deti universidade de aveiro
departamento de eletrónica, telecomunicações e informática

Distributed Music Editor

Project elaborated by **David Araújo** *(93444)* and **Filipe Barbosa** *(103064)*

June 2023

Architecture

HTTP Server

- **FastAPI** - Modern, fast (high performance), web framework for building APIs with Python 3.7+ based on standard type hints.

Message Queues

- **RabbitMQ** - Distributed message queue system.

Important Considerations

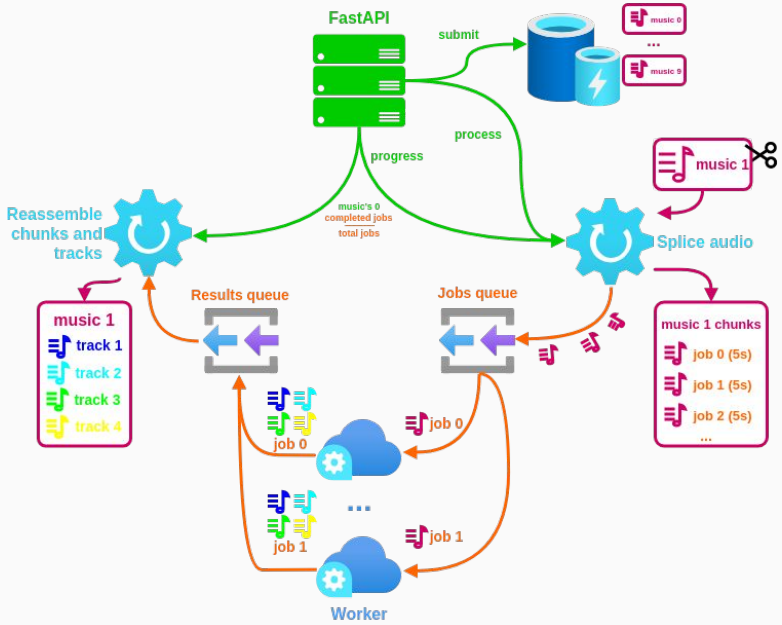
- The **server** system developed operates in **multithreading** in order to process two main operations simultaneously. In one thread, deploying **Fast API and the publishing of audio jobs** to be processed. On the other, the **reception of completed jobs and reassembly of processed audio**.

How does the HTTP Server operates ?

1. **Receive Music from User**
   a. The server receives audio files, and waits for a process request.
2. **Job Creation**
   a. When asked to be processed, each music will be splitted into chunks and these will be published to a public jobs queue.
3. **Progress acknowledge**
   a. If a user request the progress of a a music processing, the server will calculate the percentage of all of the created job of the music that have since been returned by the workers.
4. **Reception of Tracks**
   a. Periodically, the server checks for message in the second queue for jobs that have been completed.
5. **Join the Tracks**
   a. When the server detects it has received all of the completed jobs of a music, the final process begins and five new tracks will be created, one for each instrument and a fifth with the requested ones combined.

How do the Workers operate ?

1. **Check for pending jobs**
   a. Once a worker is active, it will periodically check if the HTTP Server has published chunks to process.
2. **Process chunks**
   a. As soon as worker collects a chunk , it will split it into 4 tracks, one per instrument.
3. **Return processed chunk**
   a. Each chunk will then be returned to the server, alongside with the identification of the job that it was a part of.



Details

- Each music is **identified** by a numeric representation of it's **content checksum**.
- When a music is requested to be processed, it is divided into chunks of **5 seconds** to relieve the worker of heavy processing and allow for **smaller messages**.
- Message queues function over **RabbitMQ**.
- **Containerized workers** were developed in order to better segregate the environment between workers and server.
- Failure tolerance is achieved with **RabbitMQ** which maintains the messages and if a worker were to die, eventually, **another will receive the same message and process it**.