TEC | Tecnológico de Costa Rica

Lenguajes de Programación

Proyecto 1

Imperativo y Funcional

Profesor:

Oscar Víquez Acuña

Estudiantes:

Ian Calvo Madrigal

Camila Ulate Chaves

II Semestre 2023

Para documentar el trabajo se requiere que se haga uso del formato establecido por la Carrera de Computación para este fin. Se solicita especial énfasis en la documentación de aspectos como:

· Manejo de archivos de música

· Administración en memoria de la lista de canciones del lado del servidor

· Detalles sobre manejo de conflictos y excepciones durante la ejecución de los módulos

· Hallazgos en torno a la implementación de sockets y procesos en paralelo. Cómo funciona el programa sin sincronización y cómo sería el escenario implementando cierta sincronización.

· Qué elementos consideran, deben tomarse en cuenta mejorar/cambiar, con miras en una implementación robusta de un sistema similar en producción a gran escala?

Manejo de Archivos de Música:

El manejo de archivos de música se implementa en el servidor y es recibido por el cliente, inicialmente construimos un struct con el nombre de la canción, artista, género y la dirección donde se encuentra el archivo mp3 de esa canción. El proceso para que el cliente obtenga el archivo de la canción del lado del servidor es el siguiente: Inicialmente el cliente solicita la canción que desea reproducir, en el servidor se reciba el nombre y se busca su dirección, con la biblioteca ´os´ se lee el contenido del archivo utilizando la función Readfile que es de la biblioteca mencionada anteriormente, y se envían los datos de canción al cliente.

El cliente recibe los datos enviados desde el servidor. La función Memorystream parte de la biblioteca estándar de .NET se crea porque ahí se almacenarán los

datos de la canción, se coloca un puntero al inicio del memorystream para empezar la lectura de datos. Seguidamente se crea un archivo temporal con una extensión de .mp3 para almacenar temporalmente los datos de la canción antes de reproducirla, utilizando filestream se crea para escribir los datos de la canción en el archivo temporal creado y utilizando nuevamente memorystream se copian los datos. Seguidamente se hace función de la biblioteca NAudio que su propósito es reproducir archivos mp3. Se implementa un while que va a permitir adelantar, retroceder, pausar/reanudar la canción y volver al menú principal. Finalmente se hace una limpieza del archivo mp3 y los archivos temporales.

Administración de la memoria:

Para la implementación de este proyecto las canciones se manejaron de manera local, en el caso del código se almacenan las direcciones mp3 donde se encuentran las canciones. La memoria se utiliza para almacenar temporalmente la canción que el cliente solicita, en otras palabras se lee desde los archivos y se envían al cliente a través de la conexión red. Consideramos que al tener solamente un archivo temporal con la información que el cliente solicita el programa será más eficiente ya que no gastará recursos y almacenamiento en archivos no solicitados.

Manejo de conflictos y excepciones:

En el servidor implementamos una variable que permite verificar si en el momento que se van a enviar o a recibir datos con el cliente si se hizo correctamente, y en caso de que no se haya realizado manda la excepción en vez de que el código se caiga. Además de iniciar la conexión con el cliente en F# se realizan una serie de validaciones si todos los pasos para la conexión se realizaron correctamente. Con

estas excepciones prevenimos los escenarios en lo que si hubiera un error por parte del cliente o del mismo sistema sepamos cuál es el error y al manejarlo de la manera correcta prevenimos que el sistema se vea perjudicado con un cierre imprevisto.

En el cliente implementamos try catch, estos en el primer bloque que sería catch solucionamos la situación de la manera ideal y en el segundo bloque catch manejamos los errores que se pudieron haber presentado e indicando al cliente que hubo un error. Además de los mensajes de error también se indica cuando se logró de esta manera se le asegura al cliente que se haya realizado correctamente lo solicitado.

Hallazgos en torno a la implementación de sockets y procesos en paralelo.

Cómo funciona el programa sin sincronización y cómo sería el escenario implementando cierta sincronización.

Los sockets son protocolos de comunicación entre aplicaciones de red. Para la implementación de este proyecto se utilizó net.Conn y algunos de los métodos fueron: Read, Write y Close. Esto permitió la conexión del servidor con el cliente y la transmisión de los datos.

- Sin sincronización: Cuando se dan los casos de que no hay sincronización se puede ver reflejado en el manejo de conexiones de TCP, esto podría resultar en múltiples conexiones simultáneas al servidor si es cliente las realiza rápidamente.
- Con Sincronización: Se puede implementar un mecanismo que limite la cantidad de conexiones TCP simultáneamente en el servidor, esto se podría implementar con una bandera. También la sincronización se puede

implementar para las canciones y que no permite reproducir múltiples canciones simultáneamente.

Elementos a tomar en cuenta para mejorar/cambiar un sistema para producción a gran escala?

Esta aplicación busca ser lo más completa posible, manejar todos los aspectos que podría utilizar un reproductor de música de producción a escala global, pero es cierto que hay aspectos adyacentes fuera del funcionamiento de la aplicación por sí misma, que también hay que tomar en cuenta y son fundamentales para toda aplicación que quiera ser competitiva en el mercado actual de software.

Algunos de los aspectos que se creen más importantes para hacer de este proyecto “spoticry” una aplicación más robusta para producción de gran escala son:

- Manejo cuentas de usuario: Con el fin de que este proyecto sea una aplicación de uso global y que va a estar ofreciendo un servicio de streaming, se considera que manejo de cuentas de usuario y suscripciones sería importante de implementar. Con el fin de que los usuarios puedan personalizar sus experiencias y optar por un plan que se acomode a sus necesidades, el manejo de cuentas de usuario se considera de mucha importancia si se llega a pensar en lanzar este proyecto a producción de gran escala.
- Seguridad: Ya que el usuario tiene su cuenta y esta contrata servicios que la aplicación ofrece, se necesitan mecanismos de seguridad para proteger los datos de este y datos propios de la aplicación. Los mecanismos de registro y autenticación se consideran óptimos para aumentar la seguridad de la aplicación.

- Bases de Datos: El servicio de streaming que ofrece “spoticry” es reproducción de música en tiempo real, esta música se almacena en un servidor y se reproduce en el dispositivo del cliente. Para almacenar cantidades masivas de música en el servidor se debe tener una correcta gestión de bases de datos.
- Escalabilidad: Si el propósito de la aplicación es que sea de uso mundial, esta debe tener la capacidad de manejar una cantidad de usuarios superlativa todo al mismo tiempo. El servidor deberá recibir y responder solicitudes de los clientes en un tiempo mínimo y en tiempo real, para todo aquel que contrate los servicios de “spoticry”.
- Aspectos Legales: Las canciones que se estarán reproduciendo pertenecen a artistas, disqueras y productoras en específico, estas canciones por ende tienen derechos de autor y es necesario obtener estos derechos, para poder reproducir la música, de manera legal, de x artista en mi aplicación.