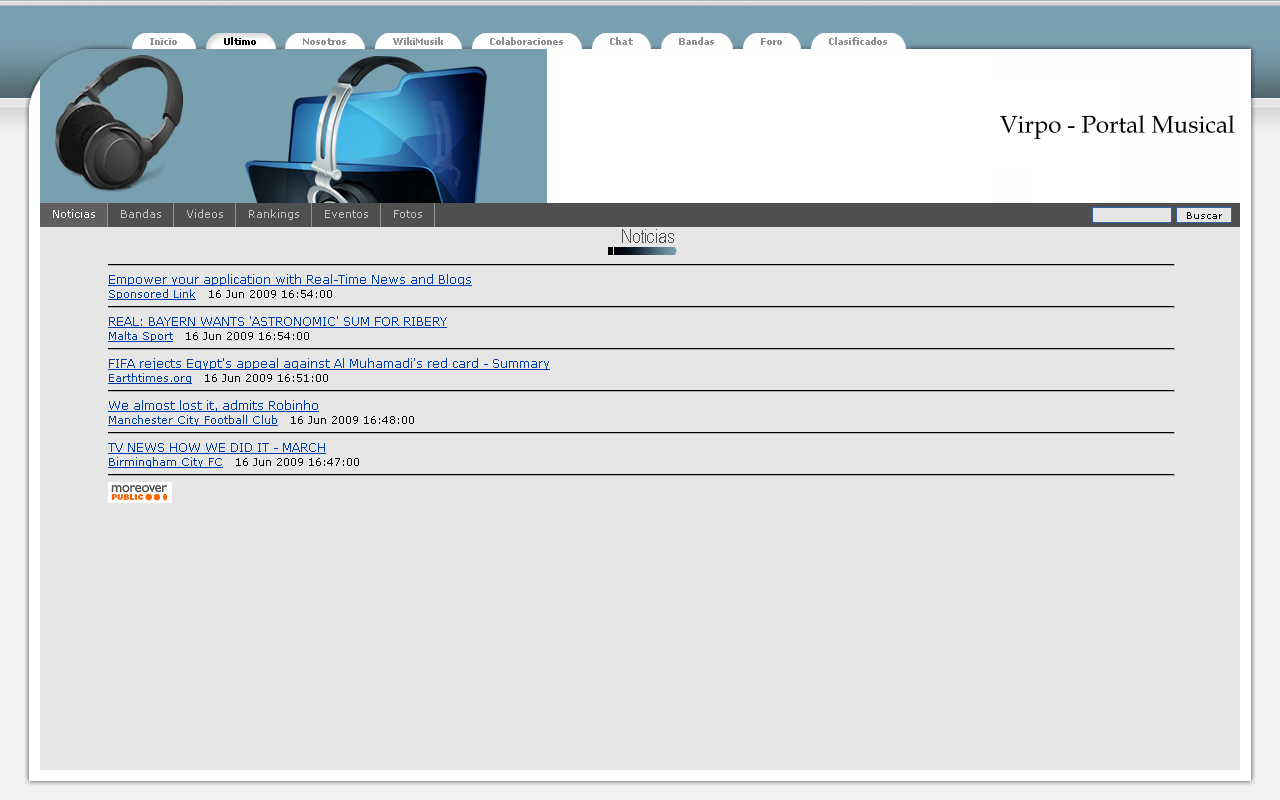
1 – Inicio – Mi Virpo



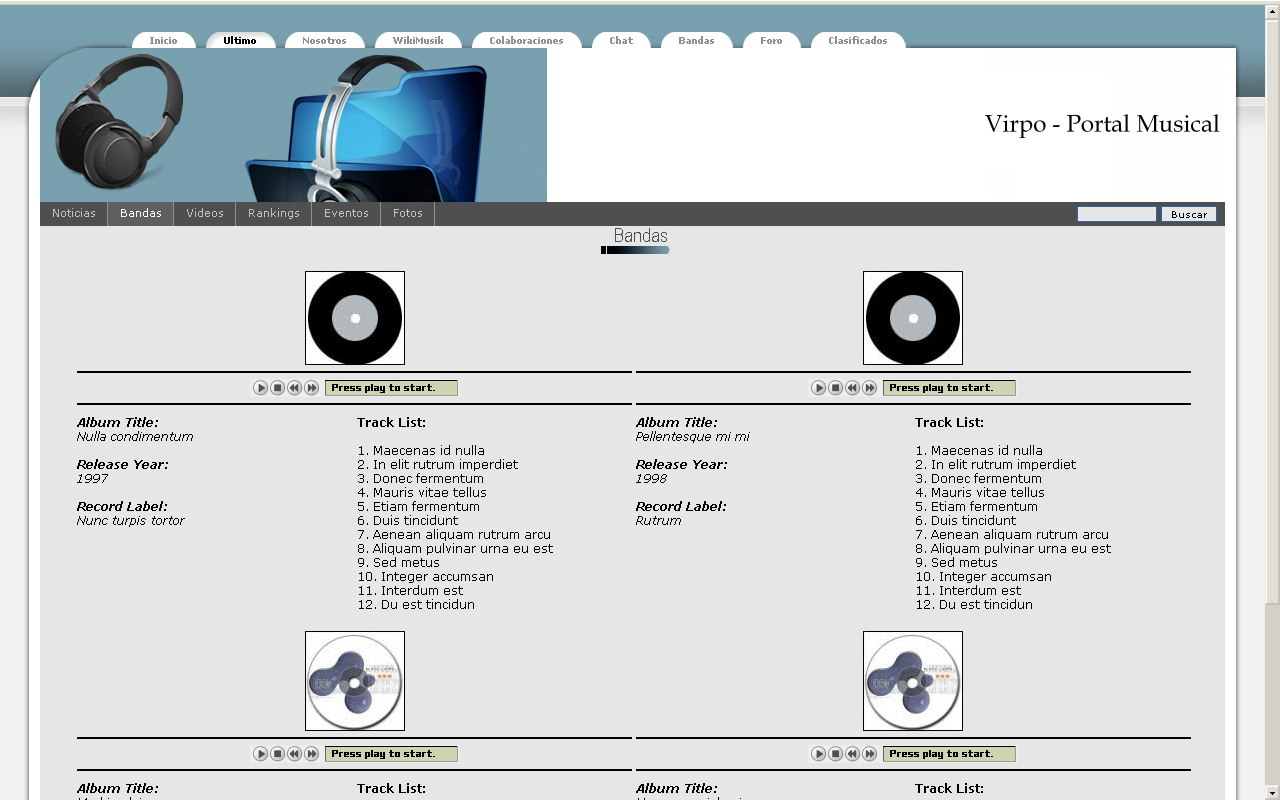
2 – Inicio – Mis Amigos



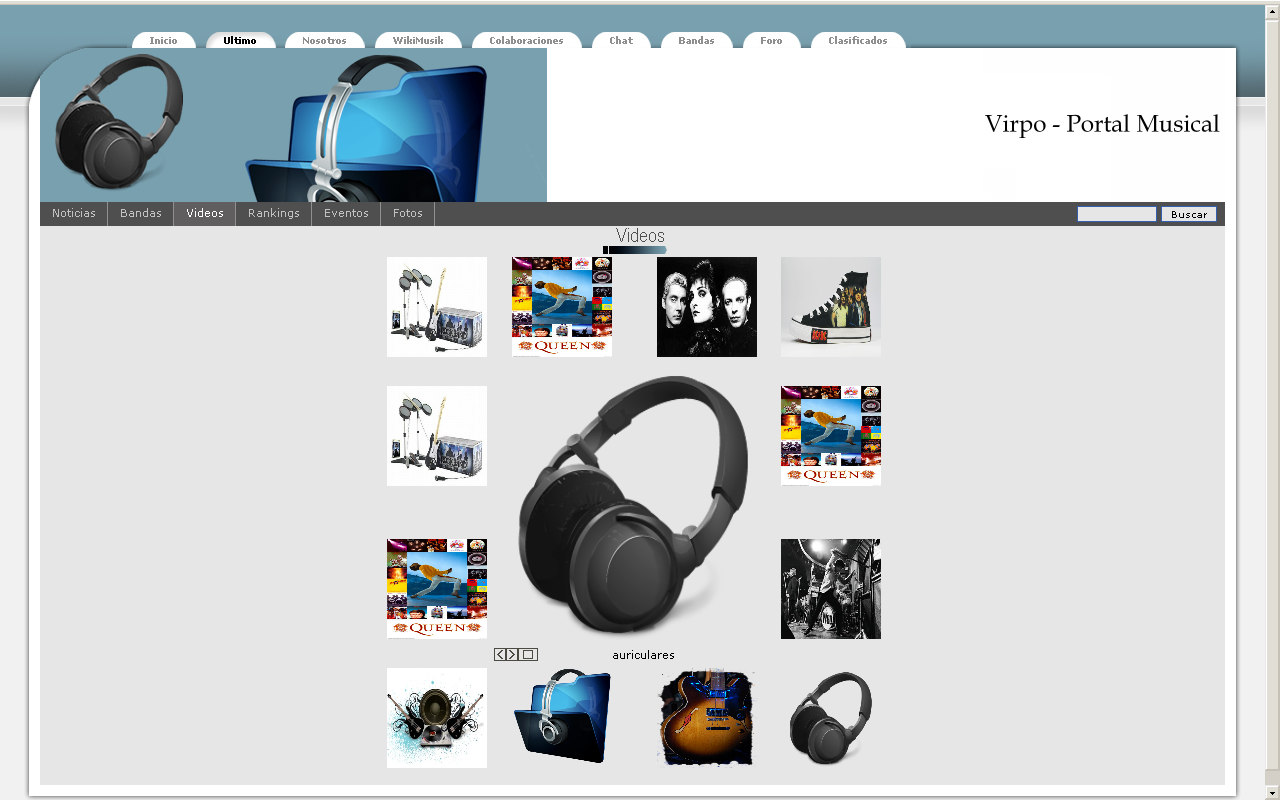
3 – Ultimo – Noticias



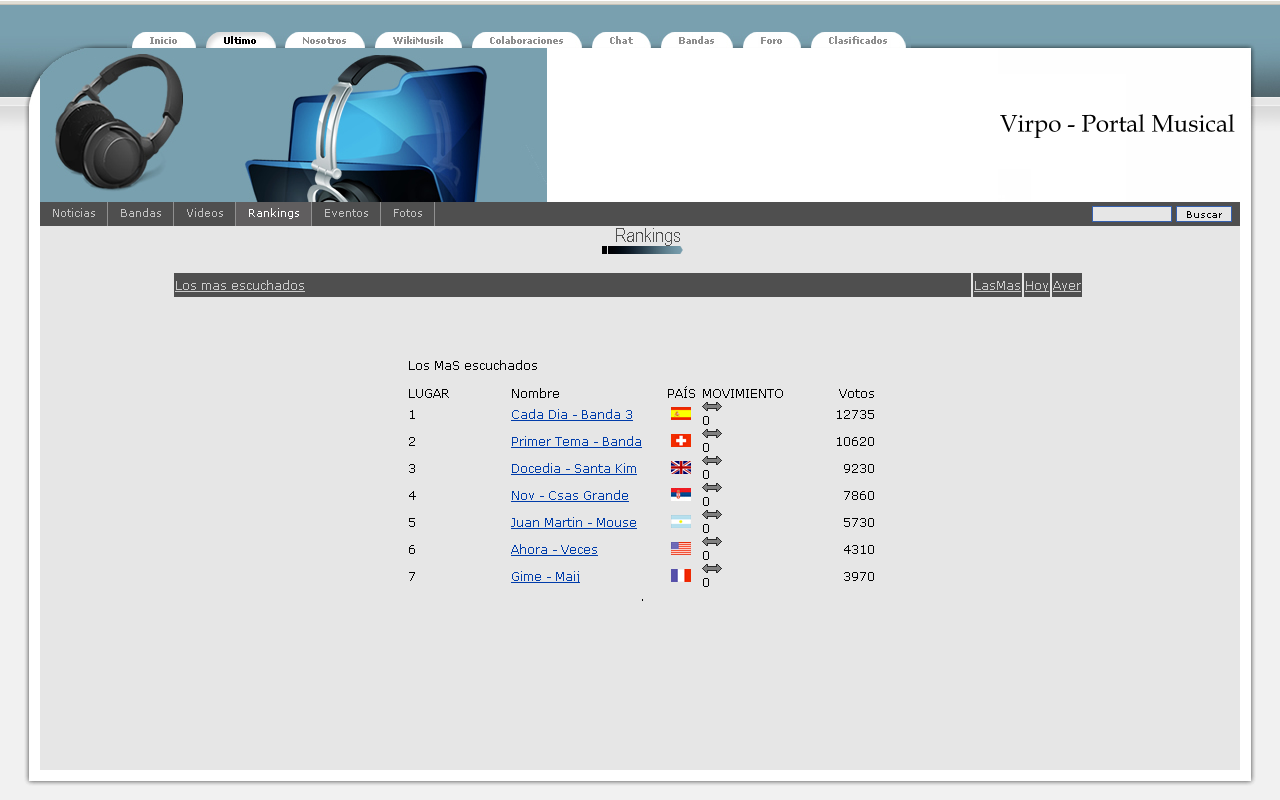
4 – Ultimo – Bandas



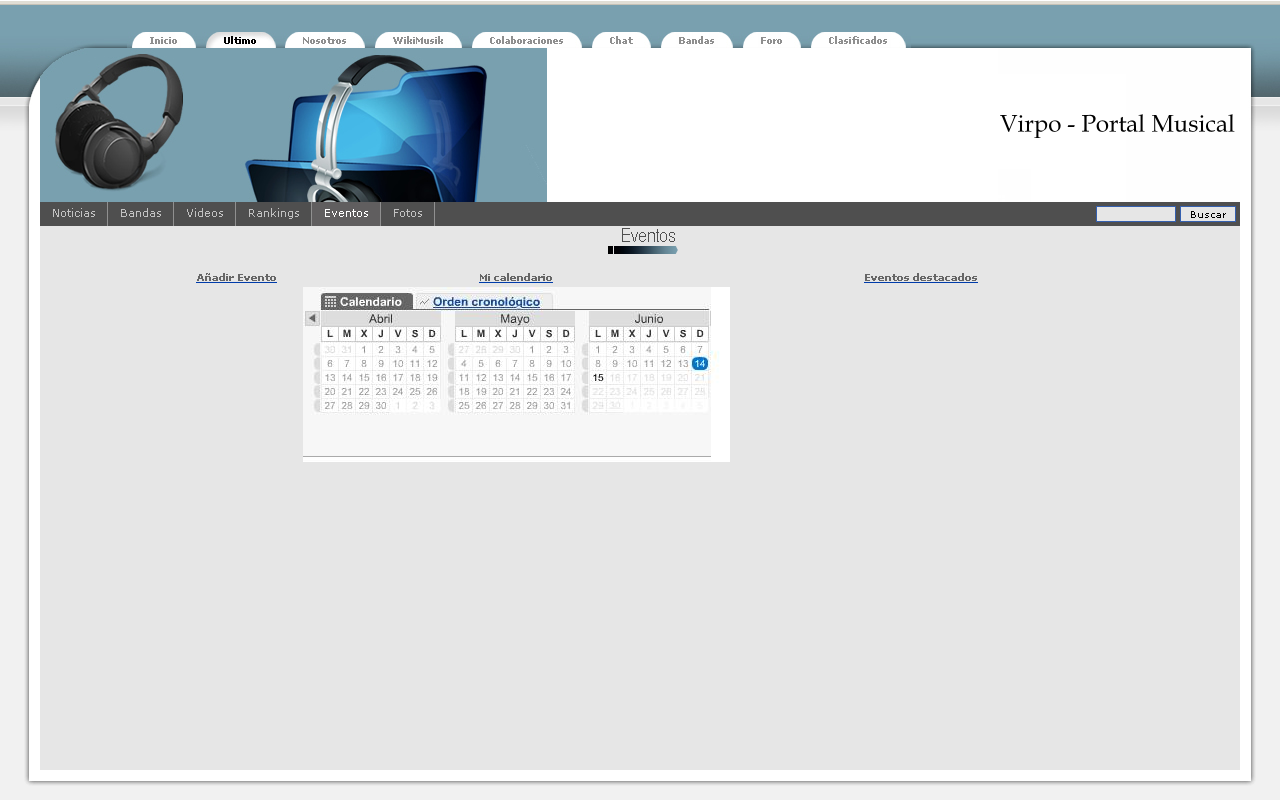
5 – Ultimo – Videos



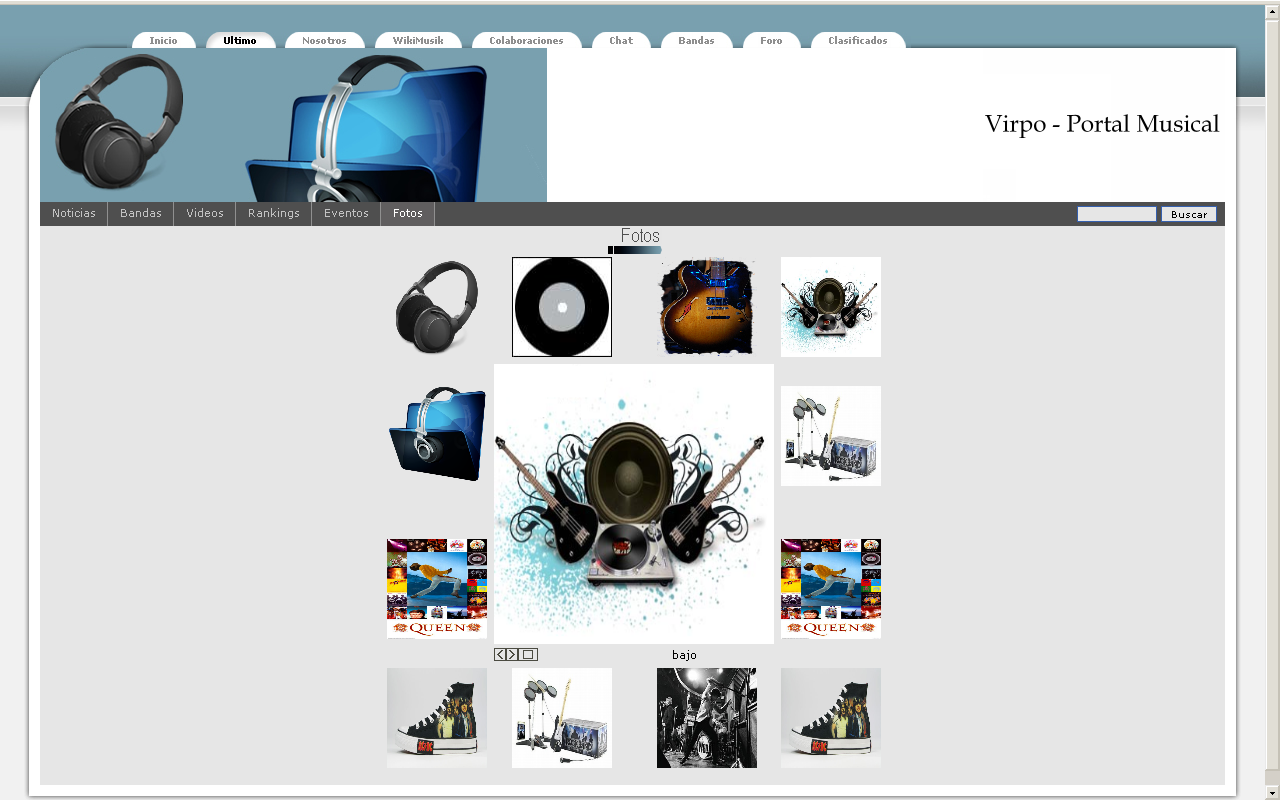
6 – Ultimo – Rankings



7 – Ultimo - Eventos



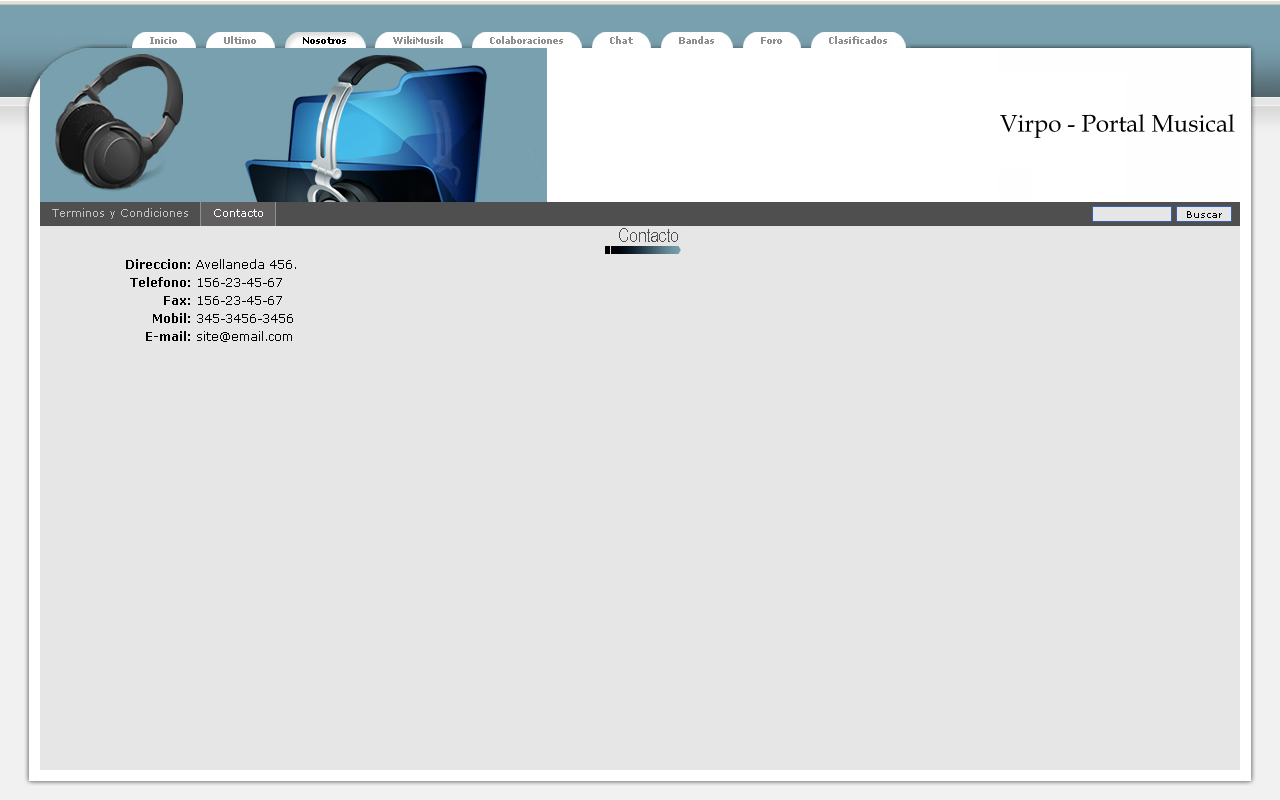
8 – Ultimo - Fotos



9 – Nosotros – Términos y Condiciones



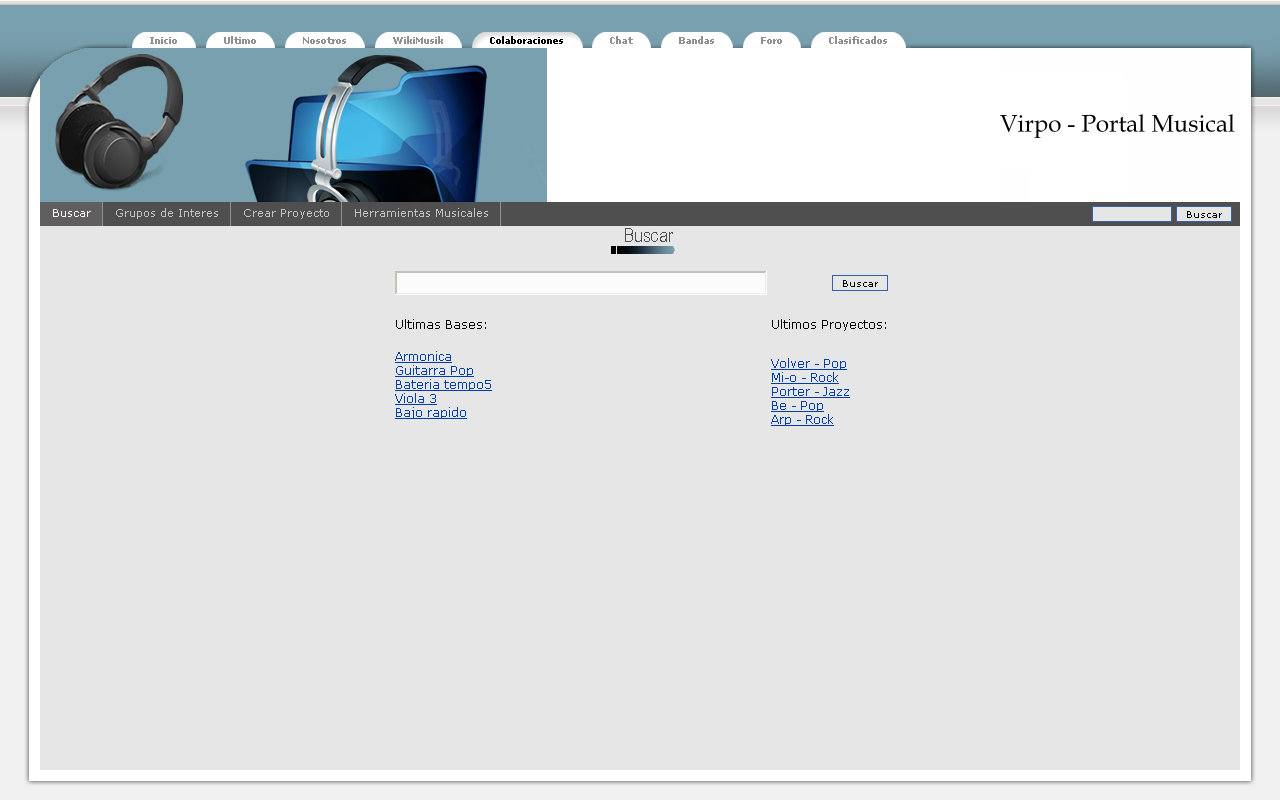
10 – Nosotros - Contacto



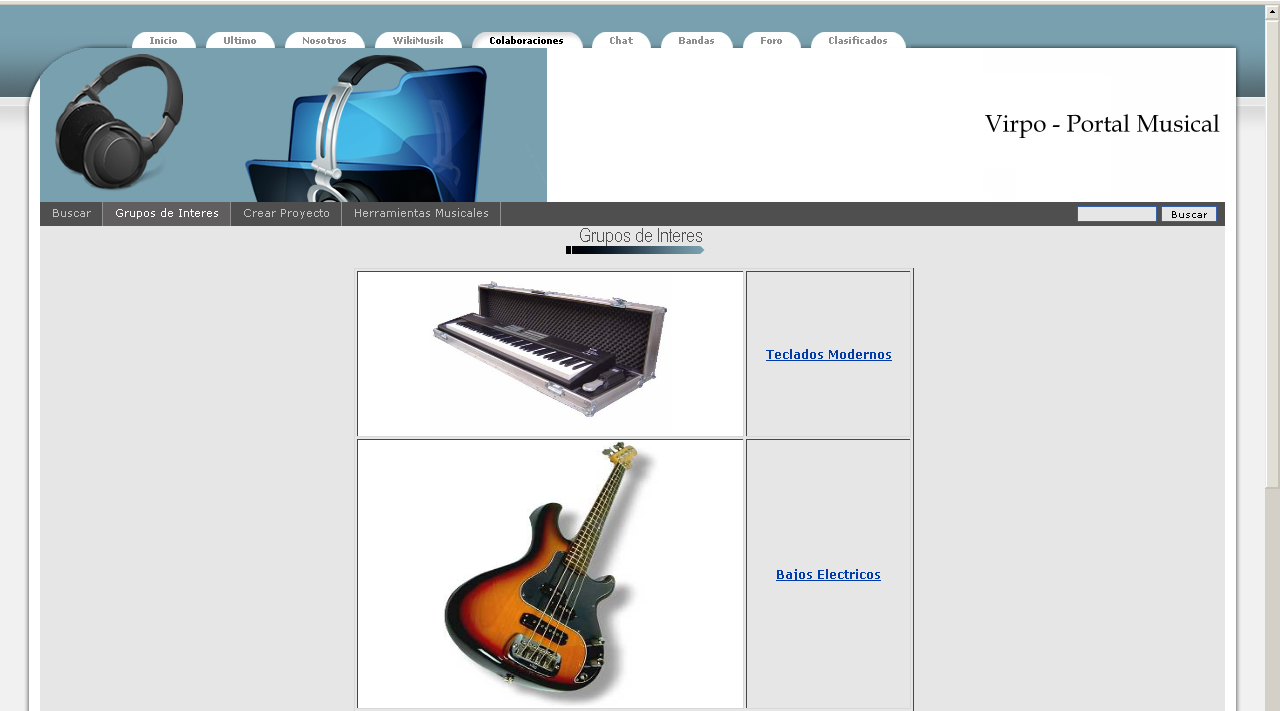
11 – Wikimusik



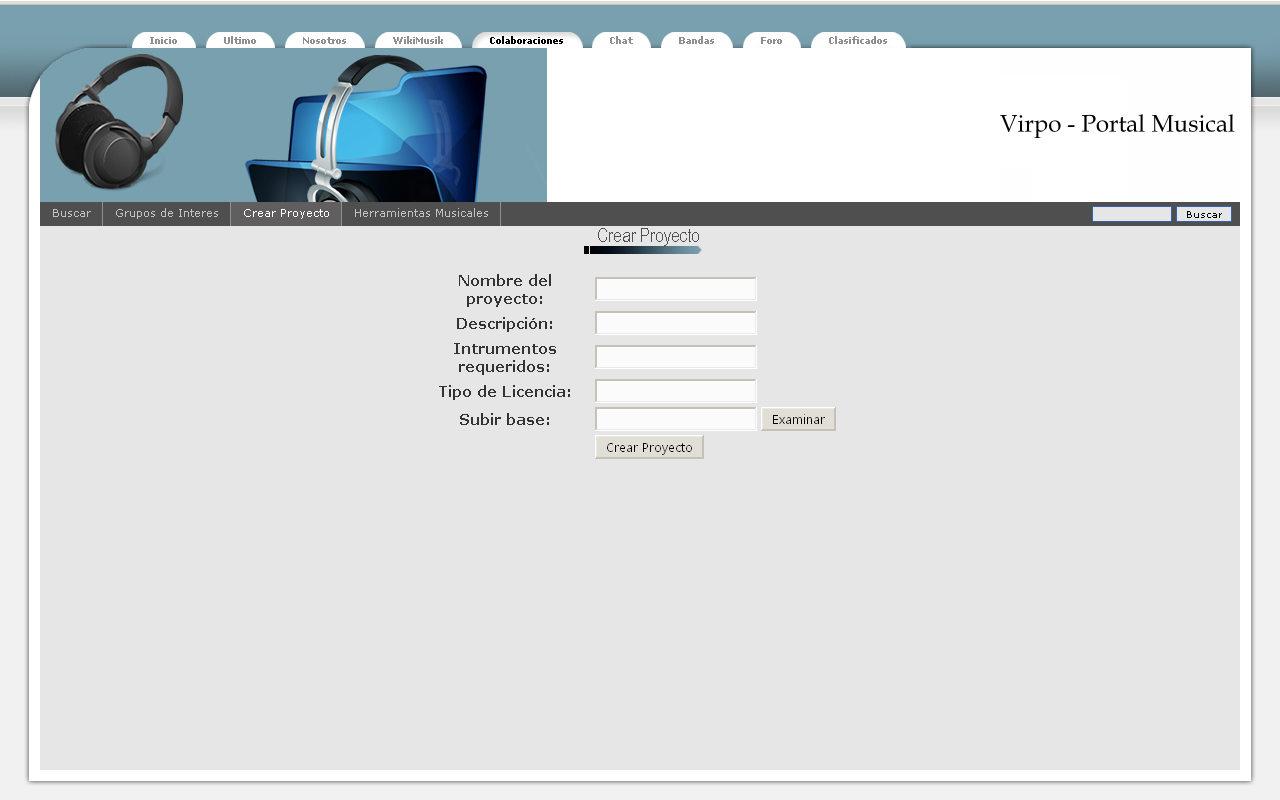
12 - Colaboraciones – Buscar



13 – Colaboraciones – Grupos de Interés



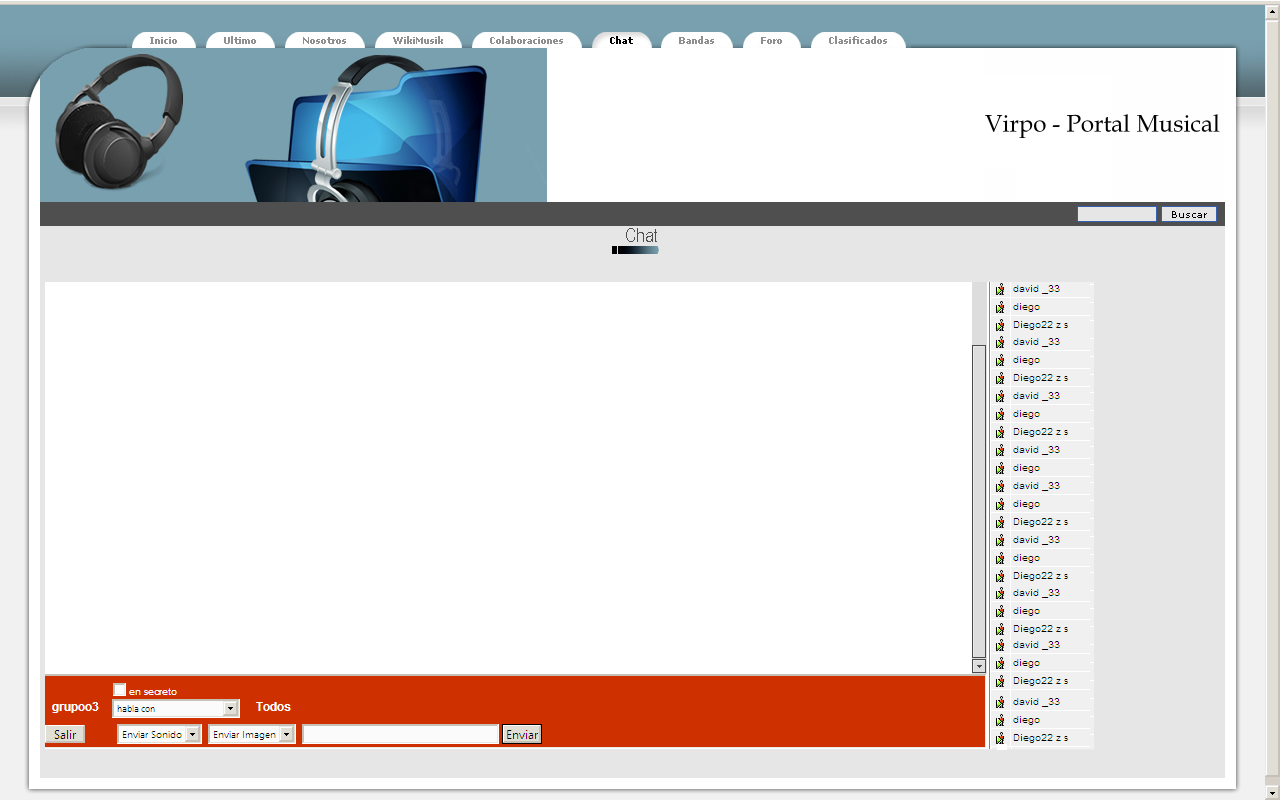
14 – Colaboraciones – Crear Proyecto



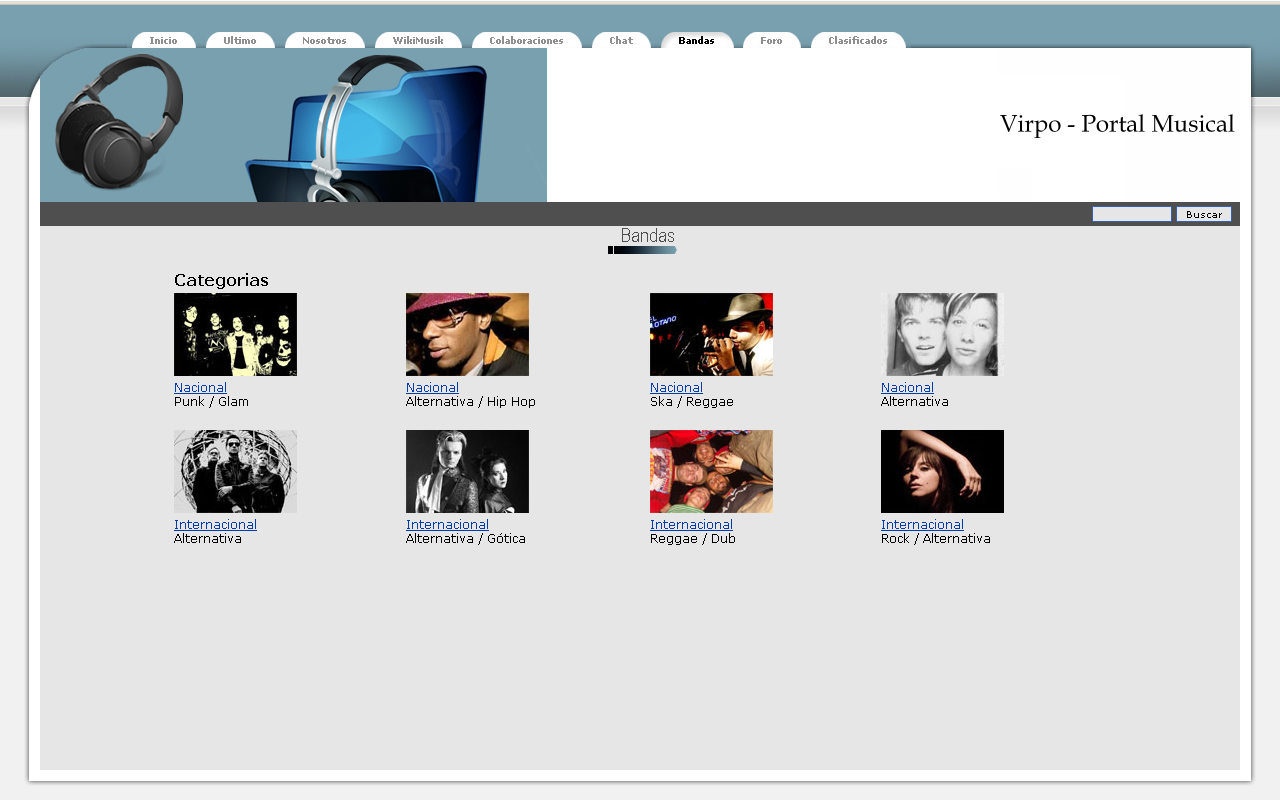
15 – Colaboraciones – Herramientas Musicales



16 – Chat



17 – Bandas



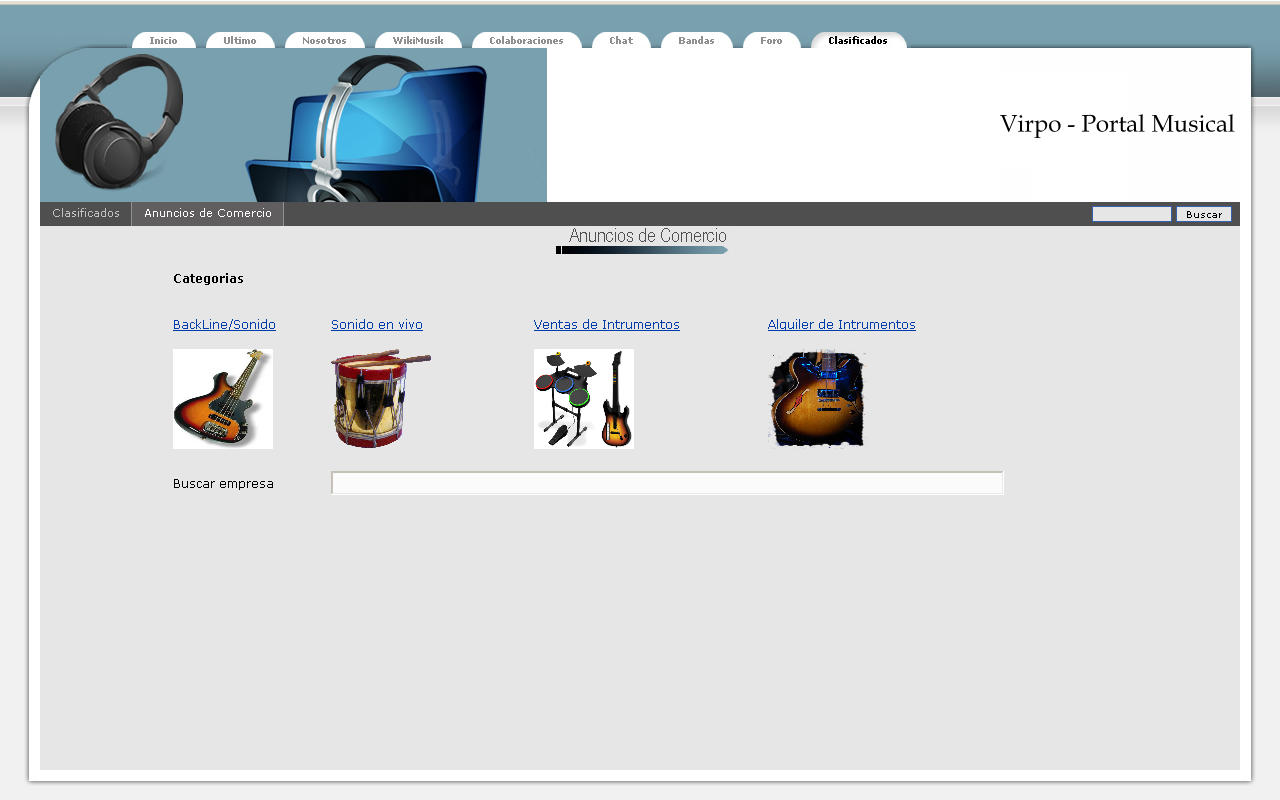
18 – Foro



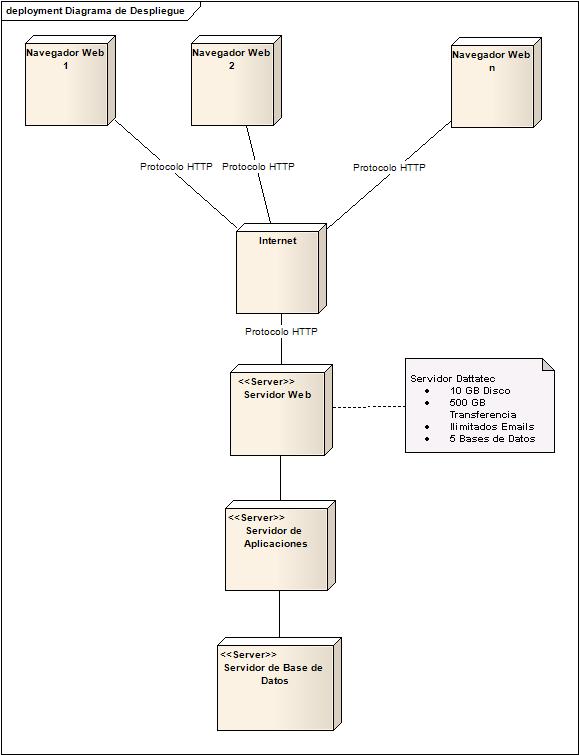
19 – Clasificados – Clasificados



20 – Clasificados – Anuncios de Comercio



# 5. Diagrama de despliegue.



# 6. Anexo

### Composiciones

Composiciones: es un término genérico utilizado para designar bases, canciones terminadas y canciones no terminadas.

Base: incluye un solo instrumento o voz.

Aclaración 1: la lógica de la música colaborativa es mantener almacenadas las bases que conforman una canción. Esto permite utilizar una misma base en tantos proyectos como se desee.

Canción terminada: es un proyecto concluido que genera una canción lista para ser votada por los usuarios del sitio.

Canción no terminada: es proyecto que no esta concluido e incluye una o mas bases.

Aclaración 2: En nuestro portal se mantendrán almacenadas canciones terminadas, no terminadas y bases.

### Creative Commons

Creative Commons es una [organización no gubernamental](http://es.wikipedia.org/wiki/Organizaci%C3%B3n_no_gubernamental) [sin ánimo de lucro](http://es.wikipedia.org/wiki/Organizaci%C3%B3n_sin_%C3%A1nimo_de_lucro) que desarrolla planes para ayudar a reducir las barreras legales de la [creatividad](http://es.wikipedia.org/wiki/Creatividad), por medio de nueva [legislación](http://es.wikipedia.org/wiki/Legislaci%C3%B3n) y nuevas [tecnologías](http://es.wikipedia.org/wiki/Tecnolog%C3%ADa). Fue fundada por [Lawrence Lessig](http://es.wikipedia.org/wiki/Lawrence_Lessig), profesor de [derecho](http://es.wikipedia.org/wiki/Derecho) en la [Universidad](http://es.wikipedia.org/wiki/Universidad_de_Stanford) de Stanford y especialista en [ciberderecho](http://es.wikipedia.org/wiki/Ciberderecho), que la presidió hasta marzo de 2008. Creative Commons (en [español](http://es.wikipedia.org/wiki/Idioma_espa%C3%B1ol): «bienes comunes creativos») es también el nombre dado a las [licencias](http://es.wikipedia.org/wiki/Licencia) desarrolladas por esta organización.

Las licencias Creative Commons permiten que los autores de trabajos susceptibles a ser protegidos por las leyes de propiedad intelectual (libros, música, artículos en blogs, fotos, vídeos, etc.) puedan dar permiso a terceros para que usen gratuita y legalmente sus trabajos, siempre que respeten una serie de condiciones, sin perder sus derechos de propiedad intelectual. Por esta razón se dice que en las licencias CC hay “algunos derechos reservados”, ya que se abre un abanico de posibilidades entre la protección total, es decir, “todos los derechos reservados” y la renuncia a todos los derechos sobre el trabajo, o lo que es lo mismo, optar por el dominio público (public domain). Hay un total de seis licencias Creative Commons para escoger:

attributionReconocimiento (Attribution): En cualquier explotación de la obra autorizada por la non commerciallicencia hará falta reconocer la autoría.

No Comercial (Non commercial): La explotación de la obra queda limitada a usos no comerciales.

no derivate worksSin obras derivadas (No Derivate Works): La autorización para explotar la obra no incluye la transformación para crear una obra derivada.

share alikeCompartir Igual (Share alike): La explotación autorizada incluye la creación de obras derivadas siempre que mantengan la misma licencia al ser divulgadas.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Con estas cuatro condiciones combinadas se pueden generar las seis licencias que se pueden escoger:   |  |  |  |  | | --- | --- | --- | --- | | by |  |  | Reconocimiento (by): Se permite cualquier explotación de la obra, incluyendo una finalidad comercial, así como la creación de obras derivadas, la distribución de las cuales también está permitida sin ninguna restricción. | | by-nc |  |  | Reconocimiento - NoComercial (by-nc): Se permite la generación de obras derivadas siempre que no se haga un uso comercial. Tampoco se puede utilizar la obra original con finalidades comerciales. | | by-nc-sa |  |  | Reconocimiento - NoComercial - CompartirIgual (by-nc-sa): No se permite un uso comercial de la obra original ni de las posibles obras derivadas, la distribución de las cuales se debe hacer con una licencia igual a la que regula la obra original. | | by-nc-nd |  |  | Reconocimiento - NoComercial - SinObraDerivada (by-nc-nd): No se permite un uso comercial de la obra original ni la generación de obras derivadas. | | by-sa |  |  | Reconocimiento - CompartirIgual (by-sa): Se permite el uso comercial de la obra y de las posibles obras derivadas, la distribución de las cuales se debe hacer con una licencia igual a la que regula la obra original. | | by-nd |  |  | Reconocimiento - SinObraDerivada (by-nd): Se permite el uso comercial de la obra pero no la generación de obras derivadas. | |