**1.4 - REMOVE SONG**

A função *removeSong* é responsável por deletar músicas da *dataframe* *tableMusic* ou de uma *playlist*. Possui como parâmetros de entrada duas *dataframes*: *tableMusic* e *playlist*; e o *input* do usuário: *songId*.

É composta por duas funções específicas: *removeSongDataBase* e *removeSongPlaylist*; uma função auxiliar: *getPlaylist* e uma função com mensagens de erro: *errorCodes*.

A screen shot of a computer program

Description automatically generated

A função *removeSongDataBase* remove a música da base de dados de músicas e da base de dados de *playlists* ao mesmo tempo. Possui como parâmetros de entrada: *tableMusic*, *playlists* e o *id* (*input* do usuário, do tipo inteiro). Para cumprir a sua finalidade foi criada uma estrutura do tipo *IF-ELSE* que questiona se o *id* é o mesmo *id* presente na coluna *id\_music* da base de dados *tableMusic* e *playlist*. Caso a resposta seja verdadeira, a música será removida de todas as bases de dados. Caso contrário, imprimirá uma mensagem de erro a informar ao usuário de que aquela música não existe no sistema. Na função é permitido inserir o número 0, caso o usuário não deseje remover nenhuma música.

A screen shot of a computer code

Description automatically generated

Já a função *removeSongPlaylist* remove a música da apenas da *playlist*. Possui como parâmetros de entrada: a *dataframe* *playlist* (*playlists*), o nome da *playlist* onde se deseja remover a música (*playlistName*) e o *id* do usuário (*id*). Funciona de forma semelhante a anterior. Há uma estrutura *IF-ELSE* que verifica se o *input* do usuário referente a *playlist* e ao *id* são correspondentes àqueles existentes no *dataframe* *playlists*, e caso a resposta para ambas as afirmações seja positiva a função executa a remoção da música da *playlist* requer*id*a. Do contrário, uma mensagem de erro é emitida informando que a *playlist* ou o *id* não existem na base de dados. Adicionalmente foi acrescentado a estrutura *try/except* que verifica se o arquivo *playlist.csv* não encontra-se em aberto, visto que se estiver, a função apresentará um erro na sua execução e não irá remover a música.

A screen shot of a computer program

Description automatically generated

**1.6 - BEST RANK STYLE**

A função *bestRankStyle* é responsável por mostrar ao usuário as melhores músicas de cada estilo (ou seja, aquelas que possuem avaliação de 4 a 5) em ordem decrescente. Possui como parâmetros de entrada: a *dataframe* *tableMusic,* e o *input* do usuário: *styleInput*.

É composta por duas funções específicas: *getSongOfStyle* e *rankingStyle*.

A screenshot of a computer program

Description automatically generated

A função *getSongOfStyle* verifica se existe um estilo musical na *dataframe* *tableMusic*. Possui como parâmetros de entrada: o *dataframe* tableMusic e o input do usuário, *style* (estilo musical). Funciona através da comparação dos estilos musicais presentes na *dataframe* *tableMusic* e o estilo musical inserido pelo usuário.

A screen shot of a computer code

Description automatically generated

Caso seja encontrado o estilo musical, a função *rankingStyle* realiza o *ranking* do estilo musical selecionado. Esta função recebe como parâmetros de entrada as músicas do estilo musical *(songs)*. Por fim, a função retorna o valor das melhores músicas em ordem decrescente, desde que a avaliação global da música seja maior do que 3, e apresenta essa lista de músicas ao usuário.

A black background with white text

Description automatically generated

**4.7 - USER RANK PLAYLIST**

A função *userRankPlaylist* é responsável por dar a possibilidade ao usuário de avaliar uma *playlist*. Possui como parâmetros de entrada a *playlist* (*playlists*).

É composta por uma função específica: *addRank*; uma função auxiliar: *getPlaylist* e *addToAverage*; por uma função com mensagens de erro: *errorCodes;* e por um módulo matemático: *math.isnan*.

A função *addRank* é responsável por adicionar a avaliação do usuário a *playlist*. Possui como parâmetro de entrada a *dataframe* *playlists.* Para cumprir a sua função, o usuário deve inserir a *playlist* que deseja avaliar. Através da função auxiliar *getPlaylist*, verificar-se-á se o nome da *playlist* inserida como *input* é o mesmo presente na *dataframe* *playlists*. Caso seja falso, retornará uma mensagem de erro a informar ao usuário de que a *playlist* não se encontra no sistema (ec. *playlist\_not\_found*). Caso contrário, os índices da *playlist* escolhida serão resetados e começarão a partir de zero (caso especialmente válido para as *playlists* que se encontram após a primeira). Com os índices a zero poder-se-á calcular a média dos valores já atribuidos anteriormente a *playlist* e da nova avaliação inserida pelo usuário, como poderá ser visto mais a frente. Após zerar os indíces, aparecerá um output informando qual a *playlist* escolhida e um input do usuário (*userRating*) para escolher um número de 1 a 5 para avaliar a *playlist*.

A computer screen with text

Description automatically generated

A condição *try/except* foi utilizada aqui com a finalidade de verificar se o *input* inserido pelo usuário se trata de um número do tipo *float*. Caso seja falso retornará uma mensagem de erro a informar que o valor inserido é inválido *(ec. sintax).* Caso seja verdadeiro, entrará na condição IF, onde será verificado se o input inserido encontra-se entre 1 e 5. Se a resposta a condição do IF for falsa, retornará uma mensagem de erro *(ec. invalid\_rating).*

A screen shot of a computer program

Description automatically generated

Caso seja verdadeiro, foram criadas variáveis: *playlistRatingColumn*, que pega a coluna de avaliação da *playlist* escolhida; e *currentRank*, que pega a avaliação da *playlist* correspondente a um elemento da *playlist*, correspondente ao índice zero (por isso o valor do índice lá em cima foi zerado pela função do pandas *‘.reset’*).

A screen shot of a computer program

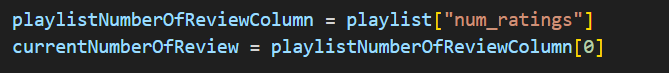
Description automatically generated

Após o *currentRank* foi posto uma condição para verificar se existe uma avaliação atual. Caso a célula encontre-se vazia, ou seja, se ainda ninguém tiver avaliado a *playlist*, o valor de *currentRank* que é igual a *NaN* (*not a number*) assume um valor de 0, para que os cálculos possam ser realizados.

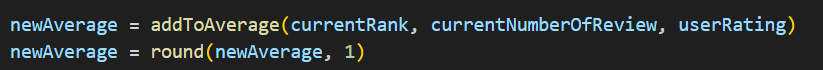
A black background with white text

Description automatically generated

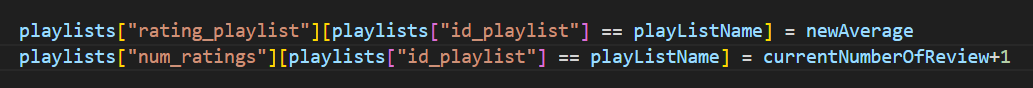
Foram criadas duas variáveis importantes para realizar o cálculo da médica da *playlist*: *playlistNumberOfReviewColumn*, que contém a coluna do número de avaliações realizadas pelo usuário; e *currentNumberOfReview*, que contém o primeiro elemento das avaliações realizadas àquela *playlist* pelo usuário.



A variável *newAverage* recebe a função *addToAverage*, que possui como parâmetros de entrada: *currentRank*, *currentNumberOfReview* e o *userRating*. Uma função do pandas denominada *“round”* foi em seguida utilizada com vistas a calcular o valor da média com uma casa decimal, para os casos em que a divisão resultante seja um número decimal muito grande.



Logo após realizados os cálculos, o valor antigo da avaliação é substituido pelo valor atual, assim como o valor antigo do número de avaliações realizadas, como pode ser visto na variável *playlists*.

******

A função após executar todos os passos grava no arquivo *playlists.csv* as alterações realizadas. Caso o ficheiro esteja aberto, entretanto, retornará uma mensagem de erro informando ao usuário *(ec. file\_open).*

A black screen with orange and white text

Description automatically generated

Por fim, a nova lista com avaliações atualizadas é mostrada ao utilizador, e é mostrada uma mensagem de êxito na operação *(ec. successfull\_execution).*

A black screen with white text

Description automatically generated

**3.4 – PLAYLIST ADD MUSIC**

A função *playlistAddMusic* é responsável por adicionar uma música a *dataframe* *playlists*. Possui como parâmetros de entrada duas *dataframes*: *playlists* e *songDataBase*.

É composta por uma função específica: *addMusic*; duas funções auxiliares: *getPlaylist* e *addToAverage*; e uma função com mensagens de erro: *errorCodes*.

A função *addMusic*, como o próprio nome indica, adiciona uma música a *playlist*. Possui como parâmetros de entrada as *dataframes*: *playlists* e *songDataBase*. Para cumprir a sua finalidade, o usuário deve inserir o nome da *playlist* que deseja adicionar a música. O sistema recolhe as informações acerca dos nomes de *playlists* presentes na *dataframe* *playlists* e compara com o input do usuário. Caso seja falso, significa que o tamanho da *playlist* não corresponde a da dataframe *playlists*, sendo a *playlist* vazia, ou há alguma impossibilidade da *playlist* corresponder a *playlist* da dataframe. Para isso foi utilizada a condição *IF* e uma mensagem de erro é retornada a informar ao usuário de que a *playlist* não foi encontrada.

A screen shot of a computer program

Description automatically generated

Caso seja verdadeiro, os índices da *playlist* são resetados e o usuário deve então inserir o índice referente a música que deseja adicionar.

A black background with white text

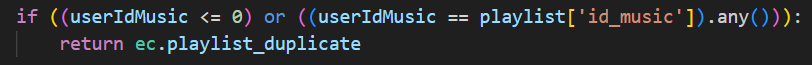
Description automatically generated

A estrutura *try/except* foi utilizada para verificar se o *input* do usuário referente ao *id* corresponde a um número inteiro. Caso seja falso, retornará um erro de sintaxe.

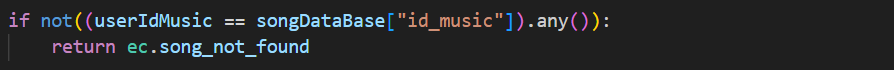
A computer screen with white text

Description automatically generated

Se, entretanto, a condição for verdadeira, será veridicado se o *id* inserido pelo usuário possui um número válido (que não seja negativo ou menor do que 0) ou se a música já não se encontra na *playlist*. Caso seja falso retornará uma mensagem de erro *(ec.playlist\_duplicate).*



Caso seja verdadeiro, o sistema irá verificar se a música está presente na dataframe *tableMusic*. Se, apesar de ser um número válido, não houver nenhuma música com aquele índice em específico outra mensagem de erro será emitida *(ec.song\_not\_found).*

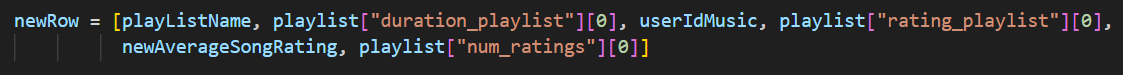


Se a música existir na dataframe *tableMusic* e todas as outras condições anteriores se verificarem como verdadeira, as variáveis *songRating*, *newAverageSongRating* e *playlists* foram deduzidas para atualizar a média da avaliação global da *playlist* com base na nova música que será inserida.

A screen shot of a computer code

Description automatically generated

Uma linha com as novas informações da avaliação da *playlist* e da música a ser adicionada é então criada e alocada na variável *newRow*.



Mediante a função do pandas *.loc* a nova linha é adicionada a *dataframe* *playlists*.

A black background with blue and white text

Description automatically generated

O sistema então tenta salvar as novas alterações, caso não consiga, é porque o ficheiro *.csv* encontra-se aberto. Se for este o caso, emite uma mensagem de erro a informar ao usuário *(ec. file\_open)*. Do contrário, emite uma mensagem de erro igual a zero, a informar que a execussão foi bem sucedida *(ec. successfull\_execution).*

**- Funções Auxiliares**

Possui duas funções auxiliares, entretanto pode vir a comportar outras funções genéricas que podem ser chamadas nas funções principais.

A função *getPlaylist* procura uma playlist numa *dataframe* de *playlists* e a função *addToAverage* retorna uma fórmula de média com base na necessidade desenvolvida na aplicação *Jukebotify*.

**- Função Error Codes**

Ficheiro de variáveis correspondentes a erros que possam surgir durante a execução do sistema. A variável ‘messages’ recebe os índices de outras variáveis e retorna a correspondente mensagem de erro.