Instituto Tecnológico de Costa Rica

Área Académica de Ingeniería en Computadores

Programa de Licenciatura en Ingeniería en Computadores

Curso: CE-3101 Bases De Datos

Realizado por:

Frander Granados Vega 201117284
Maikol Barrantes García 201230364
Julio Sánchez Jiménez 201125192

Profesor:

Marco Hernández

Fecha: Cartago, Noviembre 21, 2015

Indice

Introducción

En el presente proyecto se realiza una aplicación de escritorio y una web. Mediante un REST API se realiza la comunicación con la base de datos alojada en un servidor externo. La aplicación funciona como un reproductor musical en su aplicación de escritorio, pero con la característica de que puede sincronizar la música con CusucoMusic y escucharla desde la web en cualquier parte del mundo, se implementa una red social con gustos musicales donde se puede compartir tus mejores canciones con tus amigos.

Descripción de todas las bibliotecas externas utilizadas.

Para la aplicación de escritorio se hizo uso del lenguaje de programación Java, se describen las bibliotecas externas utilizadas con Java.

Metadatos de los archivos mp3

Para leer los metadatos de los archivos mp3 utilizamos la biblioteca *mp3agic*, esta biblioteca permite manipular las etiquetas, leer y modificarlas; la podemos encontrar en:
https://github.com/mpatric/mp3agic
Y los ejemplos se pueden encontrar en:
https://github.com/mpatric/mp3agic-examples/blob/master/src/main/java/com/mpatric/mp3agic/example/Example.java

Reproducir Archivos mp3

Para reproducir archivos mp3 en java utilizamos la biblioteca JLayer, basados en el código de la respuesta de Durandal, en el siguiente link:
http://stackoverflow.com/questions/12057214/jlayer-pause-and-resume-song

Base de datos local

Otra biblioteca utilizada fue *Java Database Connectivity*, más conocida por sus siglas JDBC, es una API que permite la ejecución de operaciones sobre bases de datos desde el lenguaje de programación Java, independientemente del sistema operativo donde se ejecute o de la base de datos a la cual se accede, utilizando el dialecto SQL del modelo de base de datos que se utilice.

JSON.simple es un simple conjunto de herramientas Java para JSON. Puede utilizar JSON.simple para codificar o decodificar texto JSON.
Se puede encontrar en el siguiente link: https://github.com/fangyidong/json-simple

HttpClient y HttpClient

Estas bibliotecas son utilizadas para acceder a recursos HTTP.

El proyecto Commons HttpClient ahora el final de la vida, y ya no se está desarrollando. Ha sido reemplazado por el proyecto Apache HttpComponents en sus HttpClient y HttpCore módulos, que ofrecen un mejor rendimiento y una mayor flexibilidad.

Se puede encontrar en el siguiente link: https://hc.apache.org/downloads.cgi

commons-login

Esta biblioteca de apache para Java permite sin importar la aplicación el registro en tiempo de ejecución.

Se puede obtener en el siguiente link: https://commons.apache.org/proper/commons-logging/download_logging.cgi

Alternativas de solución consideradas y justificación de la seleccionada.

Para solucionar el problema del servidor se hace uso de un Raspberry Pi 2, este dispositivo es un ordenador de placa reducida o de bajo coste desarrollado en Reino Unido. El Raspberry Pi 2 trae consigo un CPU de 900 MHz quad-core ARM Cortex A7, 1 GB de memoria SDRAM, 4 puertos USB, almacenamiento por MicroSD, como dato extra el Raspberry Pi 2 es de bajo consumo de energía, a demás que su costo es de $35. A pesar de tener características un poco limitadas a nivel de hardware para efectos educativos y prácticos en la entrega de este proyecto funciona a la perfección, una alternativa era el usar algún servicio web como BlueMix, Azure, AppEngine, pero para un mejor manejo de nuestra información y como parte del conocimiento que se podía obtener decidimos crear nuestro propio servicio.

Se instala un servidor GNU/Linux Debian Jessie en el Raspberry Pi 2, se toma la decision por aspectos de licencia y caracterizaras de hardware del servidor, la mejor opción fue el uso de Software libre. Para el problema de la salida a Internet y que la aplicación se pueda conectar desde cualquier parte se tiene una conexión a Internet con IP publica y se adquiere un dominio gratuito por 1 año, con esto se logra solucionar de manera económica y simple el uso de un servidor externo.

En en servidor se instalan algunos paquetes necesarios como apache y MariaDB.

MariaDB es un sistema de gestión de bases de datos derivado de MySQL con licencia GPL. Está desarrollado por Michael Widenius fundador de MySQL y la comunidad de desarrolladores de software libre. Una alternativa era el uso de SQL Server, Oracle o DB2 pero por problemas de compatibilidad SQL Server no funcionaba, y fue mas sencillo el uso de MySQL.

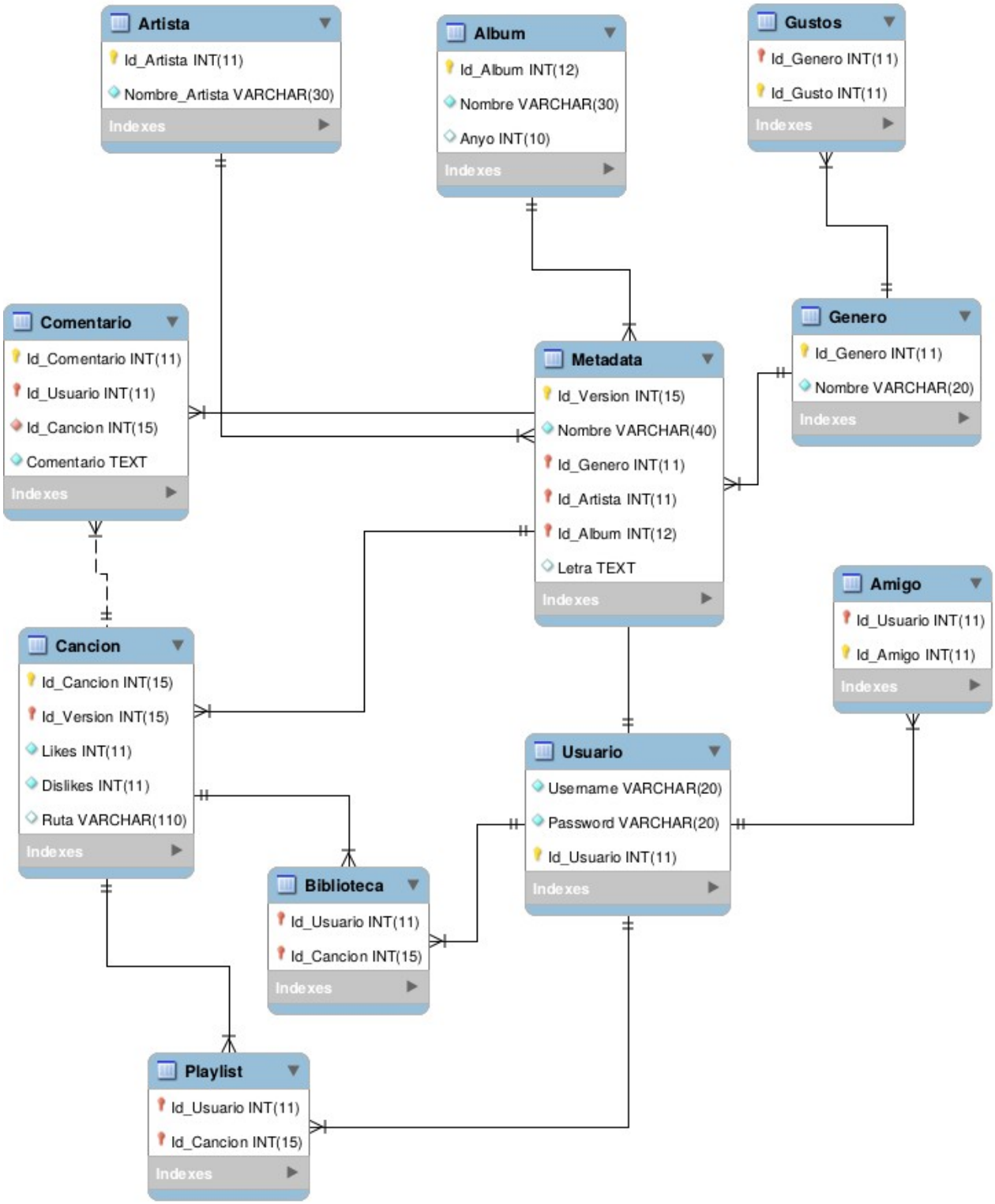
Para el proyecto se pedía que existiera una aplicación de escritorio y una versión web, para la versión web se tenían varias opciones como php, ASP .NET , ruby, java. En caso de escoger ruby se debía primero aprender el lenguaje lo que significaba invertir mucho tiempo en algo que no era lo que se quería, con java applet no porque se encuentra obsoleto, ASP .NET presentaba algunos problemas al tratar de integrar la aplicación web con el servidor, por lo que la mejor opción fue el uso de php y JavaScript.

Para realizar este proyecto se requería el uso de algún lenguaje que facilitada el manejo de web, como solución se escoge php, este es un lenguaje de programación de uso general, el código normalmente esta del lado del servidor. Originalmente diseñado para el desarrollo web de contenido dinámico. El código es interpretado por un servidor web con un módulo de procesador de PHP que genera la página Web resultante.
Puede ser usado en la mayoría de los servidores web al igual que en casi todos los sistemas operativos y plataformas sin ningún costo.

PHP se considera uno de los lenguajes más flexibles, potentes y de alto rendimiento conocidos, lo que ha atraído el interés de múltiples sitios con gran demanda de tráfico, como Facebook, para optar por el mismo como tecnología de servidor. Este lenguaje forma parte del software libre publicado bajo la licencia PHP, que es incompatible con la Licencia Pública General de GNU debido a las restricciones del uso del término *PHP.*

La ventaja de usar php es su similitud con lenguajes estudiados en cursos anteriores y fue muy sencillo y rápido aprender el lenguaje.
En cuanto a la aplicación se desarrolla en Java, se tenían varias opciones como C, C++ o python, pero por cuestiones de facilidad y tiempo se desarrolla en Java, a demás de la cantidad de las bibliotecas disponibles para realizar la conexión con la base de datos.

Diagrama Entidad-Asociación de la Base de Datos.



RESTful API

Para la Restful API se trabaja con Phalcon.

Phalcon es una nueva alternativa en frameworks para PHP. Nuestra misión es proporcionarte una herramienta avanzada para desarrollar sitios y aplicaciones web sin preocuparte por el rendimiento del framework.

Las extensiones de PHP requieren un método diferente de instalación a los frameworks o bibliotecas tradicionales. Puedes descargar tanto un paquete binario para tu sistema o compilarlo desde el código fuente.

Phalcon es multiplataforma y existe para Windows, Linux y Mac.

Para usar Phalcon en Windows debes descargar un DLL y ubicarlo en el directorio de extensiones. Edita el php.ini y agrega al final:

```
extension=php_phalcon.dll
```

Finalmente, reinicia el servidor web.

En un sistema Linux/Solaris/Mac puedes compilar e instalar la extensión fácilmente desde la fuente del repositorio:

Compilando la extensión:

```
git clone git://github.com/phalcon/cphalcon.git
cd cphalcon/build
sudo ./install
```

Añadiendo la extensión a php.ini

```
extension=phalcon.so
```

Reiniciando el servidor web.

En el siguiente link se tiene un tutorial simple de su uso:

https://docs.phalconphp.com/es/latest/reference/tutorial.html#creando-un-proyecto

Para crear la API seguimos el siguiente tutorial:
https://docs.phalconphp.com/es/latest/reference/tutorial-rest.html

Utilizando métodos HTTP mediante una Restful Api se puede:

GET para recuperar y buscar datos

POST para añadir datos

PUT para actualizar los datos

DELETE para eliminar los datos

Metodo	URL	Accion
GET	/api/albums/{name}	Busca y retorna el Id_Album del album con el nombre name
POST	/api/albums	Agrega un nuevo album a la base de datos
GET	/api/artistas/{name}	Busca y retorna el Id_Artista del artista con el nombre name
POST	/api/artistas	Agrega un nuevo artista a la base de datos
GET	/api/generos/{name}	Busca y retorna el Id_Genero del genero con el nombre name
POST	/api/generos	Agrega un nuevo genero a la base de datos
PUT	/api/canciones/{id}	Modifica la cancion con el Id_Cancion id
GET	/api/metadatas/{song}/{artist}	Busca y retorna el Id_Version del metadata de la cancion cuyo nombre es song y el artista cuyo nombre es artist
POST	/api/metadatas	Agrega un nuevo metadata a la base de datos
POST	/api/bibliotecas	Agrega un nuevo registro biblioteca a la base de datos
GET	/api/canciones/{song}/{artist}	Busca y retorna el Id_Cancion de la cancion cuyo nombre es song y el artista cuyo nombre es artist
GET	/api/canciones/{song}/{artist}/{user}	Busca y retorna el Id_Cancion de la cancion cuyo nombre es song, el artista cuyo nombre es artist y el nombre de usuario user
GET	/api/canciones/{user}	Busca y retorna el Id_Cancion

		de todas las canciones cuyo usuario sea user
POST	/api/canciones	Agrega una nueva cancion a la base de datos
GET	/api/usuarios/{name}/{pass}	Busca y retorna el Id_Usuario de la usuario cuyo nombre de usuario es name y el password pass
GET	/api/usuarios/{name}	Busca y retorna el Id_Usuario de la usuario cuyo nombre de usuario es name
GET	/api/usuarios	Agrega un nuevo usuario a la base de datos

Actividades realizadas por estudiante.

Frander Granados Vega
Se instala GNU/Linux Debian Jessie en laptop. 30-09-2015 19:00:00 a 22:00:00
Se instalan paquetes necesarios como apache, php, MySQL en servidores 1-10-2015 22:00:00 a 23:00:00
Se configura servidor para salir por Internet y se configura el router en el puerto 80 para darle salida a Internet al servidor. 2-10-2015 12:00:00 a 13:00:00
Por problemas con la laptop se instala nuevo servidor sobre un Raspberry Pi 2 y se configura como el inicial. 5-10-2015 17:00:00 a 20:00:00
Se investiga sobre PHP y se hace hola mundo para probar su funcionamiento. 5-10-2015 20:00:00 a 21:00:00
Se trabaja sobre RestFull en PHP para la conexión con la base de datos 12-10-2015 13:00:00 a 15:00:00
Se trabaja sobre la red social con PHP 14-10-2015 19:00:00 a 22:00:00
Se logra loguear y registrar usuarios en la base de datos 15-10-2015 23:00:00 a 23:45:00
Se trabaja en la API RestFull 25-10-2015 13:00:00 a 16:00:00
Se hacen cambios relacionados a la parte social, se agregan amigos 7-11-2015 13:00:00 a 14:00:00

Maikol Barrantes García

Se trabaja sobre la aplicación de escritorio, se obtienen los metadatos de los mp3 y las caratulas de los mp3 5-10-2015 17:12:00 a 17:58:00

Se crea interfaz gráfica de la cola de las canciones 5-10-2015 20:05:00 a 6-10-2015 0:51:00

Se agregan las funciones play, pausa y reanudar la reproducción de mp3, se

cambia orden de la cola gráficamente
6-10-2015 21:47:00 a 7-10-2015 4:30:00
Se trabaja en primera version del Restful Api 26-10-2015 18:00:00 a 26-10-2015 22:00:00

Se trabaja en Restful Api, Get y Post 27-10-2015 13:00:00 a 18:00:00

Se trabaja en reproductor, se borra canción, se modifica agregar canción, iconos e imagenes 27-10-2015 13:00:00 a 18:00:00

Base de datos local 29-10-2015 15:00:00 a 21:00:00

Julio Sánchez Jiménez
Se preparan paquetes de software.
Se instala MariaDB, PHP5, Apache2 en el equipo destinado como uso de servidor. 7-10-2015

Se crea una primera versión de la base de datos. 8-10-2015

Se trabaja en la creación del HTML y CSS. Se crea una nueva base de datos con un diseño mejorado. Se realizan pruebas a la base de datos. 16-10-2015

Se trabaja en la integración de un reproductor y una lista de reproducción 17-10-2015

Se finaliza una primera versión del reproductor web 20-10-2015

Se crean consultas con php para obtener rutas y nombres de canciones asociadas al usuario. 24-10-2015

Se integra el reproductor con demás componentes del website y base de datos. Se detallan mejor las consultas. 25-10-2015

Se investiga sobre MusicBrainz. 1-11-2015

Se inician pruebas de consultas a MusicBrainz. 6-11-2015

Se reorganizan elementos del website (HTML,php y css) para permitir cargas dinámicas, y cambio de contenido sin perjudicar la reproducción. 8-11-2015

Se realizan modificaciones al css y archivos php encargados de la carga de los contenidos. 14-11-2015

Se realiza la integración de la biblioteca de los amigos con la del usuario. 20-11-2015

Problemas encontrados:

Problema: Aunque tenemos acceso a todos los archivos del sistema, no se pueden seleccionar las carpetas completas, solamente archivos, tampoco podemos seleccionar varios archivos al mismo tiempo.

Solución: Para que el JFileChooser pueda seleccionar cualquier directorio o archivos específicos debemos indicárselo con la siguiente sentencia:

```
JFileChooser fileChooser = new JFileChooser();
fileChooser.setFileSelectionMode(JFileChooser.FILES_AND_DIRECTO
RIES);
```

Cambiar el tamaño de las imágenes en java (portadas de los discos)

Problema: Cuando creamos un label con un icono, este no cambia el tamaño de la imagen sino que solamente la corta.
Solución: Para cambiar el tamaño de las imágenes, se utilizó el código que proporciona Ocracoke en la página:
http://stackoverflow.com/questions/9417356/bufferedimage-resize
El código es el siguiente:

```
public static BufferedImage resize(BufferedImage img, int newW,
int newH) {
    Image tmp = img.getScaledInstance(newW, newH,
Image.SCALE_SMOOTH);
    BufferedImage dimg = new BufferedImage(newW, newH,
BufferedImage.TYPE_INT_ARGB);

    Graphics2D g2d = dimg.createGraphics();
    g2d.drawImage(tmp, 0, 0, null);
    g2d.dispose();

    return dimg;
}
```

Problema: Cuando le quitamos el marco al JFrame no podemos arrastrarlo por la pantalla.
Solución: Modificar los eventos keyPressed y KeyReleased para modificar la localización del JFrame manualmente.

```
private boolean mousePressed;
private void formMousePressed(java.awt.event.MouseEvent evt) {
      mousePressed = true;
}

private void formMouseReleased(java.awt.event.MouseEvent evt) {
      if (mousePressed)
      {
            this.setLocation( evt.getXOnScreen(),
                              evt.getYOnScreen());
      }
      mousePressed = false;
}
```

Problema, cuando colocamos un componente: setVisible(false), en los eventos mouseEntered() y mouseExited() aplicados sobre él mismo, el componente desaparece y no podemos volver a encontrarlo.
Solución: agregamos un Panel bajo el botón, cuando el mouse entre al panel el botón se coloca visible, cuando el mouse entra a otro componente que no sea el panel, el botón se hace invisible.

Completar la funcionalidad de la Aplicación:

A continuación se detallan los problemas encontrados durante la fase de pruebas:

1. Cuando la canción avanza automáticamente no cambia el resaltado en la lista de canciones. [corregido]
2. Cuando la canción avanza automáticamente no cambia la imagen y el nombre de la canción en el menú principal de la aplicación. [corregido]
3. La imagen en el fondo del menú principal (NewSkin.java) se distorsiona.

Problema: La barra de progreso no avanza correctamente debido a la función que calcula el porcentaje de la canción que se ha reproducido.
Solución:
En lugar de utilizar:

```
double songProgressBar =
(actualSongTime/song.durationInSeconds)*100;
```

Para calcular el porcentaje de la canción reproducida, se modificó por:

```
double songProgressBar =
(actualSongTime*100/song.durationInSeconds);
```

Creando y conectando JDBC con la aplicación de escritorio

Problema: al insertar canciones que contengan comillas en su nombre o el nombre del artista la base de datos empotrada retorna un error. Por ejemplo:

```
Exception Message Error de Sintaxis en sentencia SQL "SELECT * FROM SONG WHERE name = 'Cadillacs (Version '93)' AND artist = 'Los Fabulosos Cadillacs[*]' "
```

O también:

```
INSERT INTO SONG(name,artist,genre,local) VALUES ('Cadillacs (Version '93)','Los Fabulosos Cadillacs','Ska','C:\Users\Maikol\Music\Vasos Vacíos\01-los_fabulosos_cadillacs-cadillacs_(version_93)-its.mp3') [42000-190]
```

Solución: la secuencia de escape para las comillas simples en una sentencia de sql son las comillas simples, es decir, la sentencia debe escribirse de la siguiente manera:

```
SELECT * FROM SONG WHERE name = 'Cadillacs (Version ''93)' AND artist = 'Los Fabulosos Cadillacs'
```

O también:

```
INSERT INTO SONG(name,artist,genre,local) VALUES ('Cadillacs (Version ''93)','Los Fabulosos Cadillacs','Ska','C:\Users\Maikol\Music\Vasos Vacíos\01-los_fabulosos_cadillacs-cadillacs_(version_93)-its.mp3')
```

Solución basada en el post de Víctor Cuervo, tomada de:

http://lineadecodigo.com/sql/insertar-comillas-simples-en-sql/.

H2 EXAMPLE http://www.javatips.net/blog/2014/07/h2-database-example

Problema: El nombre de la tabla en la base de datos no puede ser diferente al del modelo.

```
<br />
<b>Fatal error</b>:  Uncaught exception 'Phalcon\Mvc\Model\Exception' with messa
ge 'Table 'robots' doesn't exist in database when dumping meta-data for Robots'
in C:\xampp\htdocs\cusuco\index.php:81
Stack trace:
#0 [internal function]: Phalcon\Mvc\Model\MetaData\Strategy\Introspection-&gt;ge
tMetaData(Object(Robots), Object(Phalcon\Di\FactoryDefault))
#1 [internal function]: Phalcon\Mvc\Model\MetaData-&gt;_initialize(Object(Robots
), 'robots-robots', 'robots', '')
#2 [internal function]: Phalcon\Mvc\Model\MetaData-&gt;readMetaData(Object(Robot
s))
#3 [internal function]: Phalcon\Mvc\Model\MetaData-&gt;hasAttribute(Object(Robot
s), 'name')
#4 [internal function]: Phalcon\Mvc\Model\Query-&gt;_getQualified(Array)
#5 [internal function]: Phalcon\Mvc\Model\Query-&gt;_getExpression(Array)
#6 [internal function]: Phalcon\Mvc\Model\Query-&gt;_getOrderClause(Array)
#7 [internal function]: Phalcon\Mvc\Model\Query-&gt;_prepareSelect()
#8 [internal function]: Phalcon\Mvc\Model\Query-&gt;parse()
#9 [internal function]: Phalcon\Mvc\Model\Query-&gt;execute()
#10 C:\xampp\htdocs in <b>C:\xampp\htdocs\cusuco\index.php</b> on line <b>81</b>
<br />
```

Solución: Para solucionar este problema se deben implementar las siguientes funciones en la clase del modelo:

```
public function getSource()
{
    return 'TableName';
}

public function initialize()
{
    $this->setSource('TableName');
}
```

Solución encontrada en la respuesta de Nikolaos Dimopoulos en la página: http://stackoverflow.com/questions/14042971/what-is-the-most-correct-way-to-define-source-table-in-phalcon-mvc-model

Comunicación JAVA - API

Para implementar las peticiones GET nos basamos en el ejemplo descrito en la página: http://crunchify.com/java-url-example-getting-text-from-url/
En el mismo obtienen cadenas de JSON desde una URL, la URL a la que le solicitamos los datos es la que corresponda a la API y a los datos que queremos obtener, por ejemplo **/cusuco/api/songs** nos retorna todas las canciones. Para esto utilizamos las siguientes librerías:

Problema: la clase HttpRequest no puede procesar el contenido de la respuesta de la petición HTTP.
Solución: se cambio la línea: HttpResponse result = httpClient.execute(request);
Por la línea: final String response = httpClient.execute(request,new BasicResponseHandler());
En este caso, response tiene el contenido de la respuesta que nos envía el servidor.

Conclusiones y Recomendaciones del proyecto.

Para cuestiones educativas un Raspberry Pi 2 puede ser usado como servidor, pero se recomienda como futuras mejoras pasarlo a un servidor con mejores caracterices de hardware.

PHP es un lenguaje muy poderoso para la creación de aplicaciones web, se integra de manera muy sencilla con HTML para el manejo de paginas web y mediante una API Restful se puede de manera sencilla lograr una comunicación segura.

Phalcon es una poderosa herramienta para crear una aplicación API RESTful

La opción mas simple y poderosa para una base de datos empotrada en Java es H2.

La mejor opcion para tratar los archivos mp3 y sus tags es m3agic.

JSON es mas simple que XML, ademas existe mas documentación para JSON

Siempre se deben utilizar patrones de diseño para simplificar el trabajo y evitar el gasto de tiempo innecesario

Se recomienda el uso de hilos para la carga y descarga de datos, con esto se evita problemas de espera o que se cuelgue la aplicación

Bibliografía consultada en todo el proyecto

Benson, B. (s. f.). *4.3 Strings* Recuperado de http://docs.racket-lang.org/reference/strings.html

Benson, B. (s. f.). *4.5 Characters* Recuperado de http://docs.racket-lang.org/reference/characters.html#%28def._%28%28quote._~23~25kernel%29._char~3f%29%29

Getting a line of user input in Scheme? - Stack Overflow (2010, 04 de Julio). Recuperado de https://stackoverflow.com/questions/3173327/getting-a-line-of-user-input-in-scheme

demas (2011, 07 de Octubre). *Scheme - String split function - Stack Overflow* Recuperado de https://stackoverflow.com/questions/7691769/string-split-function

Show simple open file dialog using JFileChooser (2015, 21 de Julio). Recuperado de http://www.codejava.net/java-se/swing/show-simple-open-file-dialog-using-jfilechooser

Adeeb (2013, 02 de Junio). *How to use the Java MP3 ID3 Tag Library to retrieve album artwork* Recuperado de https://stackoverflow.com/questions/14728621/how-to-use-the-java-mp3-id3-tag-library-to-retrieve-album-artwork

ndsmyter (2012, 23 de Febrero). *Bufferedimage resize* Recuperado de https://stackoverflow.com/questions/9417356/bufferedimage-resize

Crunshify (2014, 16 de Diciembre). *Java URL example: Getting text from URL ? Send HTTP request GET/POST in Java* Recuperado de http://crunchify.com/java-url-example-getting-text-from-url/

Cuervo, V. (2013, 07 de Febrero). *Insertar comillas simples en SQL - Línea de Código* Recuperado de http://lineadecodigo.com/sql/insertar-comillas-simples-en-sql/

Tutorial 7: Creating a Simple REST API (s. f.). Recuperado de https://docs.phalconphp.com/en/latest/reference/tutorial-rest.html

avasin (2012, 26 de Diciembre). *What is the most correct way to define source table in PhalconMvcModel? - Stack Overflow* Recuperado de

https://stackoverflow.com/questions/14042971/what-is-the-most-correct-way-to-define-source-table-in-phalcon-mvc-model

Josh M (2012, 21 de Agosto). *Java - JLayer - Pause and resume song - Stack Overflow* Recuperado el 21 de Noviembre del 2015, de https://stackoverflow.com/questions/12057214/jlayer-pause-and-resume-song

Pérez, B. (2015). Tupera (Versión Alnitak) [Software]. San José, Costa Rica.

admin (2014, 06 de Julio). *H2 Database Example* Recuperado el 21 de Noviembre del 2015, de http://www.javatips.net/blog/2014/07/h2-database-example

Iniciación a PHP 5 | Tutorial PHP | Aprende PHP en 10 capítulos (s. f.). *Tutorial php.net*, 1. Recuperado de http://tutorialphp.net/iniciacion-a-php-5/

Creating a RESTful API with PHP (2013, 17 de Mayo). Recuperado de http://coreymaynard.com/blog/creating-a-restful-api-with-php/