ДДД Дистрибуција на албуми

1. Опишете го сценариото кое го обработувате (200 збора).

Секој музичар/артист којшто издава албуми и песни во јавноста сака да има некоја подршка при дистрибуцијата и маркетингот на музиката. Онлајн платформите наречени музички дистрибутори му овозможуваат на секој музичар да може да стигне до голем број на публика. Соодветно, како што е поголема сумата којашто ја плаќа артистот, така поголем е и рангот на којшто истиот ќе се пласира и ќе биде дистрибуиран во поширoки рамки.

Еден артист/музичар може да направи повеќе албуми. Тој има свое име и презиме, уметничко име, како и, свој мејл и телефонски број.

Албумот е направен од страна на некој музичар/артист, со помош на продуцентот и композиторот. Тој освен тоа соодветно содржи информации за своето име, како и жанрот во којшто припаѓа. Секој албум има своја колекција од песни.

Песната којашто спаѓа во одреден албум има свое име и должина. Таа може од претходно да биде издавана како сингл.

Музичкиот дистрибутор е тој којшто му овозможува на артистот дистрибуција на музиката на секоја позната онлајн платформа. Тој дистрибутор има свое име и име на компанијата на којашто тој припаѓа. Тој чува информација за секој дистрибуиран албум којшто го покрива.

2. Идентификувајте ги ентитетите и релациите меѓу нив кои ви се потребни за имплементација на сценариото. Бројот на дефинирани ентитети мора да биде најмалку 3. Нацртајте го и ЕА дијаграмот и прикачете го како слика.

Ентитети:

- Artist

- Album

- Song

- MusicDistributor

- PublishedAlbum

Релации:

(1 -> N врска) -> (Artist -> Album)

(1 -> N врска) -> (Artist -> Song) (додадено, за покомплетно поврзување)

(1 -> N врска) -> (Album -> Song)

(1 -> N врска) -> (Artist -> PublishedAlbum)

(1 -> N врска) -> (MusicDistributor -> PublishedAlbum)

3. Идентификувајте ги потребните атрибути за секој од ентитетите. За секој од атрибутите дефинирајте го типот.

- Artist:

- artistId (String)
- firstName (String)
- lastName (String)
- artName (String)
- email (String)
- **password (String) (додадено, поради потреба од автентикација на фронт-енд)**
- telephoneNumber (String)
- albumList (List<Album>)
- **songList (List<Song>) (додадено, бидејќи песните можат да бидат и синглови, односно да постојат независно од некој албум)**

- Album:

- albumId (String)
- albumName (String)
- artistName (String)
- producerName (String)
- composerName (String)
- totalLength (Integer)
- isPublished (Boolean)
- genre (*Enum*: POP, ROCK, METAL, JAZZ, FUNK, RNB)
- songList (List<Song>)

- Song:

- songId (String)
- songLength (Integer)
- songName (String)
- isASingle (Boolean)

- MusicDistributor

- distributorId (String)
- companyName (String)
- distributorName (String)

~~- totalEarned (Double)~~ (одземено, поради тоа што имаме бизнис правило во самиот ентитет којашто ја имплементира оваа функционалност)

- publishedAlbumList (List<PublishedAlbum>)

- PublishedAlbum:

- publishedAlbumId (String)
- albumId (String)
- albumName (String)
- artistId (String)
- artistInformation (String)
- subscriptionFee (Double)

- transactionFee (Double) (додадено, бидејќи при публикување на албум, освен таксата за претплатување има и такса за трансакција)

- publishedOn (Instant)
- albumTier (*Enum*: BRONZE, SILVER, GOLD, PLATNIUM, DIAMOND)

4. Идентификувајте ги ограничените контексти (bounded contexts) во вашето сценарио.

- Bounded Context 1:

- Artist
- Album
- Song

- Bounded Context 2:

- MusicDistributor
- PublishedAlbum

- Shared Kernel

5. Идентификувајте ги агрегатите во секој од ограничените контексти.

- Bounded Context 1:

- *Aggregate 1:*
  - Artist
  - Album
  - Song

- Bounded Context 2:

- *Aggregate 2:*
  - MusicDistributor
  - PublishedAlbum

- Shared Kernel:

6. Идентификувајте го Aggregate Root на секој од идентификуваните агрегати.

- Bounded Context 1:

- *Aggregate 1:*
  - **Artist -> Aggregate Root (AR)**
  - Album
  - Song

- Bounded Context 2:

- Aggregate 2:

- MusicDistributor -> Aggregate Root (AR)

- PublishedAlbum

- Shared Kernel:

7. Идентификувајте неколку правила за конзистентност (бизнис правила) во сценариото. Специфицирајте кој ентитет ќе ги поседува имплементациите на истите?

За секој од соодветните ентитети се напишани и правилата на конзистентност коишто ги имплементираат. (означени со + знакот, кои исто така означуваат методи во класите-ентитети)

- Bounded Context 1:

- Artist

+ createAlbum (Album album): Album (креира нов албум)

+ addSongToArtist (Song song): Song (додава нова сингл песна) (изменето, бидејќи веќе има метод за додавање на песна во албум (во ентитетот албум), а немаше метод за додавање на сингл песна)

+ removeSongFromArtist (Song song): Song (брише сингл песна) (изменето, бидејќи веќе има метод за бришење на песна од албум (во ентитетот албум), а немаше метод за бришење на сингл песна)

+ makeAlbumPublished (AlbumId albumId): Boolean (додава информација на албумот дека тој е соодветно дистрибуиран)

+ makeAlbumUnpublished (AlbumId albumId): Boolean (додава информација за албумот дека е ндостапен на онлајн платформите)

- Album

+ addSong(Song song): Song (додава нова песна во албумот)

+ removeSong(SongId songId): Song (вади песна од албумот)

+ totalLength(): SongLength (ја пресметува вкупната должина на албумот)

+ publish(): Boolean (додава информација на албумот дека тој е соодветно дистрибуиран)

+ unpublish(): Boolean (додава информација на албумот дека тој е недистрибуиран, одосно не е достапен на онлајн платформите)

- Bounded Context 2:

- MusicDistributor

+ totalEarnings(): Money (го пресметува вкупниот износ којшто го заработува музичкиот дистрибутор од сите албуми во своето складиште)

+ subscribeAlbum (PublishedAlbum publishedAlbum): PublishedAlbum (додавање на нов албум во складиштето на музичкиот дистрибутор)

+ unsubscribeAlbum (PublishedAlbum publishedAlbum): PublishedAlbum (бришење на албум од складиштето на музичкиот дистрибутор)

- PublishedAlbum

+ earningsPerAlbum(): Money (го пресметува вкупниот износ којшто го заработува музичкиот дистрибутор од соодветниот албум)

+ raiseAlbumTier(Tier tier, Double subscriptionFee, Double transactionFee): Tier (го зголемува нивото на албумот) (додадено, со цел да може да има правилно распоредување на бизнис логиката за извршување на настанот)

- Shared Kernel:

8. Идентификувајте неколку вредносни објекти (value-objects) во вашето сценарио. Кои методи би ги имплементирале?

Во прилог е дадена листа од вредносни објекти, и соодветно подолу, листа од атрибутите во соодветните ентитети на којшто се задаваат вредносните објекти како тип.

Исто така, треба да се напомене дека во *PublishedAlbum* се додаваат уште два додатни атрибути: *albumId* и *artistId*, затоа што е раскината *1 -> N* врската помеѓу *Artist* и *PublishedAlbum,* преку која артистот додава некој албум на страната на музичкиот дистрибутор, бидејќи веќе имаме два различни органичени контексти.

Вредносни објекти (со знакот - се прикажани сите атрибути коишто ги содржи вредносниот објект, а со + се прикажани сите методи кои ги содржи вредносниот објект):

- DomainObjectId:

- Id (String)

- AbstractEntity<ID extends DomainObjectId>:

- Id (ID)

- ArtistId extends DomainObjectId:

+ ArtistId(): ArtistId (конструктор)

- AlbumId extends DomainObjectId:

+ AlbumId(): AlbumId (конструктор)

- SongId extends DomainObjectId:

+ SongId(): SongId (конструктор)

- PublishedAlbumId extends DomainObjectId:

+ PublishedAlbumId(): PublishedAlbumId (конструктор)

- DistributorId extends DomainObjectId:

+ DistributorId(): DistributorId (конструктор)

- ArtistPersonalInfo:

- firstName (String)
- lastName (String)
- artName (String)

+ ArtistPersonalInfo(): ArtistPersonalInfo (конструктор)
+ ArtistPersonalInfo(firstName, lastName, artName): ArtistPersonalInfo (конструктор)
+ getFirstName(): String (гетер метод за името на артистот)
+ getLastName(): String (гетер метод за презимето на артистот)
+ getArtName(): String (гетер метод за уметничкото име на артистот)

+ getFullName(): String (гетер метод за целосното име на артистот (име + презиме)) (додадено, со цел полесно и пооптимално добивање на податок за артистот на фронт-енд во апликацијата)

- ArtistContactInfo:

- email (Email)
- telephoneNumber (String)
+ ArtistContactInfo(email, telephoneNumber): ArtistContactInfo (конструктор)
+ getEmail(): Email (гетер метод за мејлот на артистот)
+ getPhoneNumber(): String (гетер метод за телефонскиот број на артистот)

- Email:

- username (String)
- domainName (*Enum*: YAHOO, GMAIL, OUTLOOK, STUDENTS-FINKI)
+ Email(username, domainName): Email (конструктор)
+ getUsername(): String (гетер метод за корисничкото име од мејлот на артистот)
+ getDomainName(): String (гетер метод за доменот од мејлот на артистот)
+ getFullAddress(): String (гетер метод за целосната мејл адреса на артистот)

- AlbumInfo:

- artistName (String)
- producerName (String)
- composerName (String)
+ AlbumInfo(artistName, producerName, composerName): AlbumInfo (конструктор)
+ getArtistName(): String (гетер метод за името на артистот)
+ getProducerName(): String (гетер метод за името на продуцентот)
+ getComposerName(): String (гетер метод за името на композиторот)

- **SongLength**

- lengthInSeconds (Integer)
+ SongLength(lengthInSeconds): SongLength (конструктор)
+ addSecondsToSongLength(): SongLength (додавање на секунди во должината на некоја песна)
+ removeSecondsFromSongLength(): SongLength (бришење на секунди во должината на некоја песна)
+ getLengthSeconds(): Integer (гетер метод за должината на песната во секунди) (изменето, заради потребата од земање на должината на песната во секунди)

- DistributorInfo

- companyName (String)

- distributorName (String)

+ DistributorInfo(companyName, distributorName): DistributorInfo (конструктор)

+ getCompanyName(): String (гетер метод за името на команијата)

+ getDistributorName(): String (гетер метод за името на дистрибуторот)

+ getDistributorInformation(): String (гетер метод за целокупната информација за дистрибутерот) (додадено, со цел оптимизација на кодот и само еден повик на фронт-енд на апликацијата кон објектот)

- Money

- amount (Double)

- currency (*Enum* - EUR, USD, MKD)

+ Money(amount, currency): Money (конструктор)

+ valueOf(currency, amount): Money (одредување на паричната вредност на некоја сума со одредена валута)

+ add(money): Money (собирање на парични вредности во соодветни валути)

+ substract(money): Money (одземање на парични вредности во соодветни валути)

+ multiply(money): Money (множење на парични вредности во соодветни валути)

Ентитети со соодветни вредносни објекти:

- Bounded Context 1:

- Artist:

- artistId (ArtistId)

- artistContactInfo (ArtistContactInfo)

- artistPersonalInfo (ArtistPersonalInfo)

- password (String) (додадено, поради потреба од автентикација)

- albumsList (List<Album>)

- songList (List<Song>) (додадено, бидејќи треба да ги чуваме и податоците за песните коишто се синглови и ги нема на ниту еден албум)

- **Album:**
    - albumId (AlbumId)
    - albumName (String)
    - albumInfo (AlbumInfo)
    - totalLength (SongLength)
    - isPublished (Boolean)
    - genre (Genre)
    - songList (List<Song>)
    - **creator (Artist) (додадено, со цел полесен пристап до креаторот)**
- **Song:**
    - songId (SongId)
    - songLength (SongLength)
    - songName (String)
    - isASingle (Boolean)
    - **creator (Artist) (додадено, со цел полесен пристап до креаторот)**
    - **album (Album) (додадено, со цел полесен пристап до албумот)**
- **Genre (ENUM):** POP, ROCK, METAL, JAZZ, FUNK, RNB

- **Bounded Context 2:**

- **PublishedAlbum:**
    - publishedAlbumId (PublishedAlbumId)
    - albumId (AlbumId)
    - **albumName (String) (додадено, со цел полесен пристап до албумот)**
    - artistId (ArtistId)
    - **artistinformation (String) (додадено, за пристап до креаторот)**
    - subscriptionFee (Money)
    - **transactionFee (Money) (додадено, за потреба како додатен трошок)**
    - publishedOn (Instant)
    - albumTier (Tier)
    - **publisher (MusicDistributor) (додадено, за пристап до публикувачот)**

- **MusicDistributor**
    - distributorId (DistributorId)
    - distributorInfo (DistributorInfo)
    - ~~**totalEarned (Money)**~~ **(одземено, поради тоа што имаме бизнис правило во самиот ентитет којашто ја имплементира оваа функционалност)**
    - listPublishedAlbums (List<PublishedAlbum>)

- **Shared Kernel:**
    - **DomainObjectId:**
        - Id (String)
    - **AbstractEntity<ID extends DomainObjectId>:**
        - Id (ID)
    - **Money**
        - amount (Double)
        - currency (Currency)
    - **Email:**
        - username (String)
        - domainName (EmailDomain)
    - **Currency (ENUM):** EUR, USD, MKD
    - **EmailDomain (ENUM):** YAHOO, GMAIL, OUTLOOK, STUDENTS-FINKI
    - **Tier (ENUM):** BRONZE, SILVER, GOLD, PLATNIUM, DIAMOND

9. Идентификувајте неколку настани (events) кои треба да протекуваат помеѓу агрегатите.

Event 1: Артистот претплатува некој албум на соодветниот музички дистрибутер преку предавање на неколку потребни информации за себе и албумот во аргумент, и правење една трансакција како надоместок за услугата на дистрибутерот. Лолку што е поголема трансакцијата, толку е поголем рангот и приоритетот на дистрибутерот за тој албум

+ AlbumPublishedEvent(String albumId, String artistId, String musicPublisherId, String albumTier, Double subscriptionFee, Double transactionFee): PublishedAlbum

Event 2: Артистот отпретплатува некој свој албум од соодветниот музички дистрибутер преку предавање на својот, и, идентификаторот на албумот во аргумент

+ AlbumUnpublishedEvent (String publishedAlbumId): PublishedAlbum

Event 3: Соодветниот музички дистрибутер го известува артистот дека неговиот албум е дистрибуиран и достапен за широката јавност, или, дека има некоја грешка при публикувањето

~~**+ albumPublishedEvent(): PublishedAlbum**~~ **(одземено, бидејќи е направена одлука бизнис логиката за претплатување на албум да биде на страната во модулот/апликацијата на музичкиот дистрибутор заради пологична и полесна имплементација, а настаните 1 и 2 го прават соодветното известување, што иницијално беше работата на овој настан)**

Event 4 (3): Артистот го зголемува рангот/достапноста на својот дистрибуиран албум кај музичкиот дистрибутор, со тоа што доплаќа повеќе за услугата, и ги задава потребните информации за себе и албумот во аргумент

+ raiseAlbumTier(String publishedAlbumId, String albumTier): PublishedAlbum