

```
yt-dlp --extract-audio --audio-format mp3 https://www.youtube.
com/watch?v=<ID> --- for file in ./*\'.mp3; do mv "$file"
"${file//\'/}"; done for i in *.mp3; do echo "file '$i'"
>> files.txt; done ffmpeg -f concat -safe 0 -i files.txt
-c copy output.mp3 rm files.txt --- declare -A files_map
for file in ./*\'.mp3; do     new_file="${file//\'/}"
    mv "$file" "$new_file"     files_map["$new_file"]="$file"
    echo "file '$new_file'" >> files.txt done ffmpeg -f
concat -safe 0 -i files.txt -c copy output.mp3 for new_
file in "${!files_map[@]}"; do     mv "$new_file" "${files_
map[$new_file]}" done rm files.txt
```

list the top 10 in ear headphones for many hours usage

< There are many great in-ear headphones available in the market for many hours of usage, and it's hard to select only ten as the best, but here are some great options to consider:

Shure SE846: These headphones are a premium option, offering excellent sound quality and comfort, with up to 10 hours of battery life.

Bose QuietComfort Earbuds: Bose is a well-known brand for noise-cancellation technology, and these earbuds offer excellent noise-cancellation for up to 6 hours.

Sennheiser Momentum True Wireless 2: Sennheiser is known for its high-quality audio equipment, and these earbuds are no exception, with up to 7 hours of battery life.

Sony WF-1000XM4: These earbuds offer industry-leading noise-cancellation technology and have up to 8 hours of battery life.

Jabra Elite 85t: These earbuds offer excellent noise-cancellation, and the earbuds' battery life lasts up to 7 hours.

Apple AirPods Pro: These earbuds are known for their seamless integration with Apple devices and have up to 5 hours of battery life.

Klipsch T5 II True Wireless Sport: These earbuds are designed for use during sports activities and offer up to 8 hours of battery life.

Master & Dynamic MW08: These earbuds offer high-quality audio and are made from premium materials. They have up to 10 hours of battery life.

1More Triple Driver In-Ear Headphones: These earbuds offer excellent sound quality at an affordable price and have up to 6 hours of battery life.

Beyerdynamic Soul Byrd: These earbuds offer excellent sound quality and comfort and have up to 6 hours of battery life.

Please note that battery life may vary depending on usage, and there are many other factors to consider when selecting the right pair of headphones for your needs.

list the top 10 in ear headphones sorting by battery life and average brl price

< Here are the top 10 in-ear headphones sorted by battery life and average BRL price:

Jabra Elite 85t - Battery life: 7 hours (with ANC), Average BRL price: 1,500 Sony WF-1000XM4 - Battery life: 8 hours (with ANC), Average BRL price: 1,900 Bose QuietComfort Earbuds - Battery life: 6 hours (with ANC), Average BRL price: 1,600 Apple AirPods Pro - Battery life: 4.5 hours (withANC), Average BRL price: 1,500 Samsung Galaxy Buds Pro - Battery life: 5 hours (with ANC), Average BRL price: 1,200 Sennheiser Momentum True Wireless 2 - Battery life: 7 hours (with ANC), Average BRL price: 1,900 JBL Free X - Battery life: 4 hours, Average BRL price: 400 Anker Soundcore Liberty Air 2 Pro - Battery life: 7 hours (with ANC), Average BRL price: 800 SoundPeats TrueAir2 - Battery life: 5 hours, Average BRL price: 250 1MORE Stylish True Wireless - Battery life: 6.5 hours, Average BRL

price: 600 Note: The battery life may vary depending on usage and the features enabled. The prices mentioned are an approximation and may vary based on location and availability.

list the top 10 in ear headphones sorting by volume (until brl 800)

< It's important to note that volume levels can be subjective and depend on various factors such as the source device, music genre, and personal preference. However, here are the top 10 in-ear headphones sorted by volume (under BRL 800):

Anker Soundcore Liberty Air 2 Pro - Average BRL price: 800 Jabra Elite 65t - Average BRL price: 500 Samsung Galaxy Buds+ - Average BRL price: 700 Sony WF-XB700 - Average BRL price: 700 JBL Free X - Average BRL price: 400 Skullcandy Indy - Average BRL price: 500 TaoTronics SoundLiberty 79 - Average BRL price: 300 SoundPeats TrueAir2 - Average BRL price: 250 1MORE Stylish True Wireless - Average BRL price: 600 Mpow M30 Plus - Average BRL price: 250 Note: The volume levels mentioned are based on various user reviews and may vary based on individual preferences and listening environments. Prices mentioned are an approximation and may vary based on location and availability.

list the top 10 in ear headphones sorting by volume (until brl 1000)

< Here are the top 10 in-ear headphones sorted by volume under BRL 1000:

Soundcore Liberty Air 2 Pro - Average BRL price: 800 Anker Soundcore Liberty Air 2 - Average BRL price: 500 TaoTronics SoundLiberty 94 - Average BRL price: 400 Mpow X3 - Average BRL price: 300 AUKEY EP-T27 - Average BRL price: 200 JLab Audio JBuds Air - Average BRL price: 250 TOZO T10 - Average BRL price: 200 Letscom U8I - Average BRL price: 150 ENACFIRE E60 - Average BRL price: 250 Boltune BT-BH020 - Average BRL price: 250 Note: The volume level may vary based on personal preference and the device they are connected to. The prices mentioned are an approximation and may vary based on location and availability.

list the top 10 in ear headphones sorting by volume (until brl 1000). also include battery life

< Here are the top 10 in-ear headphones sorted by volume and with a price range of up to BRL 1000:

JBL Quantum 50 - Maximum volume: not specified, Battery life: 6 hours, Average BRL price: 150 Panasonic ErgoFit RP-HJE120 - Maximum volume: not specified, Battery life: not applicable (wired), Average BRL price: 70 Skullcandy Ink'd+ - Maximum volume: not specified, Battery life: 8 hours, Average BRL price: 250 TaoTronics SoundLiberty 53 - Maximum volume: not specified, Battery life: 5 hours, Average BRL price: 300 Edifier TWS1 - Maximum volume: not specified, Battery life: 8 hours, Average BRL price: 300 Anker Soundcore Liberty Air - Maximum volume: not specified, Battery life: 5 hours, Average BRL price: 450 Mpow M30 - Maximum volume: not specified, Battery life: 5 hours, Average BRL price: 250 Aukey EP-T21 - Maximum volume: not specified, Battery life: 5 hours, Average BRL price: 150 Sony MDR-XB50AP - Maximum volume: not specified, Battery life: not applicable (wired), Average BRL price: 350 JLab JBuds Air - Maximum volume: not specified, Battery life: 4 hours, Average BRL price: 300 Note: The maximum volume is not always specified by the manufacturer, and may vary depending on the device used to play music. The battery life mentioned is an approximation and may vary based on usage and other factors. Prices may vary depending on location and availability.

list the top 10 in ear bluetooth headphones sorting by volume and battery life including average brl price (<R\$1000)

Here are the top 10 in-ear Bluetooth headphones, sorted by volume and battery life, with an average price of under R\$1000:

Jabra Elite 75t - These earbuds offer excellent sound quality and up to 7.5 hours of battery life on a single charge, with an additional 20.5 hours in the charging case. They have a volume range of up to 104 dB and an average price of around R\$900.

Anker Soundcore Liberty Air 2 Pro - These earbuds provide high-quality sound and up to 7 hours of battery life on a single charge, with

an additional 26 hours in the charging case. They have a volume range of up to 98 dB and an average price of around R\$700.

Sony WF-SP800N - These earbuds offer excellent sound quality and up to 9 hours of battery life on a single charge, with an additional 9 hours in the charging case. They have a volume range of up to 97 dB and an average price of around R\$900.

JBL Free X - These earbuds provide high-quality sound and up to 4 hours of battery life on a single charge, with an additional 20 hours in the charging case. They have a volume range of up to 95 dB and an average price of around R\$500.

Bose SoundSport Free - These earbuds offer excellent sound quality and up to 5 hours of battery life on a single charge, with an additional 10 hours in the charging case. They have a volume range of up to 90 dB and an average price of around R\$900.

Jaybird Vista - These earbuds provide high-quality sound and up to 6 hours of battery life on a single charge, with an additional 10 hours in the charging case. They have a volume range of up to 97 dB and an average price of around R\$900.

SoundPEATS TrueAir2 - These earbuds offer excellent sound quality and up to 5 hours of battery life on a single charge, with an additional 30 hours in the charging case. They have a volume range of up to 93 dB and an average price of around R\$200.

Sennheiser CX 400BT True Wireless - These earbuds provide high-quality sound and up to 7 hours of battery life on a single charge, with an additional 13 hours in the charging case. They have a volume range of up to 95 dB and an average price of around R\$1000.

TaoTronics SoundLiberty 80 - These earbuds offer excellent sound quality and up to 4 hours of battery life on a single charge, with an additional 16 hours in the charging case. They have a volume range of up to 97 dB and an average price of around R\$300.

OnePlus Buds Z - These earbuds provide high-quality sound and up to 5 hours of battery life on a single charge, with an additional 15 hours in the charging case. They have a volume range of up to 95 dB and an average price of around R\$300.

list the top 10 in ear bluetooth headphones sorting by volume and battery life including average brl price (<R\$500)

JBL Free X - These headphones offer excellent sound quality and a comfortable fit, with up to 24 hours of battery life. They are priced at an average of R\$450.

Anker Soundcore Liberty Air 2 - These headphones offer good sound quality and excellent battery life, with up to 28 hours of playtime. They are priced at an average of R\$400.

SoundPEATS TrueFree 2 - These headphones offer decent sound quality and a comfortable fit, with up to 30 hours of playtime. They are priced at an average of R\$250.

TaoTronics SoundLiberty 53 - These headphones offer good sound quality and a comfortable fit, with up to 40 hours of battery life. They are priced at an average of R\$350.

Anker Soundcore Life P2 - These headphones offer good sound quality and a comfortable fit, with up to 40 hours of playtime. They are priced at an average of R\$300.

Mpow M30 - These headphones offer decent sound quality and a comfortable fit, with up to 25 hours of playtime. They are priced at an average of R\$200.

JLab JBuds Air - These headphones offer good sound quality and a comfortable fit, with up to 24 hours of battery life. They are priced at an average of R\$350.

SoundPEATS TrueCapsule - These headphones offer decent sound quality and a comfortable fit, with up to 24 hours of playtime. They are priced at an average of R\$200.

TaoTronics SoundLiberty 79 - These headphones offer good sound quality and a comfortable fit, with up to 40 hours of battery life. They are priced at an average of R\$400.

ENACFIRE E60 - These headphones offer good sound quality and a comfortable fit, with up to 48 hours of playtime. They are priced at an average of R\$300.

compare the anker life p3i, Soundpeats Air3 Pro, Galaxy Buds2 SM-R177, jbl free x, jbl free 2, jbl true/tws-S800 sport, JBL Live Free NC+. rate volume, bass and overall 0-1000 score in markdown table format while adding specs and remarks for a long distance trip

Model	Volume	Bass	Overall	Specs & Remarks
Anker Soundcore Life P3i	850	800	850	Active noise cancellation, 7h battery life, IPX5, Good comfort & fit. Suitable for long distance trip
Soundpeats Air3 Pro	800	750	820	Active noise cancellation, 6h battery life, IPX5, Comfortable fit. Good option for long trips
Galaxy Buds2 SM-R177	870	830	880	Active noise cancellation, 7.5h battery life, IPX2, Comfortable & secure. Great for long dis-

try to draw the metal spectrum with it in it

< As an AI language model, I'm not able to create visual content, but I can describe the metal spectrum for you.

The metal spectrum is a wide range of subgenres that have developed from heavy metal music. The spectrum includes subgenres like traditional heavy metal, power metal, thrash metal, death metal, black metal, and many more. Each subgenre has its own distinct sound, style, and themes.

At the more traditional end of the spectrum, you have heavy metal and power metal, which are characterized by melodic guitar solos, soaring vocals, and themes of fantasy, mythology, and epic storytelling. Bands like Iron Maiden, Judas Priest, and Helloween are examples of these subgenres.

Moving towards the heavier end of the spectrum, you have thrash metal, which is known for its fast, aggressive guitar riffs and intense drumming. Bands like Metallica, Slayer, and Megadeth are examples of thrash metal bands.

Beyond thrash metal, you have death metal, which is characterized by brutal, guttural vocals, complex guitar riffs, and often macabre or violent themes. Bands like Death, Cannibal Corpse, and Morbid Angel are examples of death metal bands.

Black metal is another subgenre that emerged from the extreme metal scene in the 1990s. It is characterized by its raw, lo-fi production, tremolo-picked guitar riffs, and often controversial themes of Satanism, paganism, and anti-Christianity. Bands like Mayhem, Emperor, and Darkthrone are examples of black metal bands.

Other subgenres on the metal spectrum include doom metal, progressive metal, and metalcore, each with its own distinct sound, style, and themes.

Overall, the metal spectrum represents a diverse and constantly evolving genre of music that continues to attract fans from all over the world.

detail this spectrum even more from a bass perspective

< Sure! As an AI language model, I can provide information about the metal spectrum from a bass perspective.

Bass is an essential component of metal music, providing the low end and rhythmic foundation for the music. In general, metal bassists use a combination of fingerstyle and picking techniques to achieve a heavy and powerful sound.

At the traditional end of the metal spectrum, bassists play a prominent role in shaping the melodic and harmonic elements of the music. In heavy metal and power metal, bassists often play melodic lines that complement the guitar riffs and help to create a thick, layered sound. In these subgenres, bassists often use a pick to achieve a more precise and aggressive attack.

Moving towards the heavier end of the spectrum, bassists in thrash metal play a more supportive role, providing a driving rhythm section that underpins the fast and aggressive guitar riffs. In death metal, bassists often play complex and intricate lines that follow the guitar riffs and help to create a dense and chaotic sound.

In black metal, bass is often buried in the mix and used more for its textural and rhythmic qualities than its melodic content. In this subgenre, bassists often use a combination of fingerstyle and palm-muted picking to achieve a raw and aggressive sound.

Other subgenres on the metal spectrum, such as doom metal and progressive metal, also feature bass prominently, with bassists playing slow, heavy riffs in doom metal and complex and virtuosic lines in progressive metal.

Overall, the role of the bass in metal music is essential to creating the heavy, powerful sound that defines the genre. From melodic lines to driving rhythms, bassists in metal music play a critical role in shaping the sound of the music.

try to break down metal in 30 items, distributing subgenres where needed

< Sure, here are 30 items that break down the metal genre and its various subgenres:

Heavy Metal Power Metal Thrash Metal Death Metal Black Metal Doom Metal Progressive Metal Metalcore Symphonic Metal Folk Metal

Glam Metal Nu Metal Industrial Metal Gothic Metal Stoner Metal Sludge Metal Grindcore Mathcore Post-Metal Alternative Metal Christian Metal Viking Metal Symphonic Black Metal Technical Death Metal Atmospheric Black Metal Melodic Death Metal Progressive Death Metal Death Doom Metal Traditional Doom Metal Avant-garde Metal Some subgenres are included in multiple items, as they can have characteristics of different subgenres depending on the band and their music. Also, this list is not exhaustive and there may be additional subgenres or hybrid genres that could be included.

Overall, the metal genre is incredibly diverse, with a wide range of subgenres and styles that appeal to different audiences and musical tastes. From the melodic harmonies of power metal to the raw aggression of grindcore, there's something for everyone in the world of metal.

try to break down metal in 40 items, distributing subgenres where needed sorting by normieness rating 0-1000

< Sure, I'll give it a try! Here's a breakdown of metal into 40 items, sorted by normieness rating from 0-1000:

Heavy Metal (Normieness Rating: 600) - characterized by a focus on melody and guitar solos, with bands like Iron Maiden and Judas Priest being popular examples. Power Metal (Normieness Rating: 550) - a subgenre of metal that emphasizes fantasy and mythology themes, with bands like Helloween and DragonForce being popular examples. Thrash Metal (Normieness Rating: 450) - characterized by fast, aggressive guitar riffs and intense drumming, with bands like Metallica, Slayer, and Megadeth being popular examples. Death Metal (Normieness Rating: 300) - characterized by brutal, guttural vocals, complex guitar riffs, and often macabre or violent themes, with bands like Death, Cannibal Corpse, and Morbid Angel being popular examples. Black Metal (Normieness Rating: 200) - known for its raw, lo-fi production, tremolo-picked guitar riffs, and controversial themes of Satanism, paganism, and anti-Christianity, with bands like Mayhem, Burzum, and Darkthrone being popular examples. Nu-Metal (Normieness Rating: 700) - characterized by a fusion of metal and hip-hop influences, with bands like Korn and Limp Bizkit being popular examples. Metalcore (Normieness Rating: 600) - a fusion of metal and hardcore punk, with bands like Killswitch Engage and Converge

being popular examples. Djent (Normieness Rating: 500) - a subgenre of progressive metal characterized by the use of extended-range guitars and intricate, syncopated rhythms, with bands like Meshuggah and Periphery being popular examples. Prog Metal (Normieness Rating: 400) - characterized by complex song structures, odd time signatures, and virtuosic musicianship, with bands like Dream Theater and Tool being popular examples. Doom Metal (Normieness Rating: 300) - characterized by slow, heavy riffs and a melancholic or depressive atmosphere, with bands like Black Sabbath and Candlemass being popular examples. Folk Metal (Normieness Rating: 250) - characterized by the use of folk instruments and themes, with bands like Eluveitie and Finntroll being popular examples. Symphonic Metal (Normieness Rating: 200) - characterized by the use of orchestral elements, choirs, and operatic vocals, with bands like Nightwish and Epica being popular examples. Glam Metal (Normieness Rating: 700) - characterized by a focus on image and theatrics, with bands like Mötley Crüe and Poison being popular examples. Hair Metal (Normieness Rating: 650) - characterized by a focus on catchy hooks and anthemic choruses, with bands like Bon Jovi and Whitesnake being popular examples. Alternative Metal (Normieness Rating: 600) - a catch-all term for metal that incorporates alternative rock influences, with bands like System of a Down and Deftones being popular examples. Industrial Metal (Normieness Rating: 500) - characterized by the use of electronic and industrial sounds, with bands like Nine Inch Nails and Rammstein being popular examples. Groove Metal (Normieness Rating: 400) - characterized by a focus on heavy, groove-oriented riffs, with bands like Pantera and Lamb of God being popular examples.

try to break down metal in 20 items, distributing subgenres where needed sorting by normieness rating 0-1000

< Sure, here's a breakdown of metal into 20 items, sorted by normieness rating from 0-1000:

Heavy metal (normieness rating: 600) - characterized by melodic guitar solos, powerful vocals, and themes of fantasy and mythology. Examples: Iron Maiden, Judas Priest, Black Sabbath. Thrash metal (normieness rating: 650) - known for its fast, aggressive guitar riffs and intense drumming. Examples: Metallica, Slayer, Megadeth. Death metal (normieness rating: 700) - characterized by brutal, guttural vo-

cals, complex guitar riffs, and often macabre or violent themes. Examples: Death, Cannibal Corpse, Morbid Angel. Black metal (normieness rating: 750) - characterized by its raw, lo-fi production, tremolo-picked guitar riffs, and controversial themes of Satanism and anti-Christianity. Examples: Mayhem, Emperor, Darkthrone. Power metal (normieness rating: 550) - characterized by fast-paced guitar solos, soaring vocals, and themes of heroism and fantasy. Examples: Helloween, Blind Guardian, Stratovarius. Progressive metal (normieness rating: 500) - characterized by complex time signatures, virtuosic musicianship, and experimental song structures. Examples: Dream Theater, Opeth, Tool. Doom metal (normieness rating: 450) - characterized by slow, heavy riffs and a melancholic atmosphere. Examples: Black Sabbath (early), Electric Wizard, Candlemass. Folk metal (normieness rating: 400) - characterized by a fusion of metal with traditional folk music from various regions. Examples: Ensiferum, Korpiklaani, Eluveitie. Symphonic metal (normieness rating: 350) - characterized by the use of orchestral instruments and arrangements in combination with heavy metal. Examples: Nightwish, Within Temptation, Epica. Gothic metal (normieness rating: 300) - characterized by a fusion of metal with elements of gothic rock and classical music. Examples: Type O Negative, Paradise Lost, The Gathering. Nu-metal (normieness rating: 200) - characterized by a fusion of metal with elements of hip-hop and alternative rock. Examples: Korn, Limp Bizkit, Linkin Park. Metalcore (normieness rating: 250) - characterized by a fusion of metal with elements of hardcore punk and emo. Examples: Killswitch Engage, As I Lay Dying, Trivium. Industrial metal (normieness rating: 200) - characterized by the use of electronic and industrial music elements in combination with heavy metal. Examples: Nine Inch Nails, Ministry, Rammstein. Glam metal (normieness rating: 500) - characterized by an emphasis on image and theatrics, with a focus on flashy costumes and makeup. Examples: Poison, Motley Crue, W.A.S.P. Grindcore (normieness rating: 300) - characterized by fast, aggressive songs with short, intense bursts of sound. Examples: Napalm Death, Pig Destroyer, Brutal Truth. Stoner metal (normieness rating: 400) - characterized by heavy, fuzzy guitar riffs and a psychedelic influence. Examples: Sleep, Kyuss, Electric Wizard. Crossover thrash (normieness rating: 300) - characterized by a fusion of thrash metal with elements of hardcore punk. Examples: Suicidal Tendencies, D.R.I. (Dirty Rotten Imbeciles), Municipal Waste. Technical death metal (normieness rating: 650) - characterized by complex and intricate song structures, virtuosic musicianship, and often philosophical or scientific themes.

Examples: Necrophagist, Obscura, Beyond Creation. Melodic death metal (normieness rating: 600) - characterized by melodic guitar riffs and harmonies, harsh vocals, and often themes of death and despair. Examples: In Flames, At The Gates, Dark Tranquillity. Post-metal (normieness rating: 350) - characterized by a fusion of metal with post-rock and other experimental music genres, often with a focus on atmospheric and introspective sounds. Examples: Neurosis, ISIS, Cult of Luna. Note that the normieness rating is subjective and may vary depending on personal preferences and cultural context. This list is not exhaustive and there are many more subgenres and variations within the metal spectrum.

the list is out of order btw. can you summarize it sorting it with the normieness rating?

< Sure, here's a summary of the metal subgenres, sorted by normieness rating from highest to lowest:

Black metal (normieness rating: 750) Death metal (normieness rating: 700) Thrash metal (normieness rating: 650) Technical death metal (normieness rating: 650) Heavy metal (normieness rating: 600) Melodic death metal (normieness rating: 600) Power metal (normieness rating: 550) Glam metal (normieness rating: 500) Progressive metal (normieness rating: 500) Stoner metal (normieness rating: 400) Doom metal (normieness rating: 450) Folk metal (normieness rating: 400) Symphonic metal (normieness rating: 350) Post-metal (normieness rating: 350) Gothic metal (normieness rating: 300) Crossover thrash (normieness rating: 300) Grindcore (normieness rating: 300) Metalcore (normieness rating: 250) Industrial metal (normieness rating: 200) Nu-metal (normieness rating: 200)

Again, note that normieness rating is a subjective measurement and may vary depending on personal preferences and cultural context. This list is not exhaustive and there are many more subgenres and variations within the metal spectrum.

break down black metal in 20 subgenres with its normieness rating

< I'm afraid that breaking down Black metal into 20 subgenres would be difficult, as Black metal itself is a relatively narrow and specific

genre with fewer subgenres than other styles of metal. However, I can provide a brief description of some subgenres and offshoots of Black metal that have emerged over the years.

Traditional Black metal (normieness rating: 700) - characterized by raw, lo-fi production, tremolo-picked guitar riffs, and harsh, shrieking vocals. Examples: Mayhem, Darkthrone, Burzum. Atmospheric Black metal (normieness rating: 500) - characterized by a focus on atmosphere and melody, often with a more polished production style. Examples: Emperor, Summoning, Wolves in the Throne Room. Symphonic Black metal (normieness rating: 350) - characterized by the use of orchestral and symphonic elements in combination with Black metal. Examples: Dimmu Borgir, Cradle of Filth, Emperor. Pagan Black metal (normieness rating: 500) - characterized by a focus on pagan and folk themes, often incorporating traditional folk instruments and melodies. Examples: Bathory, Windir, Ensiferum. Blackgaze (normieness rating: 300) - characterized by a fusion of Black metal with elements of shoegaze and post-rock. Examples: Alcest, Deafheaven, Lantlos. War metal (normieness rating: 800) - characterized by a raw and aggressive sound, often with a focus on war and violence themes. Examples: Blasphemy, Revenge, Bestial Warlust. Blackened death metal (normieness rating: 700) - characterized by a fusion of Black metal with death metal elements, often incorporating complex song structures and technical musicianship. Examples: Behemoth, Belphegor, Necrophobic. Blackened thrash metal (normieness rating: 700) - characterized by a fusion of Black metal with thrash metal elements, often incorporating fast, aggressive guitar riffs and intense drumming. Examples: Aura Noir, Sodomizer, Nocturnal. Blackened doom metal (normieness rating: 550) - characterized by a fusion of Black metal with elements of doom metal, often incorporating slow, heavy riffs and a melancholic atmosphere. Examples: My Dying Bride, Woods of Ypres, Nortt. Depressive Black metal (normieness rating: 400) - characterized by a focus on themes of depression and despair, often with slow, mournful melodies and whispered or spoken word vocals. Examples: Xasthur, Shining, Leviathan. Suicidal Black metal (normieness rating: 400) - characterized by a focus on themes of suicide and self-harm, often incorporating spoken word passages and acoustic guitar interludes. Examples: Lifelover, Bethlehem, Silencer. Raw Black metal (normieness rating: 700) - characterized by a deliberately lo-fi production style, often incorporating raw, primitive riffs and aggressive vocals. Examples: Ildjarn, Gorgoroth, Satanic Warmaster. Blackened crust (normieness rating: 500) - characterized by a fusion of Black metal with elements

of crust punk, often incorporating raw, aggressive guitar riffs and a politically charged message. Examples: Iskra, Fall of Efrafa, Agri-
monia. Blackened post-punk (normieness rating: 400) - characterized
by a fusion of Black metal with elements of post-punk and darkwave,
often incorporating atmospheric, ethereal soundscapes and a more sub-
dued vocal style. Examples: Deafheaven, Oathbreaker, Bosse-de-Nage.
Blackened hardcore (normieness rating: 450) - characterized by a fu-
sion of Black metal with elements of hardcore punk, often incorporat-
ing fast, aggressive riffs and screamed vocals. Examples: Young and
in the Way, Hexis, Svalbard. Blackened death-doom (normieness rat-
ing: 550) - characterized by a fusion of Black metal with elements of
death metal and doom metal, often incorporating slow, heavy riffs and
dissonant harmonies. Examples: Mitochondrion, Dead Congregation,
Grave Miasma. Medieval Black metal (normieness rating: 500) - char-
acterized by a focus on medieval and folk themes, often incorporating
traditional folk instruments and medieval melodies. Examples: Inqui-
sition, Peste Noire, Sühnopfer. Occult Black metal (normieness rating:
600) - characterized by a focus on occult and mystical themes, often in-
corporating atmospheric, ritualistic elements into the music. Examples:
Aosoth, Watain, Funeral Mist. Blackened sludge (normieness rating:
400) - characterized by a fusion of Black metal with elements of sludge
metal, often incorporating slow, heavy riffs and a focus on atmosphere
and mood. Examples: Thou, Primitive Man, Eyehategod. Blackened
noise (normieness rating: 300) - characterized by a focus on dissonance,
distortion, and feedback, often incorporating harsh noise elements into
the music. Examples: Gnaw Their Tongues, Prurient, The Body. Note
that these subgenres are not all universally recognized, and there may
be some overlap or disagreement about how to classify certain bands or
styles of Black metal. The normieness ratings are also subjective and
may vary depending on personal preferences and cultural context.

**the list is out of order btw. can you summa-
rize it sorting it with the normieness rating?
(1)**

< Sure, here's a summary of some subgenres and offshoots of Black
metal, sorted by normieness rating from highest to lowest:

War metal (normieness rating: 800) Traditional Black metal (normieness
rating: 700) Blackened thrash metal (normieness rating: 700) Black-
ened death metal (normieness rating: 700) Occult Black metal (normieness

rating: 600) Blackened doom metal (normieness rating: 550) Medieval Black metal (normieness rating: 500) Atmospheric Black metal (normieness rating: 500) Pagan Black metal (normieness rating: 500) Blackened crust (normieness rating: 500) Depressive Black metal (normieness rating: 400) Suicidal Black metal (normieness rating: 400) Raw Black metal (normieness rating: 700) Blackened post-punk (normieness rating: 400) Blackened hardcore (normieness rating: 450) Blackened death-doom (normieness rating: 550) Blackened sludge (normieness rating: 400) Symphonic Black metal (normieness rating: 350) Blackgaze (normieness rating: 300) Blackened noise (normieness rating: 300)

Again, note that normieness rating is a subjective measurement and may vary depending on personal preferences and cultural context. This list is not exhaustive and there may be other subgenres or offshoots of Black metal that have emerged over the years.

detail the non normie end of the spectrum

< The non-normie end of the metal spectrum is characterized by subgenres and styles that are often considered extreme, experimental, or niche within the broader metal scene. These styles tend to appeal to a more selective and dedicated fan base, rather than the mainstream or casual metal listener. Here are some examples:

Noisegrind - characterized by a fusion of grindcore with elements of noise music, often incorporating chaotic, dissonant sounds and non-traditional instruments. Examples: Sissy Spacek, Sore Throat, Anal Cunt.

Raw black metal - characterized by a deliberately lo-fi production style, often incorporating raw, primitive riffs and aggressive vocals. Examples: Ildjarn, Gorgoroth, Satanic Warmaster.

War metal - characterized by a raw and aggressive sound, often with a focus on war and violence themes. Examples: Blasphemy, Revenge, Bestial Warlust.

Blackened noise - characterized by a focus on dissonance, distortion, and feedback, often incorporating harsh noise elements into the music. Examples: Gnaw Their Tongues, Prurient, The Body.

Funeral doom - characterized by slow, heavy riffs and a mournful, atmospheric sound, often incorporating funeral or death-related themes. Examples: Esoteric, Ahab, Mournful Congregation.

Avant-garde metal - characterized by a willingness to experiment with unconventional sounds, song structures, and instrumentation, often incorporating elements of jazz, classical music, or electronic music.

Examples: Diablo Swing Orchestra, Mr. Bungle, Sigh.

Technical death metal - characterized by complex and intricate song structures, virtuosic musicianship, and often philosophical or scientific themes. Examples: Necrophagist, Obscura, Beyond Creation.

Blackened death metal - characterized by a fusion of black metal with death metal elements, often incorporating complex song structures and technical musicianship. Examples: Behemoth, Belphegor, Necrophobic.

Blackened thrash metal - characterized by a fusion of black metal with thrash metal elements, often incorporating fast, aggressive guitar riffs and intense drumming. Examples: Aura Noir, Sodomizer, Nocturnal.

Blackened crust - characterized by a fusion of black metal with elements of crust punk, often incorporating raw, aggressive guitar riffs and a politically charged message. Examples: Iskra, Fall of Efrafa, Agrimonia.

Note that these subgenres and styles are not necessarily “better” or “worse” than more mainstream or normie styles of metal; they simply appeal to a different audience and often have a more specialized or underground following.

detail the non normie end of the black metal spectrum

< At the non-normie end of the Black metal spectrum, you can find some of the most extreme and underground subgenres and offshoots of the genre. These styles are often characterized by a deliberately raw and primitive sound, and a focus on extreme, confrontational themes and imagery. Here are some examples:

Raw Black metal - characterized by a deliberately lo-fi production style, often incorporating raw, primitive riffs and aggressive vocals. Examples: Ildjarn, Gorgoroth, Satanic Warmaster.

Noisecore - characterized by an emphasis on noise, distortion, and chaos, often incorporating elements of harsh noise and power electronics. Examples: Gnaw Their Tongues, Vomir, Failing Lights.

Blacknoise - characterized by a focus on dissonance, feedback, and harsh noise, often incorporating elements of power electronics and avant-garde music. Examples: Prurient, Yellow Swans, Hair Police.

Blackened death metal - characterized by a fusion of Black metal with death metal elements, often incorporating complex song struc-

tures and technical musicianship. Examples: Behemoth, Belphegor, Necrophobic.

Experimental Black metal - characterized by a willingness to incorporate elements of other genres and styles into the music, often resulting in a highly eclectic and unpredictable sound. Examples: Deathspell Omega, Blut Aus Nord, Dodecahedron.

Blackened grindcore - characterized by a fusion of Black metal with elements of grindcore and extreme punk, often incorporating fast, aggressive riffs and short, intense songs. Examples: Anaal Nathrakh, Black Witchery, Angelcorpse.

Raw noise Black metal - characterized by a fusion of Black metal with harsh noise, often incorporating elements of power electronics and experimental music. Examples: Sutekh Hexen, Ash Borer, Yellow Eyes.

Note that these subgenres are not all universally recognized, and there may be some overlap or disagreement about how to classify certain bands or styles of Black metal. The non-normie end of the Black metal spectrum can be difficult and challenging to listen to, but for fans of extreme and underground music, it offers a world of intense, confrontational, and uncompromising soundscapes.

detail the female-only black metal bands sorting by non normieness

< Sure, here are some female-only Black metal bands, sorted by non-normieness:

Ethereal Shroud - a UK-based band that combines elements of atmospheric Black metal, funeral doom, and ambient music. The band is known for their haunting and ethereal soundscapes and the use of female vocals to create a sense of melancholy and despair. Astarte - a Greek Black metal band that formed in 1995. The band is known for their aggressive and fast-paced Black metal sound, as well as the use of symphonic elements and female vocals to create a more melodic and atmospheric sound. Jarboe - an American singer and musician who has collaborated with several Black metal bands over the years, including Neurosis and Darkthrone. Her music incorporates elements of experimental, industrial, and noise music, as well as Black metal. Myrkur - a Danish Black metal musician who is known for her haunting vocals and atmospheric soundscapes. Her music incorporates elements of folk music, classical music, and ambient music, as well as Black metal. Smoulder - a Canadian Black metal band that incorporates elements

of traditional heavy metal and epic doom metal into their sound. The band is known for their powerful and dynamic female vocals, as well as their use of intricate guitar melodies and soaring solos. Sighisoara - a Romanian Black metal band that formed in 2014. The band is known for their dark and atmospheric sound, as well as their use of female vocals to create a sense of mystery and mysticism. Asagraum - a Dutch Black metal band that formed in 2015. The band is known for their raw and aggressive Black metal sound, as well as the use of female vocals to create a more melodic and atmospheric sound. Abigorum - a Russian Black metal band that formed in 2018. The band is known for their atmospheric and melancholic soundscapes, as well as the use of female vocals to create a sense of beauty and sorrow. Note that the concept of “norminess” can be subjective, so the sorting of these bands by non-norminess is not an exact science. Additionally, this list is not exhaustive and there may be other female-only Black metal bands out there.

detail the female-only death metal bands sorting by non normieness

< Sure, here are some female-only death metal bands, sorted by non-normieness rating from highest to lowest:

Nervosa - a Brazilian thrash/death metal band formed in 2010, known for their aggressive and fast-paced music and socially conscious lyrics. Crypta - a Brazilian death metal band formed in 2019, featuring former members of Nervosa and Burning Witches. Their music is characterized by powerful vocals, heavy riffs, and intricate drumming. Mortuary - a French death metal band formed in 1985, with a sound that combines elements of thrash and death metal. Their music is characterized by technical riffs, fast-paced drumming, and brutal vocals. Derkéta - an American death metal band formed in 1988, considered one of the first all-female death metal bands. Their music is characterized by slow, heavy riffs, and growled vocals. Mythic - an American death metal band formed in 1989, known for their raw and intense sound and lyrics inspired by mythology and horror. Their music is characterized by aggressive riffs, pounding drums, and guttural vocals. Abnormality - an American death metal band formed in 2005, with a sound that combines elements of technical death metal and brutal death metal. Their music is characterized by complex rhythms, intricate guitar work, and powerful vocals. Gallhammer - a Japanese

death/doom metal band formed in 2003, known for their dark and atmospheric sound and haunting vocals. Their music is characterized by slow, droning riffs and eerie melodies. Sinister - a Dutch death metal band formed in 1988, with a sound that combines elements of death and black metal. They have had several female vocalists throughout their career, including Rachel Heyzer and Michelle Nocon. Astarte - a Greek blackened death metal band formed in 1995, known for their melodic guitar work and operatic vocals. Their music also incorporates elements of symphonic and folk metal. Oceans of Slumber - an American progressive death metal band formed in 2011, with a sound that combines elements of death, doom, and progressive metal. Their music is characterized by intricate guitar work, atmospheric keyboards, and emotive vocals. Note that this list is not exhaustive, and there are many other female-only death metal bands out there. Additionally, the non-normieness rating is subjective and may vary depending on personal preferences and cultural context.

having crypta as basis, list the female-only black and death metal bands sorting by amount of albums or recent activity

< Sure, here are some female-only black and death metal bands, sorted by amount of albums or recent activity, with Crypta as the basis:

Sister Sin (Sweden) - a female-fronted heavy metal band formed in 2002, with a sound that combines elements of traditional heavy metal and hard rock. They have released six studio albums, with the most recent being "Black Lotus" in 2014. Astarte (Greece) - a blackened death metal band formed in 1995, with a sound that incorporates elements of symphonic and folk metal. They have released six studio albums, with the most recent being "Demonized" in 2007. The Agonist (Canada) - a melodic death metal band formed in 2004, with a sound that combines elements of death metal, metalcore, and progressive metal. They have released six studio albums, with the most recent being "Orphans" in 2019. Nervosa (Brazil) - a thrash/death metal band formed in 2010, known for their aggressive and fast-paced music and socially conscious lyrics. They have released four studio albums, with the most recent being "Perpetual Chaos" in 2021. Mythic (United States) - a death metal band formed in 1989, known for their raw and intense sound and lyrics inspired by mythology and horror. They released one full-length album, "Mourning in the Winter Solstice", in 1992 before disbanding

in 1992. Derkéta (United States) - a death metal band formed in 1988, considered one of the first all-female death metal bands. They released two demo tapes and a split album before disbanding in 1991, but reunited in 2006 and released a full-length album, "In Death We Meet", in 2012. Smoulder (Canada) - a traditional doom metal band formed in 2013, with a sound that incorporates elements of epic and heavy metal. They have released two studio albums, with the most recent being "Dream Quest Ends" in 2021. Note that this list is not exhaustive, and there are many other female-only black and death metal bands out there with varying amounts of albums and recent activity.

detail the non normie end of the black metal spectrum sorting by bpm range

< The non-normie end of the Black metal spectrum can be characterized by extreme tempos, unconventional song structures, and a disregard for traditional production values. Here are some subgenres and offshoots of Black metal that fit into this category, sorted by BPM range:

Raw Black metal (BPM range: 100-180) - characterized by lo-fi production, primitive riffs, and harsh, abrasive vocals. Examples: Ildjarn, Gorgoroth, Satanic Warmaster. War metal (BPM range: 180-280) - characterized by frenzied blast beats, dissonant harmonies, and a focus on themes of war and violence. Examples: Blasphemy, Revenge, Bestial Warlust. Blackened noise (BPM range: 80-200) - characterized by a focus on dissonance, feedback, and extreme sound manipulation, often incorporating elements of harsh noise into the music. Examples: Gnaw Their Tongues, Prurient, The Body. Blackened grindcore (BPM range: 200-300) - characterized by a fusion of Black metal with elements of grindcore, often incorporating fast, aggressive riffs and shouted or screamed vocals. Examples: Anaal Nathrakh, Tsjuder, Nidingr. Blackened crust (BPM range: 120-240) - characterized by a fusion of Black metal with elements of crust punk, often incorporating raw, aggressive guitar riffs and a politically charged message. Examples: Iskra, Fall of Efrafa, Agrimonia. Raw black punk (BPM range: 120-200) - characterized by a fusion of Black metal with elements of punk rock, often incorporating raw, aggressive riffs and shouted or screamed vocals. Examples: Darkthrone, Venom, Bathory. Blackened deathgrind (BPM range: 200-300) - characterized by a fusion of Black metal with elements of death metal and grindcore, often incorporating fast, ag-

gressive riffs and brutal, guttural vocals. Examples: Beherit, Archgoat, Angelcorpse. Note that these subgenres are not universally recognized and there may be some overlap or disagreement about how to classify certain bands or styles of Black metal. Additionally, BPM range can be subjective and may vary depending on personal interpretation and cultural context.

detail the non normie end of the black metal spectrum sorting by minimum bpm

< Sure, here's a breakdown of some non-normie subgenres of Black metal sorted by minimum bpm:

Raw Black metal - typically played at slow to mid-tempo speeds, with a focus on dissonant riffs and harsh, shrieking vocals. Examples: Ildjarn, Beherit, Darkthrone. Depressive Black metal - typically played at mid to slow tempos, with a focus on melancholic melodies and introspective lyrics. Examples: Xasthur, Shining, Make a Change... Kill Yourself. Blackened noise - often characterized by a focus on dissonance, distortion, and feedback, with tempos ranging from slow and atmospheric to fast and chaotic. Examples: Gnaw Their Tongues, Prurient, Abruptum. Blackened death-doom - characterized by a fusion of Black metal with elements of death metal and doom metal, often incorporating slow, heavy riffs and dissonant harmonies. Examples: Mitochondrion, Dead Congregation, Grave Miasma. War metal - characterized by fast, aggressive tempos, often with a focus on violent, militaristic themes. Examples: Blasphemy, Revenge, Conqueror. Blackened thrash metal - characterized by fast, aggressive tempos, with a focus on thrash-influenced riffs and aggressive vocals. Examples: Aura Noir, Nocturnal, Desaster. Blackened hardcore - characterized by fast, aggressive tempos, often incorporating elements of hardcore punk and crust punk. Examples: Young and in the Way, Hexis, Svalbard. Blackened grindcore - characterized by fast, chaotic tempos, with a focus on short, intense bursts of aggression. Examples: Anaal Nathrakh, Antichrist, Blut Aus Nord. Note that these subgenres are not universally recognized, and there may be some overlap or disagreement about how to classify certain bands or styles of Black metal. Additionally, while the tempos of these subgenres may generally fall on the non-normie end of the spectrum, there are certainly bands within these subgenres that play at more moderate or even fast tempos.

break down black metal in 12 subgenres with its normieness rating

< Black metal is a subgenre of heavy metal music that originated in Norway in the early 1990s. It is characterized by its raw, lo-fi production, tremolo-picked guitar riffs, and controversial themes of Satanism and anti-Christianity. Here's a breakdown of black metal into 12 subgenres, with their normieness rating:

Norwegian black metal (normieness rating: 900) - characterized by its raw, lo-fi production, tremolo-picked guitar riffs, and themes of Norse mythology and anti-Christianity. Examples: Mayhem, Burzum, Darkthrone. Swedish black metal (normieness rating: 800) - characterized by its melodic guitar riffs and atmospheric soundscapes. Examples: Dissection, Marduk, Watain. French black metal (normieness rating: 700) - characterized by its use of melodic and avant-garde elements, as well as a focus on philosophical and literary themes. Examples: Blut Aus Nord, Deathspell Omega, Alcest. American black metal (normieness rating: 650) - characterized by its fusion of black metal with other genres, such as death metal and post-rock. Examples: Wolves in the Throne Room, Liturgy, Agalloch. Finnish black metal (normieness rating: 600) - characterized by its use of melodic guitar riffs and atmospheric soundscapes, as well as a focus on nature and pagan themes. Examples: Beherit, Horna, Sargeist. Canadian black metal (normieness rating: 500) - characterized by its fusion of black metal with post-rock and shoegaze elements, as well as a focus on atmospheric and introspective soundscapes. Examples: Woods of Ypres, Thrawsunblat, Panopticon. Atmospheric black metal (normieness rating: 450) - characterized by its emphasis on creating a dark, introspective atmosphere through the use of ambient soundscapes, clean vocals, and melodic guitar riffs. Examples: Drudkh, Fen, Saor. Raw black metal (normieness rating: 400) - characterized by its use of raw, lo-fi production and abrasive, chaotic sound. Examples: Ildjarn, Müttilation, Krieg. Pagan black metal (normieness rating: 350) - characterized by its focus on pagan and folk themes, as well as its use of traditional folk instruments and melodies. Examples: Bathory, Primordial, Borknagar. Blackgaze (normieness rating: 300) - characterized by its fusion of black metal with shoegaze and post-rock elements, as well as a focus on atmospheric and introspective soundscapes. Examples: Deafheaven, Alcest, Lantlôs. Depressive black metal (normieness rating: 250) - characterized by its emphasis on themes of despair, depression, and suicide, as well as its use of clean vocals and melodic guitar

riffs. Examples: Xasthur, Shining, Lifelover. Symphonic black metal (normieness rating: 200) - characterized by its use of orchestral instruments and arrangements in combination with black metal. Examples: Dimmu Borgir, Emperor, Cradle of Filth. Note that normieness rating is a subjective measurement and may vary depending on personal preferences and cultural context. This list is not exhaustive and there are many more subgenres and variations within the black metal spectrum.

list 15 symphonic black metal bands in mark-down table form with 6 columns with relevant stats and remarks for each

< Sure! Here's a table of 15 symphonic black metal bands with relevant stats and remarks:

Band	Origin	Active Years	Discography	Notable Tracks	Remarks
Dimmu Borgir	Norway	1993-Present	10 studio albums	“Progenies of the Great Apocalypse”, “Mourning Palace”	One of the most well-known and influential symphonic black metal bands
Emperor	Norway	1991-2001, 2005, 2013-Present	4 studio albums	“Inno a Satana”, “I Am the Black Wizards”	Considered one of the pioneers of the genre
Cradle of Filth	England	1991-Present	13 studio albums	“Her Ghost in the Fog”, “Nymphetaria Fix”	Known for incorporating gothic and horror elements into their music
Old Man’s Child	Norway	1993-2009	9 studio albums	“The Plague of Sorrow”, “In Black Endless Void”	Project of Dimmu Borgir’s guitarist, Galder
Carach Angren	Netherlands	2003-Present	6 studio albums	“When Crows Tick on Windows”, “Blood	Known for their horror-themed lyrics and

Note that the stats and remarks are based on general consensus and may vary depending on personal opinions and preferences.

list 12 non repeating symphonic death metal bands in markdown table form with 6 columns with relevant stats and remarks for each

< Sure, here's a table of 12 symphonic death metal bands with relevant stats and remarks for each:

Band	Country	Active Years	Number of Albums	Popular Songs	Notable Features
Fleshgod Apocalypse	Italy	2007-present	5	The Violation, The Forsaking	Blend of death metal, symphonic and operatic elements
Septicflesh	Greece	1990-2003, 2007-present	10	Anubis, Prometheus	Use of choirs and orchestras, incorporating various musical styles
Carach Angren	Netherlands	2003-present	6	When Crows Tick on Windows, Bloodstains on the Captain's Log	Horror-themed lyrics, theatrical live shows
Chthonic	Taiwan	1995-present	8	Takao, Supreme Pain for the Tyrant	Use of traditional Taiwanese instruments, incorporation of Taiwanese mythology
MaYaN	Netherlands	2010-present	4	Dhyana, The	Features multiple

Note that these bands are subjective and there may be other symphonic death metal bands worth mentioning. The table is not exhaustive and the remarks are brief, intended to give a general idea of each band's style and notable features.

list 12 blackgaze bands in markdown table form with 6 columns with relevant stats and remarks for each

< Here's a markdown table featuring 12 blackgaze bands with relevant stats and remarks for each:

Band Name	Origin	Years Active	Number of Albums	Popular Songs	Remarks
Deafheaven	San Francisco, California	2010 - Present	4	“Dream House”, “Sunbather”, “Honeycomb”	One of the most popular and influential blackgaze bands, known for their unique fusion of black metal and shoegaze elements.
Alcest	Bagnols-sur-Cèze, France	2000 - Present	6	“Écailles de Lune”, “Autre Temps”, “Oiseaux de Proie”	Led by frontman Neige, Alcest is widely regarded as one of the pioneers of the blackgaze sub-genre.
Lantlôs	Rhineland-Palatinate, Germany	2005 - Present	4	“Melting Sun I: Azure Chimes”, “Pulse/Surreal”, “Lake Fantasy”	Known for their intricate guitar work and progressive song structures, Lantlôs

list 12 depressive black metal bands in markdown table form with 6 columns with relevant stats and remarks for each

< Sure, here is a markdown table with 12 depressive black metal bands, along with relevant stats and remarks for each:

Band Name	Country	Formed	Active	Most Popular Album	Remarks
Xasthur	USA	1995	No	“Subliminal Geno- cide” (2006)	Known for its raw, lo-fi production and depressive, atmospheric soundscapes.
Shining	Sweden	1996	Yes	“V - Halmstad” (2007)	Known for its intense and emotional live performances, as well as its controversial lyrics and themes.
Leviathan	USA	1998	Yes	“The Tenth Sub Level of Suicide” (2003)	Known for its experimental approach to black metal, incorporating elements of noise, ambient, and drone

list 12 pagan black metal bands in mark-down table form with 6 columns with relevant stats and remarks for each

< Sure, here's a table of 12 pagan black metal bands with 6 columns of relevant stats and remarks for each:

Band	Country	Year formed	Key members	Notable albums	Remarks
Bathory	Sweden	1983	Quorthon	Under the Sign of the Black Mark (1987), Hammerheart (1990)	Considered one of the pioneers of the genre and a major influence on subsequent pagan black metal bands.
Primordial	Ireland	1987	Alan Averill (Nemtheangraing)	The Gath-Wilderness (2005), Redemption at the Puritan's Hand (2011)	Blends black metal with elements of traditional Irish music and literature, creating a unique and evocative sound.
Borknagar	Norway	1995	Øystein Garnes Brun, ICS Vortex	The Olden Domain (1997), Empiricism (2001)	Combines black metal with elements of progressive and folk music, creating

**list 12 active bands similar to bathory in
markdown table form with 6 columns with
relevant stats and remarks for each**

< Sure, here's a list of 12 active bands similar to Bathory in markdown table form with 6 columns, including relevant stats and remarks for each:

Band Name	Country of Origin	Active Years	Genre	Record Label	Remarks
Darkthrone	Norway	1986-present	Black metal	Peaceville Records	Formed around the same time as Bathory, known for their lo-fi production and raw sound.
Marduk	Sweden	1990-present	Black metal	Century Media Records	Known for their aggressive sound and war-related lyrics, similar to Bathory's later albums.
Immortal	Norway	1990-present	Black metal	Nuclear Blast Records	Influenced by Bathory's epic sound, known for their use of blast beats and icy atmospheres.
Enslaved	Norway	1991-present	Black/Viking metal	Nuclear Blast Records	Known for their fusion of black metal with

Note that this is not an exhaustive list of bands similar to Bathory, and the categorization of each band's genre may vary depending on personal opinions and interpretations.

list 12 active bands similar to death in markdown table form with 6 columns with relevant stats and remarks for each

< Sure, here's a list of 12 active bands similar to Death in markdown table form with 6 columns, including relevant stats and remarks for each:

Band Name	Country of Origin	Active Years	Genre	Record Label	Remarks
Atheist	United States	1984-present	Progressive death metal	Season of Mist	Known for their complex song structures and use of jazz fusion elements, similar to Death's later albums.
Cynic	United States	1987-present	Progressive death metal	Season of Mist	Known for their fusion of death metal with jazz fusion and progressive rock elements, similar to Death's later albums.
Necrophagist	Germany	1992-2010	Technical death metal	Relapse Records	Known for their virtuosic musicianship and intricate song structures,

Note that this is not an exhaustive list of bands similar to Death, and the categorization of each band's genre may vary depending on personal opinions and interpretations.

list 12 active bands similar to dimmu borgir in markdown table form with 6 columns with relevant stats and remarks for each

< Sure, here's a list of 12 active bands similar to Dimmu Borgir in markdown table form with 6 columns, including relevant stats and remarks for each:

Band Name	Country of Origin	Active Years	Genre	Record Label	Remarks
Cradle of Filth	United Kingdom	1991-present	Symphonic black/gothic metal	Nuclear Blast Records	Known for their theatrical, operatic sound and vampiric themes, similar to Dimmu Borgir's early work.
Septicflesh	Greece	1990-present	Symphonic death metal	Season of Mist Records	Known for their use of orchestral elements and epic soundscapes, similar to Dimmu Borgir's later work.
Fleshgod Apocalypse	Italy	2007-present	Symphonic death metal	Nuclear Blast Records	Known for their blend of classical music and extreme metal, similar to Dimmu Borgir's later

Note that this is not an exhaustive list of bands similar to Dimmu Borgir, and the categorization of each band's genre may vary depending on personal opinions and interpretations.

list 12 active bands similar to cradle of filth in markdown table form with 6 columns with relevant stats and remarks for each

< Sure, here's a list of 12 active bands similar to Cradle of Filth in markdown table form with 6 columns, including relevant stats and remarks for each:

Band Name	Country of Origin	Active Years	Genre	Record Label	Remarks
Dimmu Borgir	Norway	1993-present	Symphonic black metal	Nuclear Blast Records	Known for their use of orchestral arrangements and atmospheric soundscapes, similar to Cradle of Filth's later albums.
Carach Angren	Netherlands	2003-present	Symphonic black metal	Season of Mist	Known for their horror-inspired lyrics and use of orchestral elements, similar to Cradle of Filth's "Cruelty and the Beast" album.
Septicflesh	Greece	1990-present	Symphonic death metal	Season of Mist	Known for their use of orchestral and choral arrangements.

Note that this is not an exhaustive list of bands similar to Cradle of Filth, and the categorization of each band's genre may vary depending on personal opinions and interpretations.

list 12 active bands similar to alcest in markdown table form with 6 columns with relevant stats and remarks for each

< Sure, here's a list of 12 active bands similar to Alcest in markdown table form with 6 columns, including relevant stats and remarks for each:

Band Name	Country of Origin	Active Years	Genre	Record Label	Remarks
Deafheaven	United States	2010-present	Blackgaze/metal	Sargent House	Known for their blend of black metal and shoegaze with atmospheric soundscapes and clean vocals similar to Alcest.
Lantlôs	Germany	2005-present	Blackgaze/metal	Prophecy Productions	Known for their atmospheric and introspective sound with influences of post-rock and black metal similar to Alcest.
Les Discrets	France	2003-present	Post-metal/Post-rock	Prophecy Productions	Known for their use of atmospheric soundscapes, clean vo-

Note that this is not an exhaustive list of bands similar to Alcest, and the categorization of each band's genre may vary depending on personal opinions and interpretations.

**by 2021, list the top 10 most advanced DAWs
in markdown table format with specs spread
in columns**

< I cannot provide you with real-time information as my training data only goes up to September 2021. However, I can provide you with a list of the top 10 most advanced Digital Audio Workstations (DAWs) based on the information available up to 2021. Please note that this information may not accurately reflect the current state of the market.

Rank	DAW Name	Supported Platforms	Notable Features	Technology/Framework
1	Ableton Live	Windows, macOS	Real-time audio manipulation, extensive MIDI support, Max for Live integration, Session View for live performances	C++, Python, Max for Live (Max/MSP)
2	FL Studio	Windows, macOS	Step sequencer, Piano roll, Vectorial UI, Flexible audio routing, Lifetime free updates	C++, Delphi
3	Logic Pro	macOS	Advanced audio manipulation, powerful MIDI editing,Alchemy synthesizer, Flex Pitch and Flex Time	Objective-C, Swift, C++
4	Pro Tools	Windows, macOS	Industry-standard audio production, Avid hardware integration, Elastic Audio, powerful	C++, AAX (Avid Audio eX-tension)

list the most advanced options that are open source, have projects with open formats for parsing or are using things like rust

< As of 2021, here's a list of advanced open-source DAWs or those using open formats for parsing or cutting-edge technology like Rust:

DAW Name	Supported Platforms	Notable Features	Technology/Framework
Ardour	Windows, macOS, Linux	Professional-grade audio editing, flexible signal routing, extensive plugin support	C++, LV2 (LADSPA Version 2)
LMMS	Windows, macOS, Linux	Pattern-based workflow, built-in instruments, VST support	C++, Qt, VST (Virtual Studio Technology)
Audacity	Windows, macOS, Linux	Audio editing and recording, basic mixing capabilities, wide range of supported formats	C++, wxWidgets
Hydrogen	Windows, macOS, Linux	Drum machine and pattern-based sequencer, layer-based samples, open file format	C++, Qt, XML (Project File Format)
Zrythm	Windows, macOS, Linux	Fully-featured DAW, intuitive UI, plugin support, automation	C, GTK, LV2 (LADSPA Version 2)
Tracktion T7	Windows, macOS, Linux	Single-screen interface, clip-based effects, built-in sampler, automation	C++, JUCE, VST (Virtual Studio Technology)
Soundation Studio	Web-based	Cloud-based DAW, built-in instruments, real-time collaboration	Web-based, Web Audio API

list the best places to download full bitwig projects

< As of 2021, there are several online platforms and communities where you can find and download full Bitwig Studio projects. Note that some of these resources may require user registration or may have changed since then. Here are some of the popular sources:

Bitwig Community Projects: Bitwig's official website has a collection of projects shared by the community. These projects can be downloaded and opened directly in Bitwig Studio. Visit <https://www.bitwig.com/community/community-projects/>

Blend.io: Blend.io is an online platform for music collaboration, where users can share and download projects from various DAWs, including Bitwig Studio. Visit <https://blend.io/explore/bitwig>

r/Bitwig: The Bitwig Studio subreddit is an active community of Bitwig users who occasionally share their projects and collaborate with other members. Visit <https://www.reddit.com/r/Bitwig/>

KVR Audio Forum: The KVR Audio forum has a Bitwig-specific section where users can share projects, discuss tips and techniques, and seek help. Visit <https://www.kvraudio.com/forum/viewforum.php?f=259>

YouTube: Many Bitwig Studio users and content creators share their projects in the video descriptions of their YouTube tutorials, demos, or performances...

rate 0-1000 these in ability to produce intricate 180bpm hi-tech psytrance tracks in markdown table format. fl studio, ardour, bitwig, logic, ableton, pro tools, studio one, cubase, audacity

< Rating the ability of DAWs to produce intricate 180 BPM hi-tech psytrance tracks on a scale of 0-1000 can be subjective, as it depends on the user's experience and familiarity with each DAW, as well as the availability of specific features and plugins. However, here's a general rating based on each DAW's features, flexibility, and popularity within the electronic music production community:

DAW Name	Rating
FL Studio	900
Ardour	650
Bitwig	950
Logic Pro	850
Ableton Live	1000
Pro Tools	750
Studio One	800
Cubase	900
Audacity	300

Please note that this rating is based on a general overview and is not definitive. The actual suitability of a DAW for producing hi-tech psytrance tracks depends on the individual's preferences, experience, and specific requirements.

detail this 50 disparity between ableton and bitwig

< The 50-point disparity in the rating between Ableton Live and Bitwig Studio can be attributed to a few factors, including user base, popularity in the electronic music community, and some specific features that may be more prevalent in Ableton Live. Keep in mind that these ratings are subjective, and personal experience and preferences may alter these ratings.

Popularity and User Base: Ableton Live has a larger user base and is more popular within the electronic music production community, including psytrance producers. As a result, there are more tutorials, resources, and community support available for Ableton Live users compared to Bitwig Studio.

Max for Live: Ableton Live comes with Max for Live, which is a visual programming environment that enables users to create custom devices, effects, and instruments. This provides additional flexibility and customization options for users, which can be beneficial for producing intricate psytrance tracks.

Native Instruments and Effects: While both Ableton Live and Bitwig Studio come with a comprehensive suite of native instruments and effects, some users may prefer the specific sound or workflow of Ableton Live's devices. For instance, Live's Operator and Wavetable synthesizers are popular for creating complex psytrance sounds, and its audio effect racks provide a flexible way to process and manipulate audio.

Workflow: Some producers may find Ableton Live's workflow more intuitive or efficient for their specific needs when creating complex psytrance tracks. Live's Session View, for example, allows for real-time clip launching and manipulation, which can be a powerful tool for creating intricate arrangements and transitions.

However, it's essential to note that Bitwig Studio is also a powerful DAW with a unique feature set, such as its advanced modulation system, hybrid tracks, and device nesting. Some users may find these features more appealing or suitable for their psytrance production needs. The best DAW choice ultimately depends on personal preferences, workflow requirements, and familiarity with the software.

drawing from all daws, detail the 10 state of the art features that bitwig could introduce in its next major version to stabilize score 1000

< Here are 10 state-of-the-art features that Bitwig Studio could potentially introduce in its next major version to further strengthen its position as a leading DAW:

Integrated Pitch and Time Correction: Introduce a native pitch and time correction tool, similar to Logic Pro's Flex Pitch or Cubase's VariAudio, to enable users to easily manipulate pitch and timing without relying on third-party plugins.

Advanced Vocal Comping: Improve the comping workflow by offering a dedicated comping feature for recording multiple takes and seamlessly merging the best parts of each take into a single performance.

Advanced Sampler: Develop a more advanced, native sampler instrument with granular synthesis capabilities, time-stretching, and advanced modulation options for deep sound design and manipulation.

Audio Random Access (ARA) Support: Introduce ARA support for seamless integration with Melodyne and other ARA-compatible plugins, providing a smooth and efficient editing experience for pitch and time correction.

Dynamic Audio Clip Effects: Implement real-time, non-destructive audio clip effects, allowing users to apply and automate effects directly onto audio clips within the timeline, similar to Ableton Live's audio effect racks.

MPE (MIDI Polyphonic Expression) Improvements: Enhance MPE support and integration, providing a more intuitive workflow for users working with MPE controllers and devices.

Convolution Reverb: Introduce a high-quality convolution reverb effect, allowing users to import custom impulse responses and create realistic, space-based effects.

Improved GUI and Visual Feedback: Enhance the user interface to provide better visual feedback during mixing and editing, such as a more advanced mixer view with customizable channel strips and integrated metering.

Enhanced Automation and Modulation: Improve the automation and modulation workflow with features such as automation curves, relative automation modes, and more advanced modulation routing options.

Enhanced Collaboration Features: Introduce better real-time collaboration options, allowing multiple users to work on the same project simultaneously, either through cloud-based services or a local network connection.

Please note that this list is speculative and is based on the features available in other DAWs and the general advancements in audio production technology up to September 2021. Bitwig Studio's actual development roadmap and future features may differ from this list.

if a new text-based daw using rust were to be written from scratch inspiring in tools like tidal cycles, sonic pi, bitwig, serum, etc with native blockchain integration, could you detail how its architecture could be initiated?

< Creating a text-based DAW using Rust, inspired by tools like Tidal Cycles, Sonic Pi, Bitwig, Serum, and incorporating native blockchain integration is an ambitious project. Here's a high-level outline of how its architecture could be initiated:

Core Audio and MIDI Engine: Start by implementing the core audio and MIDI engine, which will handle audio processing, MIDI events, and plugin hosting. You may consider using existing Rust audio libraries, such as CPAL (Cross-Platform Audio Library), or Rodio for

audio playback, and midir for MIDI I/O.

Plugin Support: Integrate support for audio plugins, such as VST3 or LV2. This will allow users to extend the DAW's capabilities with third-party synthesizers, samplers, and effects processors. You may use the `vst-rs` or `lv2-rs` libraries for this purpose.

Text-based Interface: Design a text-based interface for user input, resembling the code-based environments of Tidal Cycles or Sonic Pi. You can use a Rust library like `crossterm` or `termion` to handle terminal I/O and user input in a cross-platform manner.

Domain-Specific Language (DSL): Develop a domain-specific language (DSL) to enable users to interact with the DAW using code. This language should provide an expressive syntax for creating, manipulating, and sequencing sounds, as well as managing MIDI events, automation, and modulation.

Live Coding Environment: Build a live coding environment that allows users to evaluate and execute code in real-time, enabling them to make changes to their compositions and sound design on-the-fly.

Wavetable Synthesis: Integrate a native wavetable synthesizer inspired by Serum. This would involve implementing wavetable oscillators, various modulation options, and effects processing.

Modular Architecture: Implement a modular architecture similar to Bitwig Studio's device nesting and modulation system. This will allow users to create complex audio processing chains and flexible modulation routings.

Blockchain Integration: Incorporate native blockchain integration for decentralized collaboration, rights management, and royalty tracking. This may involve using a Rust library for interacting with a specific blockchain platform, such as the Substrate framework for building blockchains in Rust.

Presets and Project Management: Design a system for managing presets, samples, and projects, allowing users to save and load their work seamlessly. This may involve using serialization formats like JSON or YAML.

Documentation and Community: Lastly, create extensive documentation and tutorials for the DAW, including examples and best practices for using the text-based interface and DSL. Establish a community forum or platform for users to share their creations, ask questions, and collaborate on projects.

Keep in mind that this is a high-level outline, and the actual implementation will require a detailed understanding of audio programming, Rust development, and the specific technologies involved.

detail the architecture if the DSL were to be the ml language spiral (which already compiles to f# and rust through fable)

< If the DSL for the text-based DAW were to be the Spiral language, which compiles to F# and Rust through Fable, the architecture would need some adjustments to accommodate Spiral's specific features and capabilities. Here's an updated high-level outline:

Core Audio and MIDI Engine: Start by implementing the core audio and MIDI engine, which will handle audio processing, MIDI events, and plugin hosting. Since Spiral compiles to Rust, you can use existing Rust audio libraries, such as CPAL (Cross-Platform Audio Library) or Rodio for audio playback, and midir for MIDI I/O.

Plugin Support: Integrate support for audio plugins, such as VST3 or LV2. This will allow users to extend the DAW's capabilities with third-party synthesizers, samplers, and effects processors. You may use the vst-rs or lv2-rs libraries for this purpose.

Spiral Integration: Integrate the Spiral language into the DAW to create a domain-specific language (DSL) for user interaction. You'll need to provide an API for Spiral to interact with the core audio and MIDI engine, plugin support, and other DAW components.

Text-based Interface: Design a text-based interface for user input, where users can write and execute Spiral code. You can use a Rust library like crossterm or termion to handle terminal I/O and user input in a cross-platform manner.

Live Coding Environment: Build a live coding environment that allows users to evaluate and execute Spiral code in real-time, enabling them to make changes to their compositions and sound design on-the-fly.

Wavetable Synthesis: Implement a native wavetable synthesizer inspired by Serum. Expose the synthesizer's functionality through the Spiral API, allowing users to create and manipulate sounds using the DSL.

Modular Architecture: Implement a modular architecture similar to Bitwig Studio's device nesting and modulation system. This will allow users to create complex audio processing chains and flexible modulation routings. Expose the architecture through the Spiral API.

Blockchain Integration: Incorporate native blockchain integration for decentralized collaboration, rights management, and royalty tracking. This may involve using a Rust library for interacting with a spe-

cific blockchain platform, such as the Substrate framework for building blockchains in Rust. Expose the blockchain integration through the Spiral API.

Presets and Project Management: Design a system for managing presets, samples, and projects, allowing users to save and load their work seamlessly. This may involve using serialization formats like JSON or YAML. Provide an API for Spiral to interact with the project management system.

Documentation and Community: Create extensive documentation and tutorials for the DAW, including examples and best practices for using the text-based interface and Spiral DSL. Establish a community forum or platform for users to share their creations, ask questions, and collaborate on projects.

By integrating the Spiral language as the DSL, users can leverage Spiral's unique features, such as machine learning capabilities, to create a powerful and expressive environment for audio production and sound design.

make the most precise scale you can find (linear, log, etc) to depict different timeframes to release a rating 1000 version of such daw considering overall team members and months of development

< Creating a precise scale to estimate the timeframe for releasing a rating 1000 version of the text-based DAW is challenging, as it depends on various factors like the expertise of the team members, their familiarity with the required technologies, and the complexity of the desired features.

However, I can provide a rough estimation of the timeframes based on the number of team members and months of development. For simplicity, let's assume that the team consists of experienced developers who are familiar with audio programming, Rust, Spiral, and blockchain technologies.

Team Mem- bers	6 months	12 months	18 months	24 months	30 months	36 months
2	500	700	850	950	975	1000
4	700	900	975	1000	1000	1000
6	850	975	1000	1000	1000	1000
8	900	1000	1000	1000	1000	1000

This table represents a rough estimation of the DAW rating over different timeframes and team sizes. The rating increases with the number of team members and the length of the development period, as more time and resources allow for the implementation of more advanced features and refinements.

Please note that these estimations are highly speculative and may not accurately represent the actual timeframes or ratings of the DAW. Actual development times may vary significantly depending on the specific circumstances and challenges encountered during the project.

describe with academic rigor the previous training that 8 developers need to endure to be able to coordinate development in one year to develop such daw

< To develop a rating 1000 text-based DAW in one year, a team of 8 developers would need a diverse skill set and expertise in various domains. Here's a breakdown of the necessary training and experience the developers should possess, with academic rigor:

Audio Programming and Signal Processing: All developers should have a strong foundation in audio programming and digital signal processing. This can be achieved through formal education in computer science, electrical engineering, or music technology with a focus on audio programming, or through relevant work experience and self-study. Key concepts include digital audio representation, synthesis techniques, audio effects processing, and real-time audio processing.

Programming Languages: The developers should be proficient in Rust, the primary programming language for the DAW, as well as the Spiral language and F#. Proficiency can be achieved through formal education, work experience, or self-study, including online courses and tutorials.

Software Engineering Principles: The team should be well-versed in software engineering principles, including object-oriented programming, design patterns, testing, and version control. These skills are essential for writing efficient, maintainable, and scalable code.

Domain-Specific Languages (DSLs) and Compilers: The developers should have experience with designing and implementing domain-specific languages (DSLs) and compilers, particularly for the integration of the Spiral language as the DSL for the DAW. Formal education or work experience in language design, compiler theory, and implementation is valuable.

MIDI and Audio Plugin Formats: Familiarity with MIDI protocol, as well as audio plugin formats such as VST3 or LV2, is essential for integrating plugin support into the DAW. Experience with relevant libraries (e.g., `vst-rs`, `lv2-rs`) is a plus.

Blockchain Technologies: To incorporate native blockchain integration, developers should have a strong understanding of blockchain technologies, including consensus algorithms, smart contracts, and decentralized applications. Formal education in computer science or related fields with a focus on blockchain, relevant work experience, or self-study is necessary.

User Interface (UI) Design and Terminal I/O: Although the DAW will be text-based, developers should be familiar with user interface design principles and terminal I/O to create a responsive and user-friendly interface. Experience with Rust libraries like `crossterm` or `termion` is beneficial.

Project Management and Collaboration: The team should possess strong project management and collaboration skills to effectively coordinate the development process. Experience with Agile methodologies, such as Scrum, can help ensure smooth project execution.

It's crucial to have a diverse team of developers with complementary skills and expertise to cover all aspects of the DAW development. The team can consist of specialists in audio programming, language design, blockchain technology, and software engineering, as well as generalists who can contribute to multiple areas of the project.

In addition to formal education, developers can expand their skill set through self-study, online courses, and workshops. Attending conferences, participating in open-source projects, and staying up-to-date with the latest developments in their respective fields can also help developers stay competitive and contribute effectively to the DAW project.

list the best 10 books to conceptually master this implementation in markdown format rating 0-1000 with proper columns and remarks

< Here is a list of the top 10 books that can help you conceptually master the implementation of the text-based DAW. The rating is based on relevance, quality, and depth of coverage of the required topics.

Book Title	Author(s)	Rating	Remarks
Designing Audio Effect Plugins in C++	Will Pirkle	950	Covers audio plugin development, signal processing, and C++ programming. Although not Rust-specific, the concepts can be applied to Rust development.
The Audio Programming Book	Richard Boulanger and Victor Lazzarini	900	A comprehensive resource for audio programming, including synthesis, processing, and MIDI.
Programming Rust: Fast, Safe Systems Development	Jim Blandy and Jason Orendorff	900	Comprehensive guide to Rust programming, focusing on safety and performance.
Mastering Blockchain	Imran Bashir	850	Covers fundamental concepts and techniques for developing blockchain applications.
Real-Time Collision Detection	Christer Ericson	800	Provides insights into efficient programming and data structures that can be applied to real-time audio processing.
Computer Music: Synthesis	Charles Dodge and Thomas	800	Explores the principles of

list the best 10 books to conceptually master this implementation soft-skills wise in markdown format rating 0-1000 with proper columns and remarks

< Here is a list of the top 10 books that can help you conceptually master the soft skills needed for the implementation of the text-based DAW. The rating is based on relevance, quality, and depth of coverage of the required topics.

Book Title	Author(s)	Rating	Remarks
The Mythical Man-Month: Essays on Software Engineering	Frederick P. Brooks Jr.	950	Classic book on software project management and team coordination, with valuable lessons for any development project.
Peopleware: Productive Projects and Teams	Tom DeMarco and Timothy Lister	950	Explores the human side of software development, focusing on team dynamics, communication, and productivity.
Clean Code: A Handbook of Agile Software Craftsmanship	Robert C. Martin	900	Provides principles and best practices for writing clean, maintainable code and collaborating effectively with team members.
Soft Skills: The Software Developer's Life Manual	John Sonmez	900	Covers a range of soft skills essential for software developers, including communication, teamwork, and time management.
The Art of Agile Development	James Shore and Shane Warden	850	Provides a practical guide to Agile methodologies, including

if an intricate 180bpm hi-tech psytrance song were to be written using tidalcycles mainly with the assistance of copilot and gpt3, list the main data structures of the library

< TidalCycles (or Tidal) is a Haskell-based domain-specific language (DSL) designed for live coding, pattern-based sequencing, and algorithmic composition. To create a 180 BPM hi-tech psytrance track using TidalCycles, you'll need to understand some of the primary data structures and functions it offers.

Main data structures and functions in TidalCycles:

Pattern: A pattern is the core data structure in TidalCycles. It represents a time-varying function that maps time to events. Patterns can be created using functions like `s`, `n`, `d1`, and others.

Control Patterns: These are patterns that control various parameters of a sound, such as pitch, volume, panning, and more. Control patterns can be created using functions like `up`, `speed`, `pan`, and others.

Mini-notation: TidalCycles uses a mini-notation to represent complex rhythmic structures. It allows you to create patterns using strings like `"bd sn cp*2"` and parse them into more complex patterns.

Transformations: TidalCycles provides a wide range of functions for transforming patterns, such as time-stretching, shifting, reversing, and more. Some examples include `fast`, `slow`, `rev`, and `jux`.

Combinators: TidalCycles offers several combinators that allow you to combine and manipulate patterns in various ways. Combinators include functions like `|+|`, `|*|`, `stack`, and `cat`.

Conditionals: TidalCycles enables you to apply conditional logic to your patterns, allowing you to create dynamic and evolving sequences. Some examples of conditional functions are `whenmod`, `every`, and `while`.

Randomness: TidalCycles provides functions to introduce randomness and variability into your patterns. Functions like `rand`, `irand`,

choose, and shuffle can help create more organic and unpredictable sequences.

to achieve the desired track, list some function combinations that might be important in the structure

To create a 180 BPM hi-tech psytrance track in TidalCycles, you will want to utilize various functions and their combinations to generate intricate rhythms, melodies, and sound manipulations. Here are some function combinations that might be essential in building your track:

Setting BPM:

```
setcps ( 180 / 60 / 4 )
```

Layering multiple patterns using stack:

```
d1 $ stack [ s " bd(3,8) " , s " sn(5,16) " ,
s " hh*4(3,8) " ]
```

Combining patterns with different transformations using jux and rev:

```
d1 $ jux (rev) $ s " bd(3,8) sn "
```

Applying time transformations using fast, slow, and density:

```
d1 $ slow 2 $ s " bd*2 sn*4 " d1 $ fast 4 $ s "
hh*8 cp*4 " d1 $ density 2 $ s " bass*4 arpy*2 "
```

Modulating parameters with control patterns using |+|, |*|, and |>|:

```
d1 $ s " bass(3,8) " |+| n " 0 3 -1 2 " |*| gain
" 0.8 1 0.7 "
```

```
d1 $ s " bass*2 " # speed (choose [ 1 , 1.5 , 0.5 ,
2 ]) d1 $ s " arpy(5,16) " # pan (rand) d1 $ s " sn*4
" # gain (shuffle [ 0.7 , 0.8 , 1 , 0.9 ])
```

Applying conditional transformations with functions like whenmod, every, and while:

```
d1 $ every 4 (fast 2 ) $ s " bd*2 sn*4 " d1 $
whenmod 8 3 ( |+| n " 12 " ) $ s " bass*4 " d1 $
while ( |<| gain " 0.7 " ) (# speed 2 ) $ s " arpy*2
"
```

By combining these functions and experimenting with different patterns and parameters, you can create a complex, hi-tech psytrance track with TidalCycles. Remember that TidalCycles is a live coding environment, so feel free to tweak, modify, and explore these function combinations in real-time to achieve the desired sound.

detail in pt-br the genesis and lore of classical music

< Claro, com prazer!

A música clássica teve origem durante o período conhecido como a Idade Média, por volta do século IX, com o desenvolvimento do canto gregoriano. Esse canto era executado por monges católicos e caracterizava-se por ser monofônico, ou seja, consistia em uma única linha melódica sem acompanhamento instrumental.

Na Renascença, por volta do século XIV ao XVI, a música sofreu uma grande mudança. A polifonia, ou seja, a presença de várias linhas melódicas ao mesmo tempo, tornou-se predominante. Também foi durante a Renascença que a música secular começou a se tornar mais popular.

O período Barroco, do século XVII ao XVIII, foi uma época de muita experimentação e inovação. Durante esse período, surgiu o contraponto, uma técnica que envolve a sobreposição de diferentes linhas melódicas. Compositores barrocos famosos incluem Johann Sebastian Bach e Antonio Vivaldi.

Em seguida veio o período Clássico, que durou do meio do século XVIII ao início do século XIX. Esta era é conhecida pela sua simplicidade e equilíbrio. Wolfgang Amadeus Mozart e Ludwig van Beethoven são dois dos compositores mais notáveis desta época.

A música do período Romântico, que durou do século XIX ao início do século XX, caracterizava-se por sua expressão emocional intensa. Os compositores românticos procuravam expressar sentimentos profundos e muitas vezes usavam a música para contar uma história ou descrever uma cena. Compositores famosos deste período incluem Frederic Chopin e Richard Wagner.

O período Moderno e Contemporâneo começou no final do século XIX e continua até hoje. Durante esse tempo, compositores começaram a experimentar ainda mais, utilizando novas técnicas e ideias. A música deste período pode ser bastante diversificada, variando do impressionismo de Claude Debussy à música atonal de Arnold Schoenberg.

Cada um desses períodos contribuiu para a rica história da música clássica, influenciando compositores e músicos até os dias de hoje. Esta é uma visão geral do gênese e da tradição da música clássica, e há muito mais para explorar em cada período.

from everything available let's say on spotify, which options could be intricate ones overall? keep outputting in pt-br

< Claro, aqui estão algumas recomendações de peças de música clássica mais complexas e intrincadas disponíveis no Spotify. Essas composições são conhecidas por sua complexidade estrutural, uso avançado de harmonia e contraponto, e/ou sua exigência de alta habilidade técnica.

1. Johann Sebastian Bach - "A Oferta Musical" (Musical Offering, BWV 1079): Este é um conjunto de cânones e fugas baseadas em um único tema musical, um excelente exemplo do uso de Bach de contraponto complexo.
2. Ludwig van Beethoven - "Sonata para Piano nº 29, Op. 106" (também conhecida como "Hammerklavier"): Esta é uma das sonatas para piano mais técnicas e emocionalmente profundas de Beethoven.
3. Igor Stravinsky - "A Sagração da Primavera" (The Rite of Spring): Esta peça de balé é famosa por sua complexidade rítmica e uso

dissonante da harmonia.

4. Arnold Schoenberg - “Pierrot Lunaire, Op. 21”: Esta peça usa a técnica dodecafônica de Schoenberg e é um exemplo primordial do expressionismo na música.
5. Béla Bartók - “Concerto para Orquestra”: Este é um exemplo de como Bartók integrou elementos do folclore húngaro com harmonias modernas.
6. Olivier Messiaen - “Quarteto para o Fim do Tempo” (Quatuor pour la fin du temps): Messiaen compôs essa peça enquanto estava em um campo de prisioneiros durante a Segunda Guerra Mundial. Ela é conhecida por seu uso inovador do ritmo e da cor tonal.

Essas são apenas algumas recomendações para começar, mas há muitas outras peças complexas e emocionantes para explorar no mundo da música clássica!

if these were to be used as foundation for rnn architectures that morph them into 200+ bpm psycore/darkpsy album collections, how such schematics could be organized?

< Usar Redes Neurais Recorrentes (RNNs) para transformar peças de música clássica em coleções de álbuns de psycore/darkpsy a 200+ BPM é um projeto intrigante e desafiador! Aqui está uma estrutura geral de como tal projeto poderia ser organizado:

1. **Pré-processamento dos dados:** Comece convertendo suas músicas clássicas em um formato que uma RNN possa entender. Você pode usar a representação MIDI dos arquivos, que captura a nota, a duração e a intensidade. As peças de psycore/darkpsy também precisam ser convertidas para o mesmo formato.
2. **Treinamento da RNN:** Alimente os dados de música clássica para a RNN e treine a rede para aprender os padrões e estruturas dessa música.
3. **Transferência de Aprendizado:** Após o treinamento inicial na música clássica, alimente a rede com os dados do psycore/darkpsy.

A ideia é que a rede utilize os padrões aprendidos na música clássica para criar novas composições no estilo psycore/darkpsy.

4. **Geração de Música:** Use a RNN treinada para gerar novas músicas. Você pode controlar a saída ajustando os parâmetros de temperatura (que controla a aleatoriedade da saída) e fornecendo um “prime” inicial para a geração (uma sequência de notas para iniciar a composição).
5. **Pós-processamento:** A saída da RNN provavelmente precisará de algum ajuste. Isso pode incluir ajustar o tempo para 200+ BPM, equalizar o volume, e adicionar efeitos de som.
6. **Avaliação:** Finalmente, você precisará de uma maneira de avaliar a qualidade da música gerada. Isso pode ser feito ouvindo a música e fazendo ajustes com base em suas preferências pessoais, ou pode envolver o uso de uma rede neural adicional treinada para avaliar a qualidade da música.

Lembre-se de que isso é apenas uma visão geral e que a implementação real dependerá de vários fatores, incluindo os recursos disponíveis, a quantidade e qualidade dos dados de treinamento e as especificidades do gênero musical que você está tentando criar. É também um campo em rápido desenvolvimento, então novas técnicas e abordagens estão constantemente surgindo.

how the overall architecture could be seamlessly implemented considering the input the classical and psytrance samples in raw and preprocessed form, the output being the albums using a static dsl in text format (haskell/elixir) compatible with technologies like tidalcycles/Tidal or sonic pi (which other random samples would also be provided as input, maybe even the classical samples themselves converted - if possible)

< Uma maneira de fazer isso seria criar um sistema que combina várias técnicas de aprendizado de máquina e programação funcional para processar e transformar as amostras musicais. Aqui está uma visão geral de como esse sistema poderia funcionar:

1. **Pré-processamento das Amostras de Música:** Você começaria convertendo as amostras de música clássica e psytrance em um formato que pode ser usado para treinar uma rede neural. Isso pode envolver a extração de recursos como altura, duração, intensidade, e talvez até características mais complexas como timbre e textura. As amostras de música clássica também poderiam ser transformadas em uma forma que pode ser usada com TidalCycles ou Sonic Pi.
2. **Treinamento de uma Rede Neural:** Uma vez que as amostras estejam pré-processadas, você poderia usar uma rede neural recorrente (RNN) para aprender os padrões na música clássica e psytrance.
3. **Transformação da Música:** Depois que a RNN estiver treinada, ela poderia ser usada para gerar novas sequências de música com base nas amostras de música clássica. Essas sequências poderiam então ser transformadas em um formato compatível com TidalCycles ou Sonic Pi.
4. **Criação do DSL:** Usando a saída da RNN, você poderia criar um DSL (Domain Specific Language) em Haskell ou Elixir que

descreve a nova música. Isso permitiria que você manipulasse a música com uma sintaxe de alto nível e gerasse código para TidalCycles ou Sonic Pi.

5. **Pós-Processamento e Avaliação:** Finalmente, você poderia usar o DSL para gerar a música final, e então avaliar a qualidade da música gerada, fazendo ajustes conforme necessário.

Este é um projeto complexo e altamente técnico que exigiria conhecimentos em várias áreas, incluindo aprendizado de máquina, processamento de sinal de áudio, programação funcional e síntese de música. No entanto, a ideia de transformar música clássica em psytrance através de técnicas de aprendizado de máquina e DSLs de música é fascinante e poderia levar a alguns resultados muito interessantes.

could elixir's axon and nx be used through gleam lang for such implementation? the output could be sonic pi files, tidalcycles or even a new daw compatible format. note that such tracks for the albums could normally have hundreds of layers if made manually in a daw. maybe the available text data available for training wouldn't nearly match the complexity needed for the output. the output files might cross the megabyte mark maybe? keep brushing out

< Usar Axon e Nx em conjunto com a linguagem Gleam parece uma abordagem interessante e potencialmente viável para o projeto. Axon é uma biblioteca para aprendizado profundo em Elixir que utiliza o Nx (Numex), uma biblioteca numérica multi-dimensional também em Elixir, que é altamente otimizada para operações de aprendizado de máquina. Isso torna essas ferramentas bem adaptadas para o processamento e análise de dados musicais complexos.

Gleam é uma linguagem de programação estática e fortemente tipada que roda na BEAM, a máquina virtual Erlang, a mesma que Elixir utiliza. Ainda que Gleam seja uma linguagem nova, ela poderia permitir

uma programação funcional mais robusta e previsível devido ao seu sistema de tipos. No entanto, devido à sua juventude, o suporte para bibliotecas específicas como Axon e Nx não é garantido. Você precisaria verificar a compatibilidade e interoperabilidade entre Gleam e essas bibliotecas.

Quanto à saída em formatos compatíveis com Sonic Pi, TidalCycles ou um novo formato de DAW, seria uma questão de pós-processar e formatar a saída da RNN e do DSL para esses formatos. Isso pode envolver a conversão dos dados em formato MIDI, OSC, ou algum outro formato suportado pelo sistema de destino.

Sobre a complexidade da saída: é verdade que a música gerada por aprendizado de máquina pode não alcançar a complexidade de uma peça que seria criada manualmente em um DAW, especialmente se estivermos falando de centenas de camadas. No entanto, a quantidade de texto de saída pode não ser um problema, já que os dados gerados seriam principalmente numéricos e poderiam ser compactados eficientemente. Além disso, é possível implementar uma arquitetura de RNN mais complexa, como uma rede neural convolucional de longo prazo de memória (LSTM), que poderia lidar com uma maior complexidade na música.

Essa é uma visão ampla do projeto. Cada parte dessa implementação apresentaria seus próprios desafios e a viabilidade dependeria muito das especificidades do projeto, dos dados disponíveis para treinamento e das habilidades e conhecimentos do implementador. É um projeto ambicioso, mas potencialmente muito interessante.

to cross the complexity boundaries for 200+ bpm at the singularity level, detail how such nn architectures could function? (lstm, grus, transformers, etc)

< Para lidar com a complexidade de gerar música com uma alta taxa de BPM e atingir um nível de “singularidade”, onde a música gerada seja indistinguível daquela criada por um humano, você precisaria explorar arquiteturas de redes neurais mais avançadas. Aqui estão alguns exemplos e como eles poderiam ser usados:

1. **Long Short-Term Memory (LSTM):** LSTM é uma extensão da RNN que pode aprender dependências de longo prazo. Em uma música com um alto BPM, as relações entre as notas

podem mudar rapidamente, tornando as LSTMs uma escolha potencialmente boa. As LSTMs poderiam ser usadas para modelar a estrutura temporal da música, aprender padrões de longo prazo e gerar novas sequências de notas.

2. **Gated Recurrent Units (GRUs):** As GRUs são similares às LSTMs, mas têm uma estrutura mais simples e menos parâmetros, o que pode torná-las mais eficientes para treinamento. Elas também poderiam ser usadas para modelar a estrutura temporal da música e aprender padrões de longo prazo.
3. **Transformadores:** Os transformadores utilizam a auto-atenção para modelar relações entre todas as partes da sequência, independentemente de sua distância. Isso os torna capazes de aprender dependências de longo prazo e estruturas complexas. Eles têm sido usados com sucesso em uma variedade de tarefas de processamento de linguagem natural e poderiam ser aplicados à geração de música para modelar a estrutura e a evolução da música ao longo do tempo.
4. **Redes Neurais Convolucionais (CNNs):** Embora geralmente sejam usadas para processamento de imagens, as CNNs também podem ser usadas para processar áudio. Elas podem ser úteis para extrair características locais e padrões em diferentes escalas de tempo.

Cada uma dessas arquiteturas tem suas vantagens e desvantagens e pode ser mais adequada para diferentes aspectos do problema. Em última análise, pode ser benéfico experimentar várias arquiteturas e combinações de arquiteturas para ver qual delas produz os melhores resultados para sua tarefa específica.

Além disso, dado o alto nível de complexidade desejado para a saída, pode ser útil investigar abordagens mais avançadas de aprendizado de máquina, como treinamento de redes adversárias generativas (GANs) ou autoencoders variantes (VAEs), que foram usados para gerar resultados impressionantes em tarefas complexas de geração.

Para melhorar ainda mais a qualidade da música gerada, pode ser útil incorporar algum tipo de feedback humano no processo de treinamento, seja através de aprendizado por reforço, onde um agente é treinado para maximizar uma recompensa baseada em feedback humano, ou através de um sistema de classificação humana, onde a música gerada é classificada por humanos e essas classificações são usadas para treinar a rede.

ok, besides the ones i mentioned, you added cnns, gans and vaes. add even more the viable architectures that could be mingled in the whole pipeline making a final list sorting by importance (try to assign a score 0-1000 from each architecture in how its finetuning mastery might affect the final output quality). try to make 10 architectures/techniques

< Vou listar 10 arquiteturas/técnicas viáveis para a tarefa em questão e atribuir uma pontuação estimada de 0 a 1000 para cada uma, com base em como a maestria em seu ajuste fino poderia afetar a qualidade final do resultado. Observe que essas pontuações são bastante especulativas e dependem muito das especificidades do problema e da implementação.

1. **Redes Neurais Recorrentes (RNNs):** Pontuação 700. As RNNs são fundamentais para a modelagem de sequências temporais e, portanto, seriam uma parte essencial da pipeline. Afinar bem as RNNs poderia ter um impacto significativo na qualidade da música gerada.
2. **Long Short-Term Memory (LSTM):** Pontuação 800. LSTMs são uma extensão das RNNs que podem capturar dependências de longo prazo, tornando-as particularmente úteis para a modelagem de música. O ajuste fino das LSTMs poderia ter um grande impacto na qualidade do resultado.
3. **Gated Recurrent Units (GRUs):** Pontuação 800. As GRUs são semelhantes às LSTMs, mas com uma estrutura mais simples. O ajuste fino das GRUs também poderia ter um grande impacto na qualidade do resultado.
4. **Transformadores:** Pontuação 850. Os transformadores são poderosos na modelagem de dependências complexas e de longo prazo, e seu ajuste fino poderia ter um impacto significativo na qualidade do resultado.
5. **Redes Neurais Convolucionais (CNNs):** Pontuação 600. Embora as CNNs possam ser úteis para extrair características

locais da música, elas podem ser menos cruciais para a tarefa do que as arquiteturas focadas em sequências temporais.

6. **Redes Adversárias Generativas (GANs):** Pontuação 800. As GANs podem gerar resultados impressionantes em tarefas de geração complexas, e o ajuste fino das GANs poderia ter um grande impacto na qualidade do resultado.
7. **Autoencoders Variantes (VAEs):** Pontuação 750. Os VAEs são úteis para aprender uma representação latente da música, e seu ajuste fino poderia impactar significativamente a qualidade do resultado.
8. **Autoencoders Temporais Recorrentes (RTAEs):** Pontuação 750. RTAEs são uma combinação de RNNs e autoencoders que podem ser úteis para aprender uma representação compacta de sequências de música. O ajuste fino dos RTAEs poderia ter um impacto significativo na qualidade do resultado.
9. **Seq2Seq com atenção:** Pontuação 850. O modelo Seq2Seq com mecanismo de atenção é capaz de mapear sequências de entrada para sequências de saída, focando diferentes partes da entrada em diferentes momentos. Ajustar bem este modelo poderia ter um impacto substancial na qualidade da saída.
10. **Capsule Networks (CapsNets):** Pontuação 650. CapsNets são uma alternativa às CNNs que podem capturar relações hierárquicas entre partes da entrada, embora não sejam comumente usadas em música, a maestria em seu ajuste fino poderia potencialmente melhorar a qualidade da saída.

if these architectures would interact recursively graph-wise in the beam using gleam, how the otp framework and actor model could be explained?

< O BEAM (Bogdan/Björn's Erlang Abstract Machine) é a máquina virtual na qual Erlang e outras linguagens como Elixir e Gleam são executadas. Ela fornece suporte para o modelo de ator e o OTP (Open Telecom Platform), um conjunto de módulos e padrões para desenvolvimento em Erlang que também é amplamente utilizado em Elixir.

O **modelo de ator** é um modelo de programação para sistemas concorrentes e distribuídos. Neste modelo, “atores” são as unidades fundamentais de execução e cada um deles é responsável por uma tarefa específica. Atores se comunicam entre si exclusivamente por meio de troca de mensagens, o que torna este modelo particularmente adequado para sistemas distribuídos onde a comunicação pode ser irregular ou demorada. No contexto de seu projeto, cada ator pode ser responsável por uma parte diferente do pipeline, como pré-processamento, treinamento de uma determinada arquitetura de rede neural, pós-processamento, etc.

O **OTP** é uma coleção de módulos, padrões e princípios para a construção de aplicações robustas e tolerantes a falhas. Ele fornece várias abstrações úteis, como servidores genéricos (um modelo para processos de longa duração que mantêm estado), máquinas de estado genéricas (um modelo para processos com um estado interno que muda em resposta a eventos), supervisores (processos que monitoram outros processos e os reiniciam se falharem), e aplicações (um pacote de código e recursos com uma árvore de supervisão).

No contexto do seu projeto, você poderia usar o OTP para estruturar a interação entre diferentes partes do sistema. Por exemplo, você pode ter um servidor genérico para cada etapa do pipeline (pré-processamento, treinamento da rede neural, pós-processamento, etc.) e um supervisor para monitorar esses servidores e reiniciá-los em caso de falhas. Isso tornaria seu sistema mais robusto e capaz de lidar com erros e falhas sem interromper todo o pipeline.

Gleam, sendo uma linguagem de programação funcional estática na BEAM, pode se beneficiar dessas características de concorrência e tolerância a falhas. Embora Gleam possa não ter suporte direto para todos os módulos e comportamentos OTP, é possível interagir com esses recursos por meio de interop com Erlang e Elixir. Como Gleam é uma linguagem de tipo estático, também pode ajudar a evitar certos tipos de erros em tempo de execução e tornar o código mais fácil de refatorar e manter.

try to provide an initial sample using gleam’s discriminated union relationships and axon

< Na data do meu treinamento (setembro de 2021), Gleam ainda está em estágios iniciais de desenvolvimento e tem interoperabilidade com Elixir e Erlang, mas as bibliotecas em Elixir como Axon não têm

tipagem específica para Gleam. Além disso, Gleam não suporta diretamente manipulações complexas de matrizes ou tensores, que são necessárias para a maioria dos cálculos de aprendizado profundo.

No entanto, é possível representar tipos mais complexos e suas relações na Gleam usando uniões discriminadas. Aqui está um exemplo muito simplificado de como você poderia representar um neurônio em uma rede neural:

```
= Sigmoid | ReLU | Tanh type Neuron = { weights:
List(Float) , bias: Float , activation: ActivationFunction
}
```

Neste exemplo, `ActivationFunction` é uma união discriminada que representa as diferentes funções de ativação que um neurônio pode ter. O `Neuron` é um tipo de registro que representa um neurônio em uma rede neural, incluindo seus pesos, bias e a função de ativação que usa.

Note que este é um exemplo extremamente simplificado e uma implementação real de redes neurais em Gleam provavelmente exigiria uma abstração muito mais complexa.

Para a interação com Axon, você pode precisar usar funções de interoperabilidade de Gleam para chamar código Elixir. No entanto, a integração entre Gleam e Axon pode ser um desafio, já que Gleam não foi projetado com manipulações complexas de matrizes ou tensores em mente, e Axon não foi projetado com a tipagem estática de Gleam em mente.

provide an initial sample for a core part using axon and elixir's multi clause functions only

< Axon é uma biblioteca para Deep Learning escrita em Elixir. Aqui está um exemplo simples de como definir e treinar uma rede neural usando Axon. Neste exemplo, vamos considerar a tarefa de classificação binária.

Primeiro, precisamos definir a arquitetura da nossa rede:

```
defmodule ExampleNN do use Axon.Model def init(_params)
do Axon.input({nil, 784}) |> Axon.dense(128, activation:
:relu) |> Axon.dense(64, activation: :relu) |>
Axon.dense(1, activation: :sigmoid) end end
```

Aqui definimos uma rede com duas camadas ocultas com ativação ReLU e uma camada de saída com ativação sigmóide.

Depois, podemos definir funções para treinar a nossa rede:

```
def train(network, {inputs, targets}) do {params, _}
= Nx.Defn.grad(network, &Axon.Losses.binary_crossentropy(
targets, &1)) {inputs, targets, params} end def loop(
network, data) do for _ <- 1..100 do Enum.reduce(
data, network, &train/2) end end
```

Neste código, `train/2` recebe a rede e um conjunto de entradas e alvos, e calcula o gradiente da função de perda (neste caso, a entropia cruzada binária) em relação aos parâmetros da rede. `loop/2` executa o treinamento por 100 épocas.

Observe que este é um exemplo muito simplificado e não inclui várias partes importantes do treinamento de uma rede neural na prática, como a divisão de dados em lotes, a utilização de um otimizador para atualizar os parâmetros da rede, a validação do modelo em um conjunto de dados de teste, etc.

Além disso, em uma situação real, você provavelmente desejaria usar um conjunto de dados maior e mais complexo e uma arquitetura de rede mais complexa, possivelmente incluindo outras características como regularização, dropout, camadas de convolução (para dados de imagem), camadas recorrentes (para dados de sequência), etc.

could rust be a better choice for such project? compare both approaches underlining ecosys- tem differences

< Elixir e Rust são duas linguagens de programação muito diferentes e cada uma tem suas próprias vantagens e desvantagens. A escolha entre elas dependerá em grande parte das necessidades específicas do seu projeto. Aqui estão alguns pontos a serem considerados:

Elixir:

1. **Concorrência e Paralelismo:** Elixir, sendo construído sobre a máquina virtual de Erlang (BEAM), oferece excelentes recursos de concorrência e paralelismo com um modelo de ator muito robusto. Isso é ideal para aplicações em tempo real, sistemas distribuídos e onde a tolerância a falhas é necessária.

2. **Hot Code Swapping:** Elixir suporta a troca de código em tempo real, o que pode ser útil para a implantação de atualizações sem tempo de inatividade.
3. **Ecosistema:** Elixir tem um ecossistema forte com muitas bibliotecas para várias tarefas, incluindo web e desenvolvimento de servidor. Axon é uma biblioteca Elixir para aprendizado profundo, o que significa que você pode ficar dentro do ecossistema de Elixir para as necessidades de aprendizado de máquina.

Rust:

1. **Performance:** Rust é conhecido por sua eficiência e desempenho. Pode oferecer um desempenho comparável ao do C++, o que é ideal para computação pesada, como treinamento de modelos de aprendizado de máquina.
2. **Gerenciamento de Memória:** Rust tem um sistema de gerenciamento de memória único e seguro que elimina muitos bugs comuns em outros idiomas.
3. **Ecosistema:** Rust também tem um ecossistema em rápido crescimento com muitas bibliotecas para várias tarefas. Para aprendizado profundo, existem bibliotecas como Leaf, AutumnAI, juice, etc.

Comparativamente, se você está planejando criar um sistema com foco em desempenho, precisão de tipo e segurança de memória, Rust pode ser uma escolha melhor. Se você está mais focado em concorrência, tolerância a falhas e um sistema distribuído com troca de código em tempo real, Elixir seria mais adequado.

No entanto, ambos não são tradicionalmente usados em tarefas de aprendizado profundo. Python, com bibliotecas como TensorFlow e PyTorch, domina o espaço de aprendizado profundo, principalmente devido ao seu ecossistema rico e suporte à comunidade. Portanto, dependendo do projeto, pode ser mais produtivo usar uma linguagem e um ecossistema que já tenha um amplo suporte para as tarefas que você está tentando realizar.

if i wanted static typing, which ecosystem could yield better results? (classical music album list as seed for psy album list) also, which architecture could be the hardest but most rewarding to implement using discriminated union relationships?

< Se a tipagem estática é uma preocupação fundamental, Rust se destacaria. Rust é uma linguagem com tipagem estática, o que oferece benefícios como detecção precoce de erros, melhor documentação de funções e variáveis (uma vez que seus tipos são declarados explicitamente), e otimizações de tempo de compilação.

Além disso, Rust tem suporte para “enums”, que são um pouco semelhantes às uniões discriminadas que você mencionou. Rust enums permitem criar um tipo que pode ser uma de várias “variantes”, cada uma possivelmente com seus próprios dados associados. Este recurso pode ser muito útil para modelar estruturas de dados complexas em um contexto de aprendizado de máquina.

Dito isso, é importante ressaltar que as redes neurais geralmente envolvem muitas operações de matriz que podem não ser bem representadas por uniões discriminadas ou enums. Em vez disso, essas operações são geralmente manipuladas por bibliotecas de matemática otimizadas e geralmente envolvem tipos de dados específicos para matrizes e tensores.

Quanto à arquitetura que poderia ser mais difícil, mas mais recompensadora para implementar usando uniões discriminadas, provavelmente seria algo como um modelo Transformer. Os Transformers envolvem uma complexidade significativa, com várias partes diferentes (camadas de atenção, redes feed-forward, camadas de normalização, etc.) que podem ser modeladas como variantes em uma união discriminada. No entanto, os Transformers também provaram ser muito eficazes para uma ampla gama de tarefas de aprendizado de máquina, por isso, apesar da complexidade de sua implementação, eles podem ser bastante recompensadores em termos de desempenho do modelo.

detail the most intricate enum of the transformer model for such albums task (omit the type definition of the adjacent types and onward for later, but if there's recursion try to leave a comment)

< Vamos considerar uma simplificação de alto nível da arquitetura Transformer e como ela poderia ser representada em Rust usando enums. Isso é apenas uma representação ilustrativa e uma implementação real seria muito mais complexa.

```
enum Layer {      // Other layer types might be added
here      Transformer ( TransformerLayer ) } enum Trans-
formerLayer {      EncoderLayer ( EncoderLayer ) ,
DecoderLayer ( DecoderLayer ) , } struct EncoderLayer
{      self_attn: MultiHeadAttention,      feed_forward
: PositionWiseFeedForward,      // SubLayerConnection
for each of the above      // ... } struct Decoder-
Layer {      self_attn: MultiHeadAttention,      cross-
attn: MultiHeadAttention,      feed_forward: PositionWiseFeedForward,
// SubLayerConnection for each of the above      /
/ ... } struct MultiHeadAttention {      // Q, K, V
linear layers, output linear layer,      // and attention
mechanism (like ScaledDotProductAttention)      // ...
} struct PositionWiseFeedForward {      // Two
linear layers and an activation function in between
// ... } // Other struct definitions...
```

Neste exemplo, um `Layer` é um enum que pode representar vários tipos diferentes de camadas. Neste caso, estamos apenas considerando `TransformerLayer`, que pode ser ou uma `EncoderLayer` ou uma `DecoderLayer`.

`EncoderLayer` e `DecoderLayer` são structs que contêm as partes constituintes das camadas do codificador e decodificador do Transformer, respectivamente. Ambas contêm um `MultiHeadAttention` e um `PositionWiseFeedForward`, e o `DecoderLayer` contém um `MultiHeadAttention` adicional para a atenção cruzada entre as saídas do codificador e as entradas do decodificador.

Novamente, isso é uma simplificação e uma implementação real envolveria muitos outros detalhes e complexidades. Além disso, o próprio

Rust não é tradicionalmente usado para implementar modelos de aprendizado de máquina devido à falta de bibliotecas de aprendizado profundo maduras, comparado a linguagens como Python, embora bibliotecas como `tch-rs` (uma interface para PyTorch) estejam ajudando a mudar isso.

detail the long term worth of using technologies like urbit and its languages to build such albums project

< Urbit é uma rede ponto a ponto descentralizada e um sistema operacional virtual construído do zero. Ele tem suas próprias linguagens de programação (Hoon para o desenvolvimento do sistema e Nock como uma linguagem de máquina virtual), e seu próprio protocolo de rede (Ames).

Há alguns aspectos em que o uso de uma tecnologia como Urbit poderia potencialmente ser benéfico para um projeto de criação de álbuns de música:

1. **Descentralização:** Urbit é construído para ser uma plataforma completamente descentralizada e autônoma. Isso poderia permitir a criação de um sistema de música verdadeiramente peer-to-peer, onde cada nó na rede pode contribuir com suas próprias criações musicais e ajudar a distribuir as criações dos outros.
2. **Persistência:** Cada nó Urbit tem um “ship” que possui um ID único e persistente. Isso poderia ser usado para rastrear a propriedade e a proveniência das criações musicais ao longo do tempo.
3. **Privacidade e Segurança:** Urbit é construído com um forte foco na segurança e privacidade dos dados do usuário. Isso pode ser importante se os usuários estão compartilhando suas próprias criações musicais e querem ter controle sobre quem pode acessá-las.

No entanto, também há desafios significativos em usar Urbit para um projeto como este:

1. **Maturidade da Tecnologia:** Na data do meu último treinamento em setembro de 2021, Urbit ainda é uma tecnologia muito

nova e não testada. Sua comunidade e ecossistema de desenvolvimento são muito menores do que os de outras linguagens de programação e plataformas de desenvolvimento.

2. **Complexidade de Aprendizado:** Hoon, a linguagem de programação de Urbit, tem uma sintaxe e semântica muito diferentes das linguagens de programação mais tradicionais. Isso pode tornar a curva de aprendizado íngreme.
3. **Lack of Machine Learning Libraries:** Urbit e Hoon não têm bibliotecas de aprendizado de máquina comparáveis às disponíveis em Python, Rust ou outras linguagens de programação mais estabelecidas. Implementar essas bibliotecas do zero seria um projeto enorme por si só.

Portanto, enquanto Urbit oferece algumas possibilidades interessantes para um projeto de criação de álbuns de música descentralizado, também há desafios significativos a serem superados. Se a descentralização e a propriedade autônoma dos dados são prioridades para o projeto, então Urbit pode ser uma opção a considerar. No entanto, se o foco principal do projeto é o aprendizado de máquina e a geração de música, pode ser mais produtivo usar uma linguagem de programação e um ecossistema com suporte mais estabelecido para essas tarefas.

if these benefits would be solved using the near protocol, describe how the rust and gleam/elixir on such project could be distributed considering strengths and needs

< O Near Protocol é uma plataforma de contrato inteligente escalável e orientada para o desenvolvedor, que também é descentralizada e usa a linguagem Rust para o desenvolvimento de contrato inteligente. Isso permite que você aproveite os benefícios da descentralização e também as vantagens da linguagem Rust, como segurança, eficiência e desempenho.

Em um projeto de criação de álbuns musicais descentralizados, você pode pensar em dividir as responsabilidades entre Rust e Elixir (ou Gleam) de acordo com suas respectivas forças:

Rust com Near Protocol:

Rust pode ser usado para escrever contratos inteligentes que são implantados na rede Near. Estes contratos inteligentes poderiam lidar com aspectos como:

1. **Propriedade e Proveniência:** Cada álbum ou faixa de música poderia ser representado como um token não fungível (NFT) na rede Near, com a propriedade do token representando a propriedade do álbum ou faixa. Isso permitiria que a propriedade e a proveniência do álbum ou faixa fossem rastreadas de forma segura e transparente.
2. **Transações:** As transações para comprar, vender ou transferir álbuns ou faixas poderiam ser processadas através de contratos inteligentes na rede Near. Isso garantiria que todas as transações sejam seguras, transparentes e irreversíveis.
3. **Gerenciamento de Direitos:** Você poderia criar contratos inteligentes para gerenciar direitos e royalties para cada álbum ou faixa, assegurando que os criadores sejam justamente compensados por seu trabalho.

Elixir ou Gleam:

Enquanto isso, Elixir (ou Gleam, se você quiser um sistema de tipo mais forte) pode ser usado para criar o backend e o frontend do seu aplicativo, gerenciando aspectos como:

1. **Interfacing com a rede Near:** Você pode usar Elixir para se comunicar com a rede Near, enviando transações, lendo dados de contrato inteligente, etc. Isso poderia ser feito usando uma biblioteca Near API para Elixir, se disponível, ou por meio de solicitações HTTP diretas à API JSON-RPC do Near.
2. **Aplicação de Aprendizado de Máquina:** Elixir poderia ser usado para implementar qualquer funcionalidade de aprendizado de máquina necessária para o projeto, por exemplo, para gerar novas faixas ou álbuns com base nos dados de entrada. Isso poderia ser feito usando bibliotecas como Axon, ou chamando para bibliotecas de aprendizado de máquina escritas em outras linguagens via Erlang NIFs ou portas.
3. **Interface do Usuário:** Elixir (ou Gleam) pode ser usado para desenvolver a interface do usuário do aplicativo, permitindo aos usuários navegar pelos álbuns disponíveis, comprar ou vender álbuns, ouvir faixas, etc.

Desta forma, você pode aproveitar as vantagens de ambas as linguagens - Rust para a segurança e eficiência dos contratos inteligentes, e Elixir para a concorrência e a facilidade de desenvolvimento do backend e frontend do aplicativo.

detail the jurisprudence of the implementation of a smart contract made to replace how labels work today

< O uso de contratos inteligentes na indústria da música tem o potencial de revolucionar o modelo de negócio atual, removendo intermediários e oferecendo maior transparência e eficiência na gestão de direitos e pagamentos. Porém, existem várias questões jurídicas e regulatórias que precisam ser consideradas.

1. **Direitos Autorais:** É fundamental garantir que o uso de contratos inteligentes na gestão de direitos musicais esteja em conformidade com as leis de direitos autorais existentes. Isso inclui garantir que todos os detentores de direitos relevantes tenham dado permissão para a distribuição de sua música e que os pagamentos de royalties sejam feitos de acordo com os acordos de direitos autorais relevantes.
2. **Contratos Legais:** Contratos inteligentes podem ser considerados contratos legais em muitas jurisdições, o que significa que eles devem cumprir certos requisitos legais para serem considerados válidos e aplicáveis. Isso pode incluir a necessidade de identificar claramente as partes do contrato, a existência de um acordo mútuo e a presença de uma consideração (ou seja, algo de valor sendo trocado).
3. **Regulamentação de Valores Mobiliários:** Se você está planejando usar tokens ou criptomoedas como parte do seu modelo de negócio, é importante estar ciente das leis e regulamentos de valores mobiliários. Em muitos casos, tokens ou criptomoedas podem ser considerados valores mobiliários, e a oferta e venda de tais tokens devem cumprir as leis de valores mobiliários.
4. **Privacidade e Proteção de Dados:** Contratos inteligentes e blockchains são muitas vezes transparentes e imutáveis, o que pode levantar questões de privacidade e proteção de dados. É

importante garantir que qualquer dado pessoal coletado ou processado como parte do seu negócio esteja em conformidade com as leis de privacidade e proteção de dados relevantes.

5. **Disputas e Resolução de Conflitos:** Por último, é importante considerar como as disputas ou conflitos serão resolvidos. Contratos inteligentes são codificados e executados automaticamente, o que pode tornar difícil resolver disputas ou erros. É importante ter um mecanismo claro para a resolução de conflitos e a possibilidade de corrigir erros ou fraudes.

Essas são apenas algumas das questões jurídicas e regulatórias a serem consideradas. É altamente recomendável que você busque aconselhamento jurídico antes de prosseguir com a implementação de um contrato inteligente na indústria da música.

Por último, vale ressaltar que a legislação e a jurisprudência podem variar significativamente de uma jurisdição para outra, e as informações fornecidas aqui são apenas uma visão geral. É importante procurar aconselhamento jurídico específico para a jurisdição em que você pretende operar.

detail the skills needed for a software team to implement such project as a whole (near protocol, rust, axon, gleam, elixir, tidalcycles, sonic pi, ipfs, ffmpeg for video, etc) with having the intent of also replacing apps like spotify and youtube (the instrumental part) instead of only labels

< A implementação de um projeto com tal escopo e complexidade exigirá um conjunto bastante diversificado de habilidades e experiência em diversas tecnologias. Aqui está um resumo das principais competências que uma equipe de software precisaria:

1. **Blockchain e Smart Contracts:** A equipe precisará de desenvolvedores experientes em blockchain, especialmente aqueles familiarizados com o protocolo Near e a linguagem Rust para a escrita de contratos inteligentes. Esses profissionais precisarão

entender como projetar, implementar e otimizar contratos inteligentes seguros e eficientes.

2. **Machine Learning:** A criação de novas músicas ou álbuns exigirá profissionais familiarizados com técnicas de aprendizado de máquina, especialmente aquelas relacionadas ao processamento de áudio. Eles precisarão ser proficientes em bibliotecas de aprendizado de máquina, como Axon, e na linguagem Elixir ou Gleam.
3. **Programação Funcional e Concorrente:** Elixir e Gleam são linguagens funcionais e concorrentes, por isso a equipe precisará de desenvolvedores que estejam confortáveis com esses conceitos. Isso inclui o entendimento do OTP (Open Telecom Platform) do Elixir e do modelo de ator para gerenciar a concorrência.
4. **Audio Programming:** A experiência com TidalCycles e Sonic Pi será necessária para a geração e manipulação de músicas. Isso exigirá um entendimento dos conceitos de programação de áudio, bem como a sintaxe e semântica específicas dessas linguagens de programação de áudio.
5. **Storage e IPFS:** Para criar uma plataforma descentralizada, os desenvolvedores precisarão entender tecnologias de armazenamento distribuído como IPFS. Isso envolverá a compreensão de como distribuir e recuperar arquivos em uma rede descentralizada, além de garantir a segurança e a privacidade dos dados.
6. **Processamento de Vídeo:** Se o projeto também visar substituir partes do YouTube, será necessário algum conhecimento de processamento de vídeo. FFmpeg é uma ferramenta poderosa para isso, portanto, a experiência na sua utilização seria extremamente valiosa.
7. **UI/UX Design:** Para criar uma plataforma que os usuários queiram usar, será necessária uma boa experiência de design de interface do usuário e experiência do usuário. Isso envolverá a compreensão de como projetar interfaces atraentes e intuitivas, além de realizar testes de usuário para otimizar a experiência do usuário.
8. **Segurança:** Como o projeto envolverá transações financeiras e armazenamento de dados de usuário, a equipe precisará ter forte conhecimento em segurança de TI. Isso pode envolver a garantia

de segurança do contrato inteligente, a segurança da rede, a segurança dos dados e a conformidade com as regulamentações de privacidade de dados.

9. **Jurisprudência:** Dada a natureza inovadora do projeto, seria útil ter alguém com conhecimento em direito da tecnologia, especialmente no que diz respeito aos direitos autorais, à privacidade e aos contratos inteligentes.

Essas são apenas algumas das principais competências que serão necessárias. O desenvolvimento de um sistema tão complexo e inovador será um grande desafio, mas também uma oportunidade para criar algo verdadeiramente único.

break down mathematically a spreadsheet of possible revenue paths considering one album unit as example using the hypothetical smart contract to distribute it independently

< Para facilitar a discussão, vamos considerar que um artista vai lançar um álbum como um NFT único e vendê-lo diretamente para seus fãs por meio de um contrato inteligente.

Nesse contexto, podemos traçar alguns possíveis caminhos de receita. Para simplificar, vamos supor que o artista fixou o preço do álbum-NFT em 1 ETH. Para fins desta análise, 1 ETH vale 3.000 dólares. Além disso, o artista implementa royalties de 10% em seu contrato inteligente, de modo que sempre que o NFT for revendido no mercado secundário, o artista receberá 10% do preço de venda.

Caminho 1: Venda direta

O artista vende o álbum-NFT diretamente para um fã.

$$\text{Receita} = \text{Preço de venda} = 1 \text{ ETH} = \$3.000$$

Caminho 2: Revenda (sem valorização)

O fã decide vender o álbum-NFT a outro fã pelo mesmo preço.

$$\begin{aligned} \text{Receita} &= \text{Royalties} = 10\% * \text{Preço de revenda} = 10\% * 1 \text{ ETH} = \\ 0.1 \text{ ETH} &= \$300 \end{aligned}$$

Caminho 3: Revenda (com valorização)

O álbum-NFT valoriza e o fã decide vendê-lo por 2 ETH.

Receita = Royalties = 10% * Preço de revenda = 10% * 2 ETH = 0.2 ETH = \$600

E assim por diante para subsequentes revendas.

Observe que, na realidade, a análise seria mais complexa e levaria em consideração outros fatores, como as taxas da plataforma, a volatilidade do preço do ETH e o valor percebido do álbum-NFT pelos fãs.

Espero que isso tenha fornecido uma visão clara de como os contratos inteligentes podem permitir novas formas de monetização para artistas e criadores de conteúdo.

considering the near protocol, detail the smart contract's architecture for such generative album

< Developing a smart contract for a generative album on the Near Protocol involves a number of considerations and steps. For simplicity's sake, let's assume the album consists of 100 tracks and each track is an NFT (Non-Fungible Token).

Here's a brief breakdown of the smart contract architecture:

1. Token Standards:

Near Protocol has its own token standards. The one to be used for this case is the NEP-171 standard which is for NFTs, comparable to ERC721 on Ethereum.

2. Smart Contract Development:

You'll need to create a Rust or AssemblyScript smart contract that adheres to the NEP-171 standard. This contract will manage the creation, ownership, and transfer of your NFTs.

3. Generative Art Creation:

The generative aspect of the album can be built into the smart contract. This involves writing a function that generates a new track (NFT) based on specific parameters or randomness. This could include different beats, melodies, lyrics, etc.

4. Album Sales:

The smart contract can include functionality for selling the album. This could be done in a number of ways, for example, by using a simple transfer function where buyers send NEAR tokens to the contract and receive an NFT in return.

5. Royalty Functionality:

A unique feature that can be added to the smart contract is a royalty system. This allows the original creator of the album to receive a percentage of secondary sales. This can be implemented by adding a royalty function to the contract which sends a percentage of each secondary sale to the creator's NEAR account.

6. Metadata:

Metadata for each track (NFT) can be included in the contract. This could include the name of the track, a description, the date of creation, and even a link to where the track can be streamed.

The smart contract would need to be deployed to the Near Protocol using the Near CLI. After deployment, the contract functions can be called to mint the NFTs, set them for sale, and perform other actions as needed.

Remember that building a smart contract requires a good understanding of blockchain technology and programming, and it's always recommended to have any smart contract audited for security vulnerabilities before deploying it.

Finally, it's worth noting that Near Protocol is designed to be more scalable and efficient than many other blockchains, and it uses a sharded, proof-of-stake model to achieve high throughput and low transaction fees. This makes it a suitable choice for a project like a generative album.

provide an initial mvp sample using rust with only the sales part, focusing the discriminated union relationships in such business logic

< Below is a minimal smart contract in Rust focusing only on the sales part of a generative album. It's a rough template, and you would need to expand this according to your specific needs. In particular, note that the real-world smart contract should handle more safety checks and edge cases.

```
use near_sdk :: borsh :: { self , BorshDeserialize ,
    BorshSerialize } ; use near_sdk :: { near_bindgen ,
    AccountId , Balance , Promise } ; # [ near_bindgen ] #
[ derive ( BorshDeserialize , BorshSerialize ) ] pub
```

```

struct Album {      owner_id: AccountId,      price_
per_track: Balance, } # [ near_bindgen ] impl Album {
    # [ init ]      pub fn new ( owner_id: AccountId,
price_per_track: Balance ) -> Self {      Self
    {      owner_id ,      price_per_track ,
    }      }      pub fn purchase_track ( & mut
self , buyer_id : AccountId, track_id : u64 ) -
> Promise {      assert! ( /* Track exists and
is available for purchase */ ) ;      Promise :: new
( self .owner_id. clone ( ) )      . transfer (
self .price_per_track )      /* Add functionality
to transfer ownership of the track to the buyer */
    }      /* Getter methods would go here */ } # [ cfg (
test ) ] mod tests {      /* Add your tests here */}

```

This contract only includes the basic sale functionality where someone can purchase a track by calling the `purchase_track` method. The balance is then transferred to the owner's account. It is assumed that there's a way to check if the track exists and is available for purchase (which you should implement), and also a way to transfer ownership of the track to the buyer.

The `#[init]` attribute is used to denote the constructor method which initializes the contract.

The discriminated union in Rust is `enum`. For this case, we didn't need it but if you have a requirement where you need to have a type which can have several different variants, then you can use the `enum` keyword.

For example, suppose you had different kinds of albums in your marketplace, like a single track album and a multiple tracks album, you can model that using an enum:

```

enum AlbumType {      SingleTrack ( Track ) ,      MultipleTracks
( Vec < Track > ) , }

```

Remember that building a smart contract requires a good understanding of blockchain technology and programming, and it's always recommended to have any smart contract audited for security vulnerabilities before deploying it.

keep outputting in pt-br. considering the cranky state of the music industry, with many oddities like ‘single track albums’, detail other things that could be nitpicked as additional complexity for such contract in order to adhere to retro compatibility

< Certamente, existem muitos fatores a considerar ao projetar um contrato inteligente para um projeto de álbum de música gerativo para se ajustar ao estado diversificado e em constante mudança da indústria da música. Aqui estão alguns pontos a serem considerados para adicionar complexidade e retrocompatibilidade ao contrato:

1. **Edições de Faixas:** Músicas podem ter várias versões, como edições de rádio, remixes, versões acústicas, versões ao vivo, etc. O contrato inteligente precisará de uma maneira de lidar com todas essas versões.
2. **Colaborações:** Muitas músicas são o produto de colaborações entre diferentes artistas. O contrato inteligente precisaria ser capaz de dividir pagamentos entre vários artistas de forma justa.
3. **Direitos Autorais:** A indústria da música é notória por suas complexidades legais em relação aos direitos autorais. Seu contrato inteligente precisará de uma maneira de lidar com esses direitos autorais, possivelmente distribuindo royalties para compositores, produtores e artistas conforme apropriado.
4. **Lançamentos Regionais:** Muitos álbuns têm lançamentos diferentes em diferentes regiões. O contrato inteligente precisará ser capaz de lidar com essa complexidade.
5. **Vendas Físicas e Digitais:** Embora o contrato inteligente esteja lidando principalmente com vendas digitais, ainda pode haver a necessidade de registrar vendas físicas para total transparência e rastreabilidade. Como isso será incorporado ao sistema precisaria ser considerado.
6. **Tabelas de Classificação:** Álbuns e faixas costumam ser classificados em várias tabelas. Pode ser útil ter uma maneira de registrar essas classificações dentro do contrato inteligente.

7. **Recursos de Prevenção à Fraude:** Para evitar comportamentos fraudulentos, como a manipulação de streamings ou vendas, o contrato inteligente deveria ter salvaguardas embutidas.

Esses são apenas alguns exemplos de possíveis complicações que você pode encontrar ao tentar adaptar um contrato inteligente para a indústria da música. Como sempre, seria crucial garantir que todas as soluções sejam juridicamente sólidas e seguras do ponto de vista da tecnologia da informação.

good intro, add more intricate points for completion

< I'm glad you found the initial points helpful. Here are some more intricate aspects to consider:

1. **Escalation of Royalties:** Some contracts in the music industry have clauses that increase an artist's royalty rate after a certain number of units are sold or streams are reached. This would add complexity to your smart contract, which would need to be programmed to adjust royalty rates accordingly.
2. **Recoupment:** In traditional record deals, artists often have to repay their advances before they start earning royalties. This concept might need to be integrated into your smart contract, though it could be complex, given the decentralized and transparent nature of blockchain technology.
3. **Sampling:** Many songs sample other songs. The smart contract could potentially handle the automatic payment of royalties to the artists being sampled, but this would require careful planning and programming.
4. **Cover Songs:** Cover songs are another point of complexity. The royalties from cover songs need to be split between the original artist and the artist performing the cover. This could add another layer of complexity to your smart contract.
5. **Merchandising and Ancillary Rights:** Many artists earn a significant portion of their income from merchandise sales, concert tickets, and other ancillary rights. Incorporating these into your smart contract could increase its complexity.

6. **Sync Licensing:** When music is used in commercials, films, or TV shows, the artists are typically paid through synchronization licenses. This could also be integrated into your smart contract, but would require careful consideration.
7. **Subscription Models and Freemium Models:** If your music project is considering alternative revenue models like subscriptions or freemium models, these would also need to be reflected in your smart contract.
8. **Legacy Catalog Integration:** If you want to be retrocompatible, integrating older music catalogs into your system could present challenges due to their existing contracts and the variety of formats they might be in.

Each of these points adds additional layers of complexity to the design of your smart contract, requiring careful consideration and planning, as well as thorough legal and technical review.

stop switching back to en-us, gpt. detail features related to randomization (prng with the last hash as seed) that can be added that can become unique ways of overthrowing web2 tools like spotify, youtube or soundcloud

< Desculpe pela confusão anterior. Agora, vou abordar as possíveis funcionalidades relacionadas à aleatoriedade que podem ser adicionadas ao seu contrato inteligente, as quais podem proporcionar uma vantagem única sobre ferramentas web2 como Spotify, YouTube ou SoundCloud.

1. **Geração de Playlists Únicas:** Em vez de simplesmente aleatorizar as faixas de um álbum, você poderia usar a randomização para gerar playlists únicas para cada usuário. Isso criaria uma experiência única para cada usuário toda vez que eles ouvissem a música, proporcionando um elemento de surpresa e exclusividade.
2. **Sorteios de Criptomoedas:** Você poderia usar a aleatoriedade para realizar sorteios entre seus ouvintes. Por exemplo, você poderia sortear uma quantidade de criptomoeda entre todos os

usuários que ouvirem um álbum em um determinado período. Isso incentivaria as pessoas a ouvirem suas músicas e participarem ativamente da sua plataforma.

3. **Experiências Musicais Evolutivas:** Usando a aleatoriedade, você poderia criar faixas musicais que mudam e evoluem ao longo do tempo. Por exemplo, a melodia, harmonia ou ritmo de uma música pode ser alterada aleatoriamente a cada vez que é tocada. Isso tornaria a música verdadeiramente “viva” e proporcionaria uma experiência de audição sempre nova.
4. **Sistema de Recompensas Dinâmico:** Você poderia usar a aleatoriedade para gerar um sistema de recompensas dinâmico. Por exemplo, a quantidade de royalties que um artista recebe por cada stream poderia ser ligeiramente alterada com base em uma função de randomização. Isso criaria uma dinâmica interessante, onde tanto artistas quanto ouvintes teriam um incentivo adicional para engajar com a plataforma.
5. **Lançamentos de Álbuns Aleatórios:** Usando a aleatoriedade, você poderia ter lançamentos de álbuns que são únicos para cada usuário. Por exemplo, a ordem das faixas ou até mesmo quais faixas aparecem em um álbum poderia ser determinada aleatoriamente. Isso criaria uma sensação de antecipação e surpresa cada vez que um novo álbum fosse lançado.

Lembrando que todas essas funcionalidades dependem de um design e implementação cuidadosos do contrato inteligente, e devem ser feitas levando em conta as implicações legais e éticas de tais mecanismos. Além disso, é crucial garantir a segurança e a transparência ao usar qualquer forma de aleatoriedade na blockchain.

these are good ideas, but it might be somewhat needed by nature for tracks to be static, no? how to go ham with this hash prng, but on dynamics that don't change the song itself after being crystalized

< Peço desculpas pela confusão anterior. Você está absolutamente correto em observar que uma vez que uma música é “cristalizada” na

blockchain, ela se torna imutável. A utilização de PRNG (Pseudorandom Number Generator) com base no último hash de bloco poderia ser usada para introduzir dinâmica em outros aspectos da experiência musical que não alteram a música em si. Aqui estão algumas ideias:

1. **Ordem de Reprodução:** O PRNG pode ser usado para determinar a ordem em que as músicas são reproduzidas. Isso seria útil para álbuns ou playlists onde a ordem das músicas pode ser alterada para oferecer uma nova experiência ao ouvinte cada vez que a música é tocada.
2. **Descoberta de Música:** O PRNG pode ser usado para apresentar novas músicas aos ouvintes. A plataforma poderia usar o PRNG para selecionar uma música que ainda não foi ouvida pelo usuário. Isso poderia incentivar a descoberta de novas músicas e artistas.
3. **Interatividade do Ouvinte:** Os ouvintes poderiam “girar uma roda” PRNG que determina que tipo de conteúdo extra eles recebem. Isso poderia ser qualquer coisa, desde arte do álbum até ingressos para shows, passando por faixas bônus.
4. **Sistema de Recompensas:** O PRNG poderia ser usado para determinar recompensas para os ouvintes. Por exemplo, cada vez que um ouvinte reproduz uma música, ele tem a chance de receber uma recompensa, como um desconto em mercadorias, ingressos para um show ou acesso a conteúdo exclusivo.
5. **Distribuição de Royalties:** Em vez de distribuir royalties de forma uniforme, o PRNG poderia ser usado para introduzir algum elemento de variabilidade. Por exemplo, um artista poderia optar por distribuir uma porcentagem maior de royalties para as primeiras 1.000 reproduções de uma música, incentivando os ouvintes a serem os primeiros a ouvir.
6. **Engajamento de Fãs:** Use o PRNG para criar eventos especiais para os ouvintes mais engajados. Por exemplo, você poderia selecionar aleatoriamente um ouvinte para uma sessão de perguntas e respostas ao vivo com o artista após um determinado número de reproduções.

Essas são apenas algumas ideias de como você pode usar o PRNG para adicionar dinâmica à experiência do ouvinte sem alterar a música em si. O céu é o limite quando se trata de inovação nesta área!

list in pt-br 15 things that can be listened to while programming

1. Música clássica
2. Trilhas sonoras de filmes
3. Podcasts sobre tecnologia
4. Palestras TED
5. Música instrumental
6. Livros em áudio
7. Sons da natureza
8. Música eletrônica
9. Jazz
10. Música de meditação
11. Música ambiente
12. Música lo-fi (low fidelity)
13. Palestras de programação
14. Música barroca
15. Sons de chuva ou ruídos brancos

for each item, give one example and one trivia

1. Música clássica: Exemplo: “Concerto para Piano No. 21” de Mozart. Curiosidade: Estudos sugerem que a música clássica pode melhorar a concentração e a eficiência.
2. Trilhas sonoras de filmes: Exemplo: Trilha sonora de “Interstellar” de Hans Zimmer. Curiosidade: Muitas trilhas sonoras são projetadas para evocar emoções específicas sem distrair o público do filme.
3. Podcasts sobre tecnologia: Exemplo: “Hipsters Ponto Tech”. Curiosidade: O termo “podcast” é uma junção das palavras “iPod” e “broadcast”.
4. Palestras TED: Exemplo: “The Future of Programming” de Bret Victor. Curiosidade: O nome TED é um acrônimo para Tecnologia, Entretenimento e Design.
5. Música instrumental: Exemplo: “Ocean” de John Butler. Curiosidade: Música instrumental é conhecida por ajudar na concentração por não ter letras que podem distrair.
6. Livros em áudio: Exemplo: “Clean Code: A Handbook of Agile Software Craftsmanship” de Robert C. Martin. Curiosidade: A primeira tentativa de criação de um audiolivro data de 1931, com o intuito de ajudar os cegos.
7. Sons da natureza: Exemplo: Sons de pássaros e água corrente. Curiosidade: A “ecoacústica” é o estudo científico dos sons da natureza.
8. Música eletrônica: Exemplo: “Strobe” de Deadmau5. Curiosidade: A música eletrônica tem suas origens em experimentos com equipamentos eletrônicos e tecnologia no final do século 19.
9. Jazz: Exemplo: “Take Five” de Dave Brubeck. Curiosidade: O Jazz é reconhecido como um gênero musical que começou entre a comunidade afro-americana em Nova Orleans no final do século 19 e início do 20.
10. Música de meditação: Exemplo: “Weightless” de Marconi Union. Curiosidade: Essa faixa foi cientificamente comprovada como a música mais relaxante já criada.

11. Música ambiente: Exemplo: “Music for Airports” de Brian Eno. Curiosidade: Brian Eno é muitas vezes creditado como o inventor do termo “música ambiente”.
12. Música lo-fi (low fidelity): Exemplo: “we’ve never met but can we have a coffee or something” de In Love With A Ghost. Curiosidade: A estética lo-fi é apreciada por seu som “imperfeito” ou “não polido”.
13. Palestras de programação: Exemplo: “The Myth of the Genius Programmer” de Google Tech Talks. Curiosidade: Essas palestras são uma ótima maneira de se manter atualizado sobre as últimas tendências e técnicas em programação.
14. Música barroca: Exemplo: “As Quatro Estações” de Vivaldi. Curiosidade: A música barroca, como a de Bach e Vivaldi, é frequentemente citada em estudos de concentração e aprendizado.
15. Sons de chuva ou ruídos brancos: Exemplo: Sons de uma tempestade de chuva. Curiosidade: O “ruído branco” recebe esse nome porque contém todas as frequências, assim como a luz branca que contém todas as cores.
