












<b>Grupo: 2SI</b>	
<b>Curso: 2017/2018</b>	<b>Fecha de entrega: 13/12/2017</b>
<b>Nombre:</b>	<b>Tiempo:</b>
<b>Examen 1ª EVALUACIÓN “PROGRAMACIÓN MULTIMEDIA” Y</b>  <b>“ACCESO A DATOS”</b>	
<b>OBJETIVO “SPOTIFY”:</b>  <div style="display: flex; align-items: center; justify-content: center;">  <div> <p><b>¡Pon música que te encanta!</b></p> <p>Escucha tus canciones favoritas, descubre nuevos lanzamientos o pon temas que te traen recuerdos.</p> </div> </div> <p>Desarrollar una aplicación que permita <i>la gestión de los artistas de Spotify</i>.</p> <p>El software debe permitir las <b>siguientes acciones</b>:</p> <ol style="list-style-type: none"> <li>1. Listar Artistas.</li> <li>2. Alta Artista.</li> <li>3. Borrar Artista.</li> <li>4. Modificar Artista.</li> </ol> <p>La arquitectura del proyecto debe cumplir con el patrón MVC y la siguiente estructura de capas:</p> <ul style="list-style-type: none"> <li>• El <b>cliente</b> hará <u>peticiones al Servidor</u> a través de <u>programación concurrente (AsyncTask)</u> e <u>intercambio de información en JSON</u> (versión más nueva de XML)</li> <li>• El <b>Servidor</b> <u>recibirá y responderá</u> las <u>peticiones en JSON</u> y <u>cumplirá</u> con patrones como <u>FrontController</u>, <u>Action's</u> y <u>DAO's</u>.</li> <li>• Los <b>DAO's</b> ejecutarán las <u>peticiones a base de datos a través de SQL</u>.</li> </ul> <p><b>Nombre del Proyecto: “Apellido_Android”</b></p> <p><b>Capa Model.</b> – Preparando el SQL. (Hasta 5,00 ptos en la asignatura de Acceso A Datos):</p> <div style="border: 2px solid red; padding: 10px; margin-top: 10px;"> <p style="text-align: center; background-color: #c00000; color: white; margin: -10px -10px 10px -10px;"><b>Requerimientos</b></p> <ul style="list-style-type: none"> <li>• Prepara un <b>las consultas necesarias en SQL</b> que permita el trabajo con la base de datos.</li> <li>• Columnas: <u>ID_USER</u>, <u>NAME</u>, <u>SURNAME</u>, <u>EMAIL</u>, <u>PASSWORD</u>.             <ul style="list-style-type: none"> <li>○ TABLA “<b>USUARIO</b>”                 <ul style="list-style-type: none"> <li>▪ <u>EXISTS_USER</u>: SQL que permite validar un Usuario.</li> <li>▪ <u>ADD_USER</u>: SQL que permite insertar un usuario con un ID Autonumérico y los datos que correspondan.</li> <li>▪ <u>DELETE_USER</u>: SQL que permite eliminar un usuario por ID.</li> <li>▪ <u>UPDATE_USER</u>: SQL que permite modificar un usuario por ID y los datos que correspondan.</li> <li>▪ <u>FIND_ALL</u>: SQL que devuelva todos los usuarios.</li> </ul> </li> </ul> </li> <li>• <b>Preparar una batería de pruebas unitarias</b> para comprobar que todas las transacciones con la base de datos se ejecutan correctamente. Teniendo en cuenta que:             <ul style="list-style-type: none"> <li>○ <u>Casos especiales</u>: Protección contra usuarios duplicados, emails sin repetir, etc.</li> </ul> </li> </ul> </div>	

**Capa View-Controller.** – Preparar Android y Servidor en Java Web. (Hasta 5,00 pto en la asignatura de Móviles):

### Requerimientos

- Construye una app móvil basada en **Activities** como se muestra en la figura y que cumplan con los requisitos descritos al inicio del examen (a excepción del filtrar):

<b>User:</b> <input type="text"/>  <b>Pass:</b> <input type="password"/>  <input type="button" value="Login"/>	 <b>Añadir Usuario</b>
<ul style="list-style-type: none"> <li>•  <b>Lana del Rey</b>  </li> <li>•  <b>Fergie</b>  </li> <li>•  <b>ColdPlay</b>  </li> </ul>	

### CRITERIOS DE CORRECCIÓN

<u>Ejecución:</u> Cumplir con todos los requisitos que se solicitan	<b>50%</b>
<u>Calidad del Código, Patrones de Desarrollo</u>	<b>50%</b>