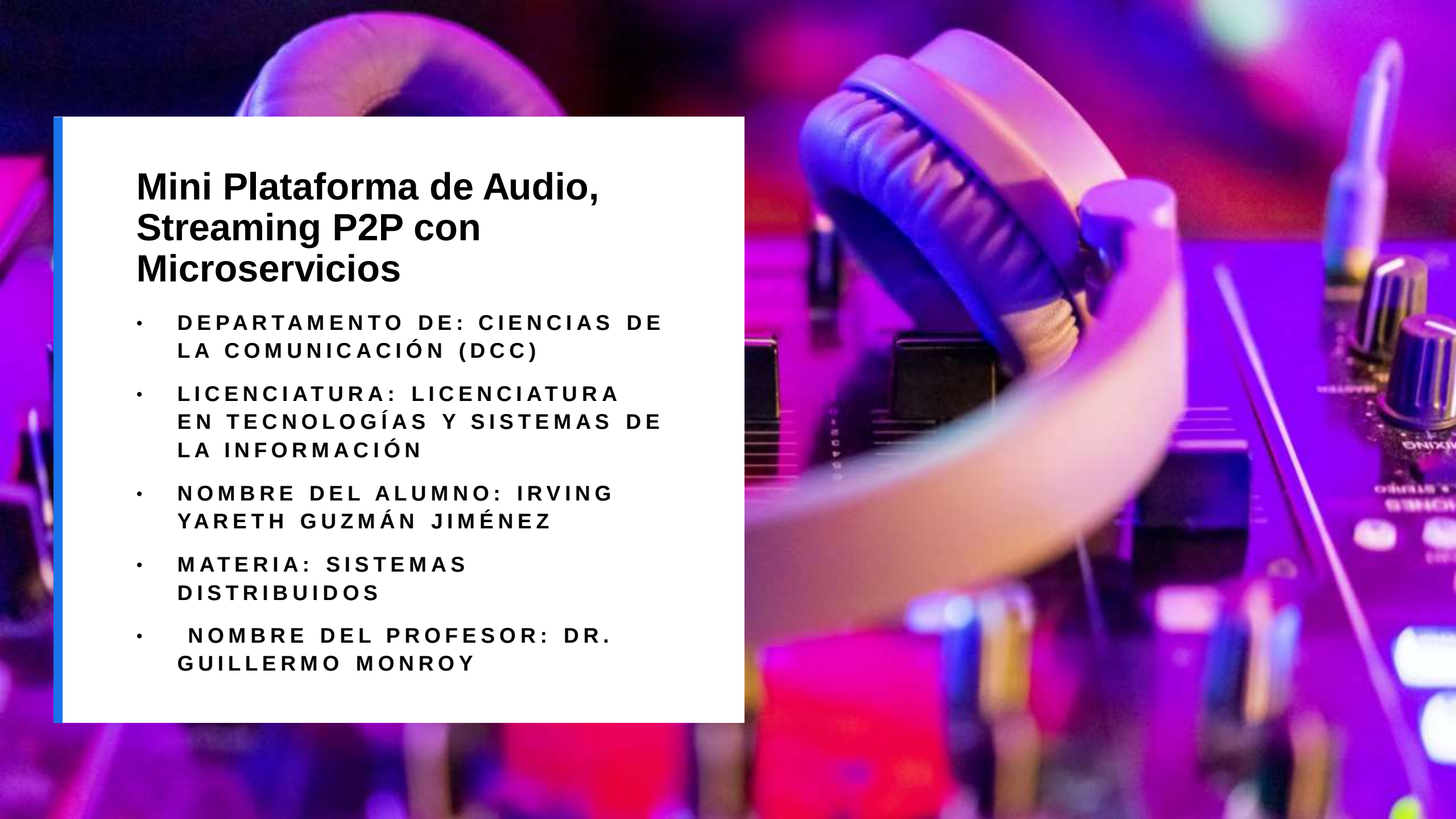


Mini Plataforma de Audio, Streaming P2P con Microservicios

- DEPARTAMENTO DE: CIENCIAS DE LA COMUNICACIÓN (DCC)
- LICENCIATURA: LICENCIATURA EN TECNOLOGÍAS Y SISTEMAS DE LA INFORMACIÓN
- NOMBRE DEL ALUMNO: IRVING YARETH GUZMÁN JIMÉNEZ
- MATERIA: SISTEMAS DISTRIBUIDOS
- NOMBRE DEL PROFESOR: DR. GUILLERMO MONROY

Introducción

- Este proyecto fue realizado en base a los conocimientos vistos y desarrollados durante la clase de sistemas distribuidos.
- Para la realización de este usaremos diferentes tecnologías entre ellas Docker Compose este nos ayudara a contener y facilitar el uso de los entornos ejecutables de la app.
- Usando Python realizaremos diferentes proyectos para dividir un archivo tipo .mp3 en un total de 10 fragmentos, que después podrán ser solicitados por un usuario de manera individual o solicitar el archivo de forma completa.
- Se usara Fast (API REST) para una interfase simple
- Al igual que el sistema Pub/Sub para la publicación de los fragmentos mediante nodos

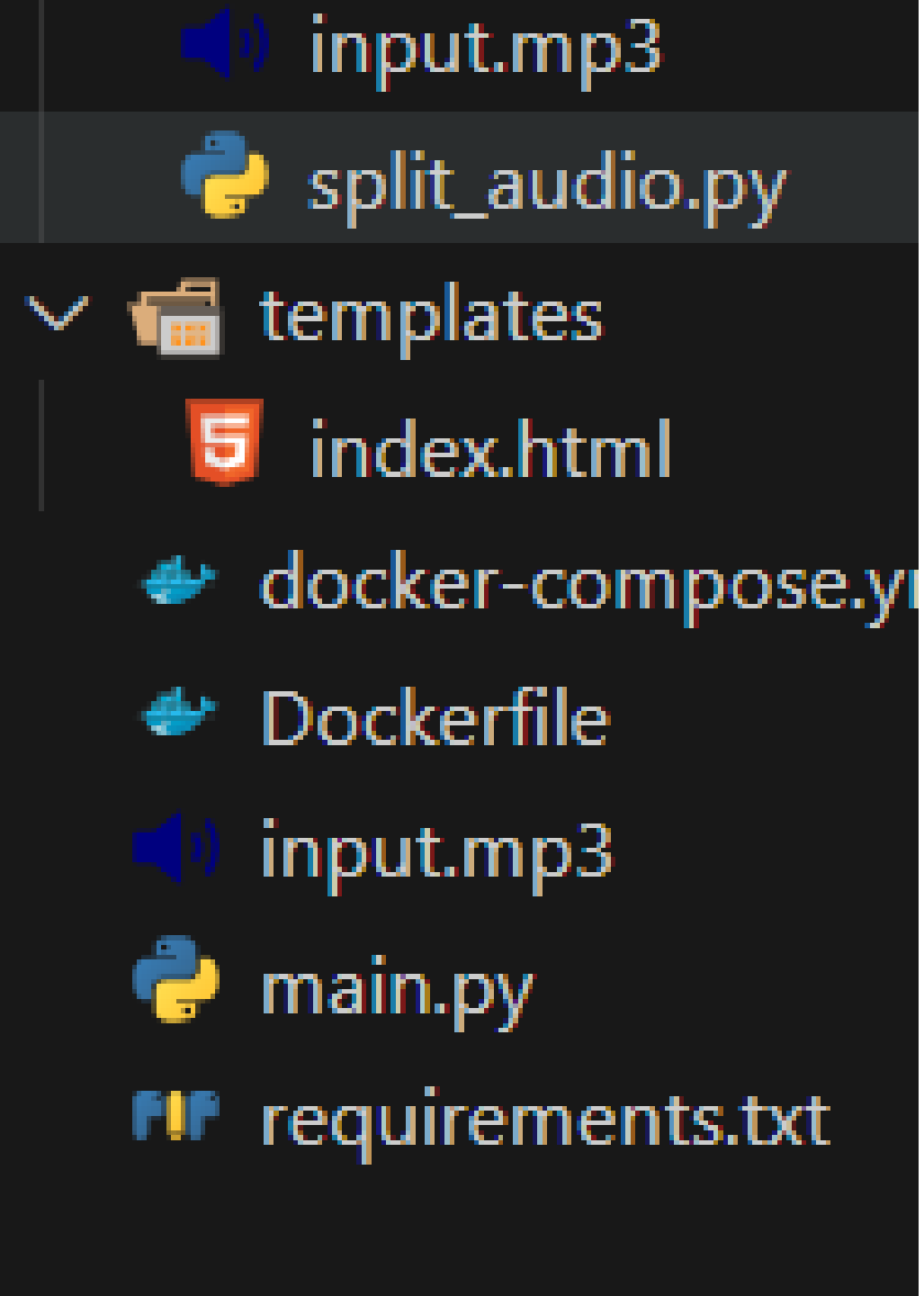
Descripción

Este proyecto implementa un microservicio en **Fast API** que permite:

- Subir archivos de audio (.mp3)
- Dividirlos en 10 fragmentos
- Reensamblarlos en un solo archivo
- Acceder a fragmentos individuales o al audio completo
- Ejecutarse en contenedores Docker
- Integración con sistemas Pub/Sub
- ffmpeg -version

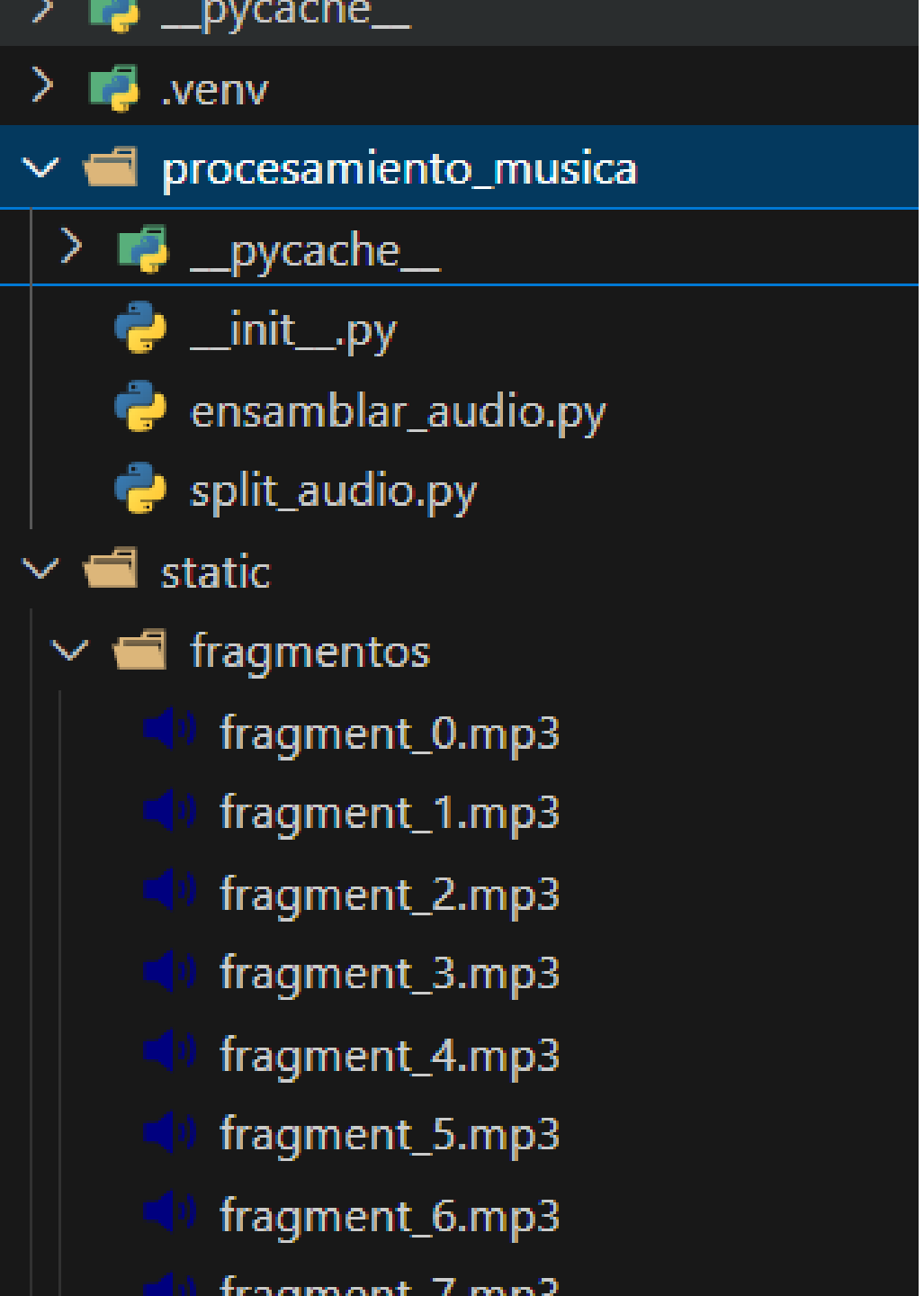
Objetivos

- Desarrollar un microservicio de audio que permita subir, dividir, reensamblar y distribuir archivos MP3 mediante una API REST.
- Implementar una arquitectura con sistemas de mensajería como Pub/Sub, permitiendo la comunicación entre servicios.
- Optimizar el manejo de archivos grandes, permitiendo su fragmentación para hacer mas seguro el envió de estos archivos.
- Hacer una interfaz web intuitiva para que los usuarios puedan subir los archivos .MP3 fácilmente.
- Garantizar la portabilidad y escalabilidad del servicio mediante el uso de contenedores Docker.



Estructura del Proyecto

- En esta imagen se muestra como esta construido el proyecto
- En el main.py se encuentra la API principal del proyecto donde se procesara el audio.
- El apartado de requirements.txt es necesaria para el funcionamiento correcto de la app y así no estar instalando las bibliotecas/paquetes de Python uno por uno.
- El apartado de Docker es la parte donde se crearan nuestros contenedores de la app
- El archivo input.mp3 es el audio de ejemplo que tomaremos para dividirlo



Estructura del Proyecto

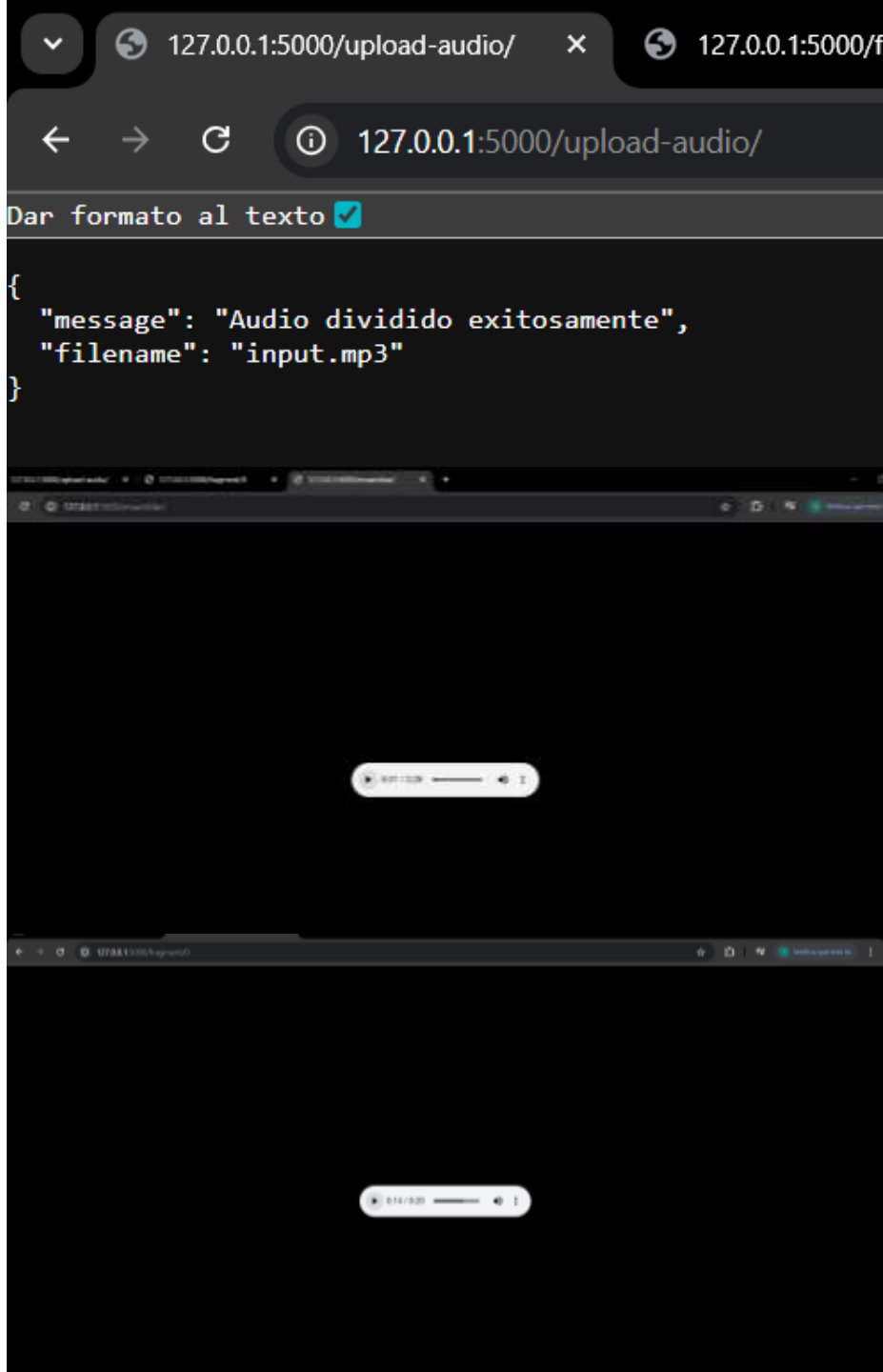
- En el Docker-Compose.yml nos permite especificar varios contenedores que trabajen juntos como una aplicación completa. Por ejemplo, un servicio web
- En la carpeta procesamiento_música se encontraran dos archivos de Python.
- El primer archivo se trata de un archivo que divide el audio en 10 fragmentos y los guarda en una carpeta ya definida y con un nombre definido para que al unirlos estos tengan un orden.
- El segundo archivo se encargara de unir los fragmentos del audio en uno solo por orden ascendente y guardarlo con un nombre especifico y definido en el código.

Instalación y ejecución

1. Abre terminal en la carpeta del proyecto.
2. Ejecuta el comando: docker-compose build
3. Después ejecuta el comando: docker-compose up
4. Abre tu navegador en http://localhost:5000

Instalación y ejecución

- Una vez subido el audio podrás acceder a las diferentes rutas para escuchar cada uno de los fragmentos o acceder a la ruta donde esta el audio completo.
- http://127.0.0.1:5000/fragment/0
- http://127.0.0.1:5000/fragment/1
- ...
- http://127.0.0.1:5000/fragment/9
- http://127.0.0.1:5000/ensamblar/



Conclusión

- Este proyecto demuestra cómo construir un **microservicio eficiente** utilizando tecnologías modernas como Fast(API REST), Docker y Python. A través de una arquitectura clara.
- **Subir, dividir y reensamblar archivos de audio** de forma automatizada.
- El hacer este proyecto nos ayuda a saber como estructurar, desplegar y ejecutar una aplicación mediante contenedores.
- También al usa Fast (API REST) nos ayuda a comprender las dependencias necesarias para la creación de una interfaz simple.
- Repositorio: https://github.com/Irving308/ProyectoSD.git