**Models**

Estrutura de dados e metodos planejados

### **Cipher**

id String

title String

author String

tempo String

tags List~String~

key Char

language String

createdAt dateTime

updatedAt dateTime

isLocal bool

musicStruct MusicalStructure

toJson() -> Map~String, dynamic~

fromJson(Map) -> Cipher

transpose(String newKey) Cipher

### **MusicalStructure**

structureMap map ~String, String~

exemplo: {

‘verso1’: Conteúdo,

‘chorus’: Conteúdo,

}

getMap() List ~String~

### **Playlist**

id String

name String

description String?

cipherIds List~String~

createdBy String

createdAt DateTime

updatedAt DateTime

collaborators List~String~

toJson() -> Map~String, dynamic~

fromJson(Map) Playlist

addCipher(cipherId)

removeCipher(cipherId)

reorderCipher (List~string~)

### **User**

id String

email String

displayName String

photoUrl String?

createdAt DateTime

lastLoginAt DateTime

preferences UserPreferences

toJson() -> Map~String, dynamic~

fromJson(Map) -> User

**UserPreferences**

language String

fontSize int

fontFamily String

showChords bool

showLyrics bool

showNotes bool

toJson -> Map~String, dynamic~

fromJson(Map) -> UserPreferences

CLAUDE

classDiagram

class User {

+String id

+String email

+String displayName

+String? photoUrl

+DateTime createdAt

+DateTime lastLoginAt

+UserPreferences preferences

+toJson() Map~String, dynamic~

+fromJson(Map) User

}

class UserPreferences {

+String language

+int fontSize

+String fontFamily

+bool showChords

+bool showLyrics

+bool showNotes

+String defaultKey

+toJson() Map~String, dynamic~

}

class Cifra {

+String id

+String title

+String artist

+String originalKey

+String? capo

+String content

+String language

+List~String~ tags

+MusicalStructure structure

+DateTime createdAt

+DateTime updatedAt

+String? createdBy

+bool isLocal

+toJson() Map~String, dynamic~

+fromJson(Map) Cifra

+transpose(String newKey) Cifra

}

class MusicalStructure {

+List~Verse~ verses

+List~Chorus~ choruses

+List~Bridge~ bridges

+List~Note~ notes

+String? intro

+String? outro

+addVerse(Verse) void

+addChorus(Chorus) void

+getStructureOrder() List~String~

}

class Verse {

+String id

+String content

+int order

+String color

+String? label

}

class Chorus {

+String id

+String content

+String color

+bool isRepeated

}

class Bridge {

+String id

+String content

+String color

+String? label

}

class Note {

+String id

+String content

+String type

+int position

+String color

}

class Playlist {

+String id

+String name

+String? description

+List~String~ cifrIds

+String createdBy

+DateTime createdAt

+DateTime updatedAt

+bool isPublic

+bool isCollaborative

+List~String~? collaborators

+PlaylistSettings settings

+toJson() Map~String, dynamic~

+fromJson(Map) Playlist

+addCifra(String cifrId) void

+removeCifra(String cifrId) void

+reorderCifras(List~String~) void

}

class PlaylistSettings {

+bool autoAdvance

+int autoAdvanceDelay

+bool showChords

+bool showLyrics

+bool showNotes

+bool presentationMode

}

class RealtimeSession {

+String id

+String bandId

+String playlistId

+String createdBy

+DateTime createdAt

+bool isActive

+int currentSongIndex

+String currentSongId

+int currentPage

+DateTime lastUpdated

+String lastUpdatedBy

+Map~String, Participant~ participants

+toJson() Map~String, dynamic~

+fromJson(Map) RealtimeSession

}

class Participant {

+String userId

+String displayName

+String instrument

+DateTime joinedAt

+DateTime lastSeen

+bool isLeader

+bool isConnected

+toJson() Map~String, dynamic~

}

class Band {

+String id

+String name

+String? description

+List~String~ memberIds

+String createdBy

+DateTime createdAt

+List~String~ adminIds

+toJson() Map~String, dynamic~

+fromJson(Map) Band

+addMember(String userId) void

+removeMember(String userId) void

}

class PageTurnEvent {

+String sessionId

+String userId

+String songId

+int pageNumber

+DateTime timestamp

+String eventType

+toJson() Map~String, dynamic~

}

class CacheManager {

+Duration cacheExpiry

+int maxCacheSize

+getCachedCifra(String id) Cifra?

+cacheCifra(Cifra cifra) void

+clearExpiredCache() void

+getCacheStats() CacheStats

}

class DatabaseService {

+Database db

+insertCifra(Cifra cifra) Future~int~

+getCifra(String id) Future~Cifra?~

+getAllCifras() Future~List~Cifra~~

+updateCifra(Cifra cifra) Future~void~

+deleteCifra(String id) Future~void~

+searchCifras(String query) Future~List~Cifra~~

}

class FirestoreService {

+FirebaseFirestore firestore

+downloadCifra(String id) Future~Cifra~

+searchOnlineCifras(String query) Future~List~Cifra~~

+uploadCifra(Cifra cifra) Future~String~

+syncUserData(User user) Future~void~

}

class RealtimeService {

+DatabaseReference database

+createSession(RealtimeSession session) Future~void~

+joinSession(String sessionId, Participant participant) Future~void~

+turnPage(String sessionId, int page) Future~void~

+leaveSession(String sessionId, String userId) Future~void~

+listenToSession(String sessionId) Stream~RealtimeSession~

}

*%% Relacionamentos*

User ||--|| UserPreferences : has

User ||--o{ Cifra : creates

User ||--o{ Playlist : creates

User ||--o{ Band : memberOf

Playlist ||--o{ Cifra : contains

Playlist ||--|| PlaylistSettings : has

Cifra ||--|| MusicalStructure : has

MusicalStructure ||--o{ Verse : contains

MusicalStructure ||--o{ Chorus : contains

MusicalStructure ||--o{ Bridge : contains

MusicalStructure ||--o{ Note : contains

RealtimeSession ||--|| Playlist : syncs

RealtimeSession ||--o{ Participant : has

RealtimeSession ||--o{ PageTurnEvent : generates

Band ||--o{ User : has

Band ||--o{ RealtimeSession : creates

*%% Services*

DatabaseService ..> Cifra : manages

DatabaseService ..> Playlist : manages

FirestoreService ..> Cifra : syncs

RealtimeService ..> RealtimeSession : manages

CacheManager ..> Cifra : caches