Especificaciones de Don Pepito

# Base de datos

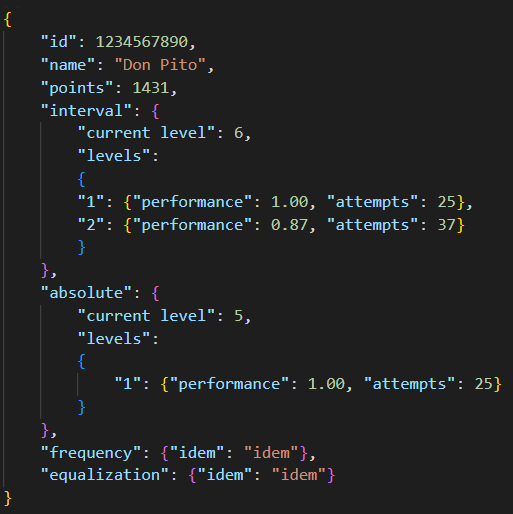
El propósito de la base de datos es almacenar de forma organizada los datos de la performance de cada usuario para proporcionar dicha información de forma que éste conozca sus fortalezas y sus debilidades y esto le facilite la toma de decisiones al respecto de sus entrenamientos.

Por ejemplo, si un usuario tiene una tasa de aciertos suficiente pero menor de la esperada en el nivel 7 de la aventura interválica, se le puede sugerir que debe reforzar dicho nivel antes de seguir con la aventura.

También se puede implementar un comando /entrenar que te sugiera los 3 ejercicios que más te cuestan hasta el momento. En este sentido, la base de datos facilita mucho la extracción organizada de dichos datos.

Tipo de base de datos: NoSQL.

Ejemplo de la estructura de datos de un usuario:



Claves de la estructura de datos:

* Id: Identificador único de cada usuario de Discord.
* Name: Nombre de usuario de Discord (prescindible, teniendo en cuenta que se le puede pedir a Discord dicho nombre aportando su ID).
* Points: puntos ganados por el usuario en los entrenamientos (y duelos). Se inicializa en 0. Quizá sería interesante separar estos puntos en “puntos interválicos”, “puntos absolutos, etc., con el fin de crear ladderboards de cada rama de aprendizaje.
* Interval: Clúster de datos de los entrenamientos de reconocimiento interválico.
* Absolute: Clúster de datos de los entrenamientos de reconocimiento de alturas absolutas.
* Frequency: Clúster de datos de los entrenamientos de reconocimiento frecuencial.
* Equalization: Clúster de datos de los entrenamientos de reconocimiento de ecualizaciones.
* Current level: Es el nivel que actualmente debe superar el usuario al continuar la aventura. Se inicializa en 1 en todas las aventuras.
* Levels: Es un clúster de datos que contiene los datos de rendimiento y número de intentos para cada nivel de la aventura. Se van generando sus elementos a medida que el usuario avanza en cada aventura. Cada elemento tiene como clave el número del nivel, y como valor, otro pequeño clúster que contiene los valores de “performance” y “attempts” del nivel.
* Performance: Ratio de aciertos del último entrenamiento completado de un nivel en concreto expresado en tanto por uno. Se inicializa en 0.
* Attempts: Número de intentos totales realizados de un nivel en concreto. Se inicializa en 0.

# Comandos

## 2.1 Obligatorios:

2.1.1 Generales:

* Comando de ayuda con un argumento opcional que sea el nombre de otros comandos
  + /help

Devuelve una lista de comandos con un brief.

* + /help arg1

Devuelve una descripción del comando escrito en el argumento o un mensaje de error si no existe un comando con ese nombre.

* Ladderboard (top 3, top 5, top 10). Usando la base de datos, haciendo un query ordenando descendentemente los últimos 3, 5, 10.
* Perfil. Hace un query a la base de datos con la clave primaria (ID) y construye un mensaje de tipo embed con la información importante

2.1.2 Comandos de entrenamiento:

* Aventura interválica: Entrenamiento progresivo del reconocimiento de intervalos.

Función de procesamiento de audio: unión de 2 archivos de audio horizontal o verticalmente con los crossfade pertinentes.

* Aventura absoluta: Entrenamiento progresivo de oído absoluto.

Constante que almacene varias nomenclaturas

Función que procese la respuesta (por teclado) del usuario.

* Aventura frecuencial: Entrenamiento progresivo de localización de frecuencias.

Función de procesamiento de audio2: generación de un archivo de audio que contenga una onda de un tipo concreto en una frecuencia concreta y durante un tiempo concreto. (tipos de onda: senoidal, cuadrada, triangular, diente de sierra, etc)

* Aventura ecualizada: Entrenamiento progresivo de la localización de ecualización

2.1.2.1 Funciones en común.

Envío de archivo de audio.

Creación de botones y agrupamiento.

Callbacks de los botones. (dentro de esto preferentemente solo llamadas a otras funciones)

Modificación de los datos del usuario en la base de datos

Procesamiento de la información de usuario (si no existe, se registra en la base de datos, si existe, se modifican. En cualquier caso, se crea un objeto “user” con la info resultante)

Función que pregunta al usuario si quiere reiniciar (Función para resetear un usuario en la db y empezar de cero) o continuar por donde lo había dejado.

Función corregir que compare la respuesta del usuario con la respuesta correcta.

2.1.2.2 Funciones especificas

Función que maneje archivos de audio existentes

## 2.2 Opcionales:

* Modo duelo con otro usuario que tenga que aceptar.
* Selección de entrenamientos específicos
* Comando de sugerencias. Se guardarían en un txt para el desarrollador. La respuesta sería responder con un ¡Gracias! Enviaré tu sugerencia a los desarrolladores.
* Comando “entrenar” que elija el entrenamiento que necesita el usuario para seguir mejorando. (¿de todas las ramas o de una concreta?

# Librerías

* Discord API: Pycord <https://github.com/Pycord-Development/pycord>
* Base de datos: TinyDB <https://github.com/msiemens/tinydb>
* Procesamiento de audio: Pydub <https://github.com/jiaaro/pydub>
* Time zones: pytz https://github.com/stub42/pytz