Proyecto #1 - CE Music Player

Tecnológico de Costa Rica
Área Académica Ingeniería en Computadores
CE-1103 Algoritmos y Estructuras de Datos I
II Semestre 2022
Valor 25%

Objetivo general

- ➔ Implementar una aplicación que permita la gestión de una biblioteca musical, reproduciendo música en la aplicación y controlado por un arduino.

Objetivos específicos

- ➔ Diseñar una solución que permita resolver el problema descrito en esta especificación aplicando patrones de diseño y utilizando el estándar UML.
- ➔ Implementar una solución que permita resolver el problema descrito en esta especificación utilizando Programación Orientada a Objetos en Java y haciendo uso de estructuras de datos lineales.
- ➔ Elaborar la documentación correspondiente a la solución implementada para la evidencia del trabajo desarrollado utilizando estándares de documentación técnica y herramientas de gestión de proyectos.

Descripción del problema

El proyecto consiste en la implementación de una aplicación que permita gestionar bibliotecas musicales, interactuando con un Arduino, para la reproducción de las canciones de las bibliotecas.

CE Music Player

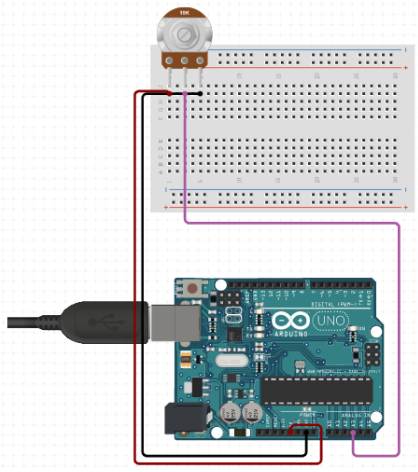
Es una aplicación en Java que permite gestionar una biblioteca de archivos musicales (MP3). Posee las siguientes funciones:

1. **Gestionar usuarios.** La aplicación debe permitir la autenticación de usuarios y así gestionar la información de cada usuario. Para cada usuario es necesario considerar la siguiente información: nombre completo, correo electrónico, provincia (seleccionar de una lista predefinida) y contraseña. *Al iniciar la aplicación se debe cargar la información de todos los usuarios en una lista simple.*
2. **Gestionar bibliotecas musicales.** La aplicación debe permitir la gestión de las bibliotecas de cada usuario. Esta gestión implica agregar, consultar, editar y eliminar las bibliotecas de cada usuario. Los usuarios solo pueden gestionar sus propias bibliotecas. Después de autenticar a cada usuario se debe mostrar en pantalla la lista de bibliotecas del usuario, indicando el nombre de la biblioteca, cantidad de canciones y fecha de creación de la biblioteca (esta fecha debe tomarse de la fecha del computador). *La información de las bibliotecas debe implementarse en una lista simple.*

3. **Gestionar canciones de una biblioteca musical**. Esta gestión implica:
    a. Mostrar un diálogo de selección de carpetas/archivos que cada usuario puede agregar a la biblioteca.
    b. Eliminar canciones de la biblioteca.
    c. Modificar la metadata de cada canción (género, artista, álbum, año, letra). La información de la metadata de cada canción debe ser gestionada a través de archivos XML.
    d. Mostrar canciones favoritas. La aplicación debe permitir a cada usuario seleccionar las canciones favoritas y que se visualice cuáles son canciones favoritas.
4. **Reproducir canciones.** La reproducción de canciones implica la implementación de las funciones de reproducir, pausar, siguiente, anterior, subir y bajar volumen. Cada usuario debe tener la posibilidad de activar la reproducción continua, lo cual implica que al llegar al final de la lista de reproducción continua con la primera canción de la lista de reproducción. Cuando esté activa la reproducción continua debe visualizarse en la aplicación. *Para efectos de lo anterior, la lista de canciones en reproducción debe cargarse en una lista doblemente enlazada circular.*

CE Music Player Controller
Implementado en Arduino. Las funcionalidades principales son:

1. **Reproducir música desde el Arduino:** El usuario puede enviar a reproducir música desde el Arduino. La canción sonará por medio de la computadora.
2. **Controles para la música:** El usuario puede controlar la música que está sonando. Para esto, por medio de botones, la canción se puede reproducir, pausar, ir a la siguiente canción o a la anterior. Adicionalmente, con otro botón se puede activar o desactivar el modo continuo.
3. **Controles de volumen:** El usuario, por medio de un potenciómetro rotatorio, puede controlar el volumen de la canción. Para la conexión del potenciómetro al Arduino, puede basarse en la siguiente figura donde un extremo se conecta a 5V, el otro extremo a GND y el pin del centro a una entrada analógica del Arduino.



4. **Marcar canción favorita:** El usuario, por medio de un botón, puede marcar una canción como favorita.
5. **Identificar canción favorita:** El usuario, por medio de un LED, puede observar si la canción que está sonando es favorita.

6. **Identificar niveles de volumen:** El usuario, por medio de un cinco LEDs, puede observar el porcentaje de volumen. Donde 0% corresponde a todos los led apagados y 100% los cinco led encendidos.
7. **Identificar reproducción continua:** El usuario, por medio de un LED, puede observar si el modo de reproducción es continua.

Requerimientos no funcionales

1. Cabe resaltar que es de manera obligatoria realizar la maqueta del control para interactuar con la aplicación. No se permite presentar el circuito únicamente en la protoboard.
2. Bibliotecas para comunicación serial con Arduino: jSerialComm o RXTX.

Documentación requerida

1. Internamente, el código se debe documentar utilizando Javadoc. Se debe generar el HTML.
2. La documentación externa se hará en un documento que incluya lo siguiente (deberá entregarse un PDF):
    a. Breve descripción del problema.
    b. Diagrama de clases.
    c. Descripción de las estructuras de datos desarrolladas.
    d. Problemas encontrados en forma de bugs de *github*: En esta sección se detalla cualquier problema que no se ha podido solucionar en el trabajo.
3. **Planificación y administración del proyecto:** se utilizará Azure DevOps para la administración del proyecto. Debe incluir:
    a. Lista de features e historias de usuario identificados de la especificación.
    b. Plan de iteraciones que agrupen cada bloque de historias de usuario de forma que se vea un desarrollo incremental
    c. Descomposición de cada user story en tareas.
    d. Asignación de tareas entre los integrantes del grupo.

Aspectos operativos y evaluación:

1. **Fecha de entrega: De acuerdo con el cronograma del curso y lo establecido en el TEC Digital**
2. El proyecto tiene un valor de 25% de la nota del curso.
3. El trabajo es **en parejas.**
4. Deben entregar en el TEC Digital un documento con el link del repositorio de GitHub, Azure DevOps y el PDF de la documentación. Para ambas herramientas deben dar acceso al correo del profesor.
5. Es obligatorio utilizar un Git y GitHUB para el control de versiones del código fuente y evidenciar el uso de Commits frecuentes.
6. Es obligatorio integrar toda la solución.
7. El código tendrá un valor total de 70%, la documentación externa 20% y la defensa un 10%. **El valor correspondiente al rubro de código corresponde a la implementación de la Aplicación y su interacción con el arduino.**
8. De las notas mencionadas en el punto anterior se calculará la Nota Final del Proyecto.
9. Se evaluará que la documentación sea coherente, acorde a la dificultad/tamaño del proyecto y el trabajo realizado. Se recomienda que realicen la documentación conforme se implementa el código.
10. La nota de la documentación externa es proporcional a la completitud del proyecto.

11. Las citas de revisión oficiales serán determinadas por el profesor durante las lecciones o mediante algún medio electrónico.
12. Los estudiantes pueden seguir trabajando en el código hasta 15 minutos antes de la primera cita de revisión oficial.
13. Aún cuando el código y la documentación externa tienen sus notas por separado, se aplican las siguientes restricciones
    a. Si no se entrega documentación externa, automáticamente se obtiene una nota de cero en la nota final del proyecto.
    b. Si no se utiliza un manejador de código se obtiene una nota de cero en la nota final del proyecto.
    c. Si la documentación externa no se entrega en la fecha indicada se obtiene una nota de cero en la nota final del proyecto.
    d. Sí el código no compila se obtendrá una nota de cero en la nota final del proyecto, por lo cual se recomienda realizar la defensa con un código funcional.
    e. El código debe ser desarrollado en Java (Windows), en caso contrario se obtendrá una nota de cero en la nota final del proyecto.
    f. Si alguna persona integrante del proyecto no se presenta a la revisión se asignará una nota de cero en la nota final del proyecto.
14. La revisión de la documentación podrá ser realizada por parte del profesora antes, durante o después de la revisión del proyecto.
15. Cada estudiante tendrá como máximo 30 minutos para exponer su trabajo al profesor y realizar la defensa de éste, es responsabilidad de los estudiantes mostrar todo el trabajo realizado, por lo cual se recomienda tener todo listo antes de ingresar a la defensa.
16. Cada excepción o error que salga durante la ejecución del proyecto y que se considere debió haber sido contemplada durante el desarrollo del proyecto, se castigará con 2 puntos de la nota final del proyecto.
17. Cada estudiante es responsable de llevar los equipos requeridos para la revisión, si no cuentan con estos deberán avisar al menos 2 días antes de la revisión al profesor para coordinar el préstamo de estos.
18. Durante la revisión únicamente podrán participar el estudiante, asistentes, otros profesores y el coordinador del área.