**import** Foundation

**import** FirebaseFirestore

**struct** Client: Identifiable, Codable, Equatable, Hashable {

@DocumentID **var** id: String?

**var** typ: String

**var** name: String

**var** vorname: String

**var** geschlecht: String

**var** vereinID: String?

**var** nationalitaet: [String]?

**var** geburtsdatum: Date?

**var** alter: Int?

**var** kontaktTelefon: String?

**var** kontaktEmail: String?

**var** adresse: String?

**var** liga: String?

**var** vertragBis: Date?

**var** vertragsOptionen: String?

**var** gehalt: Double?

**var** schuhgroesse: Int?

**var** schuhmarke: String?

**var** starkerFuss: String?

**var** groesse: Int?

**var** gewicht: Int?

**var** positionFeld: [String]?

**var** sprachen: [String]?

**var** lizenz: String?

**var** nationalmannschaft: String?

**var** profilbildURL: String?

**var** transfermarktID: String?

// Equatable-Implementierung

**static** **func** == (lhs: Client, rhs: Client) -> Bool {

**return** lhs.id == rhs.id &&

lhs.typ == rhs.typ &&

lhs.name == rhs.name &&

lhs.vorname == rhs.vorname &&

lhs.geschlecht == rhs.geschlecht &&

lhs.vereinID == rhs.vereinID &&

lhs.nationalitaet == rhs.nationalitaet &&

lhs.geburtsdatum == rhs.geburtsdatum &&

lhs.alter == rhs.alter &&

lhs.kontaktTelefon == rhs.kontaktTelefon &&

lhs.kontaktEmail == rhs.kontaktEmail &&

lhs.adresse == rhs.adresse &&

lhs.liga == rhs.liga &&

lhs.vertragBis == rhs.vertragBis &&

lhs.vertragsOptionen == rhs.vertragsOptionen &&

lhs.gehalt == rhs.gehalt &&

lhs.schuhgroesse == rhs.schuhgroesse &&

lhs.schuhmarke == rhs.schuhmarke &&

lhs.starkerFuss == rhs.starkerFuss &&

lhs.groesse == rhs.groesse &&

lhs.gewicht == rhs.gewicht &&

lhs.positionFeld == rhs.positionFeld &&

lhs.sprachen == rhs.sprachen &&

lhs.lizenz == rhs.lizenz &&

lhs.nationalmannschaft == rhs.nationalmannschaft &&

lhs.profilbildURL == rhs.profilbildURL

}

// Hashable-Implementierung

**func** hash(into hasher: **inout** Hasher) {

hasher.combine(id)

hasher.combine(typ)

hasher.combine(name)

hasher.combine(vorname)

hasher.combine(geschlecht)

hasher.combine(vereinID)

hasher.combine(nationalitaet)

hasher.combine(geburtsdatum)

hasher.combine(alter)

hasher.combine(kontaktTelefon)

hasher.combine(kontaktEmail)

hasher.combine(adresse)

hasher.combine(liga)

hasher.combine(vertragBis)

hasher.combine(vertragsOptionen)

hasher.combine(gehalt)

hasher.combine(schuhgroesse)

hasher.combine(schuhmarke)

hasher.combine(starkerFuss)

hasher.combine(groesse)

hasher.combine(gewicht)

hasher.combine(positionFeld)

hasher.combine(sprachen)

hasher.combine(lizenz)

hasher.combine(nationalmannschaft)

hasher.combine(profilbildURL)

}

}

**struct** Funktionär: Identifiable, Codable, Hashable {

@DocumentID **var** id: String?

**var** name: String

**var** vorname: String

**var** abteilung: String?

**var** vereinID: String?

**var** kontaktTelefon: String?

**var** kontaktEmail: String?

**var** adresse: String?

**var** clients: [String]?

**var** profilbildURL: String?

**var** geburtsdatum: Date?

**var** positionImVerein: String? // Neues Feld für die Position

**var** mannschaft: String? // Neues optionales Feld für die Mannschaft

// Equatable-Implementierung für Hashable

**static** **func** == (lhs: Funktionär, rhs: Funktionär) -> Bool {

**return** lhs.id == rhs.id &&

lhs.name == rhs.name &&

lhs.vorname == rhs.vorname &&

lhs.abteilung == rhs.abteilung &&

lhs.vereinID == rhs.vereinID &&

lhs.kontaktTelefon == rhs.kontaktTelefon &&

lhs.kontaktEmail == rhs.kontaktEmail &&

lhs.adresse == rhs.adresse &&

lhs.clients == rhs.clients &&

lhs.profilbildURL == rhs.profilbildURL &&

lhs.geburtsdatum == rhs.geburtsdatum &&

lhs.positionImVerein == rhs.positionImVerein &&

lhs.mannschaft == rhs.mannschaft

}

// Hashable-Implementierung

**func** hash(into hasher: **inout** Hasher) {

hasher.combine(id)

hasher.combine(name)

hasher.combine(vorname)

hasher.combine(abteilung)

hasher.combine(vereinID)

hasher.combine(kontaktTelefon)

hasher.combine(kontaktEmail)

hasher.combine(adresse)

hasher.combine(clients)

hasher.combine(profilbildURL)

hasher.combine(geburtsdatum)

hasher.combine(positionImVerein)

hasher.combine(mannschaft)

}

}

**struct** Contract: Identifiable, Codable {

@DocumentID **var** id: String?

**var** clientID: String?

**var** vereinID: String?

**var** startDatum: Date

**var** endDatum: Date?

**var** gehalt: Double?

**var** vertragsdetails: String?

}

**struct** Transfer: Identifiable, Codable {

@DocumentID **var** id: String?

**var** clientID: String?

**var** vonVereinID: String?

**var** zuVereinID: String?

**var** datum: Date

**var** ablösesumme: Double? // Umbenannt von gebuehr

**var** isAblösefrei: Bool // Neues Feld für ablösefrei

**var** transferdetails: String?

**init**(

id: String? = **nil**,

clientID: String? = **nil**,

vonVereinID: String? = **nil**,

zuVereinID: String? = **nil**,

datum: Date = Date(),

ablösesumme: Double? = **nil**,

isAblösefrei: Bool = **false**,

transferdetails: String? = **nil**

) {

**self**.id = id

**self**.clientID = clientID

**self**.vonVereinID = vonVereinID

**self**.zuVereinID = zuVereinID

**self**.datum = datum

**self**.ablösesumme = ablösesumme

**self**.isAblösefrei = isAblösefrei

**self**.transferdetails = transferdetails

}

}

**struct** Club: Identifiable, Codable, Hashable {

@DocumentID **var** id: String?

**var** name: String

**var** league: String?

**var** abteilung: String?

**var** kontaktTelefon: String?

**var** kontaktEmail: String?

**var** adresse: String?

**var** clients: [String]?

**var** land: String?

**var** memberCount: Int?

**var** founded: String?

**var** logoURL: String?

// Equatable-Implementierung für Hashable

**static** **func** == (lhs: Club, rhs: Club) -> Bool {

**return** lhs.id == rhs.id &&

lhs.name == rhs.name &&

lhs.league == rhs.league &&

lhs.abteilung == rhs.abteilung &&

lhs.kontaktTelefon == rhs.kontaktTelefon &&

lhs.kontaktEmail == rhs.kontaktEmail &&

lhs.adresse == rhs.adresse &&

lhs.clients == rhs.clients &&

lhs.land == rhs.land &&

lhs.memberCount == rhs.memberCount &&

lhs.founded == rhs.founded &&

lhs.logoURL == rhs.logoURL

}

// Hashable-Implementierung

**func** hash(into hasher: **inout** Hasher) {

hasher.combine(id)

hasher.combine(name)

hasher.combine(league)

hasher.combine(abteilung)

hasher.combine(kontaktTelefon)

hasher.combine(kontaktEmail)

hasher.combine(adresse)

hasher.combine(clients)

hasher.combine(land)

hasher.combine(memberCount)

hasher.combine(founded)

hasher.combine(logoURL)

}

}

**struct** Match: Identifiable, Codable {

@DocumentID **var** id: String?

**var** heimVereinID: String?

**var** gastVereinID: String?

**var** datum: Date

**var** ergebnis: String?

**var** stadion: String?

}

**struct** Sponsor: Identifiable, Codable {

@DocumentID **var** id: String?

**var** name: String

**var** kontaktTelefon: String?

**var** kontaktEmail: String?

**var** adresse: String?

**var** gesponsorteVereine: [String]?

}

**struct** Activity: Identifiable, Codable {

@DocumentID **var** id: String?

**var** clientID: String

**var** description: String

**var** timestamp: Date

**var** category: String?

**var** comments: [String]?

**enum** CodingKeys: String, CodingKey {

**case** id

**case** clientID

**case** description

**case** timestamp

**case** category

**case** comments

}

// Benutzerdefinierter Initialisierer

**init**(id: String? = **nil**, clientID: String, description: String, timestamp: Date, category: String? = **nil**, comments: [String]? = **nil**) {

**self**.id = id

**self**.clientID = clientID

**self**.description = description

**self**.timestamp = timestamp

**self**.category = category

**self**.comments = comments

}

}

// Repräsentiert eine einzelne Nachricht

**struct** Message: Identifiable, Codable {

@DocumentID **var** id: String?

**var** senderID: String // UID des Absenders

**var** senderEmail: String // Für Anzeigezwecke

**var** content: String // Text der Nachricht

**var** timestamp: Date // Wann die Nachricht gesendet wurde

}

// Repräsentiert eine Konversation

**struct** Chat: Identifiable, Codable {

@DocumentID **var** id: String?

**var** participantIDs: [String] // Liste der Benutzer-IDs in der Konversation

**var** lastMessage: String? // Text der letzten Nachricht (für Vorschau)

**var** lastMessageTimestamp: Date? // Zeitstempel der letzten Nachricht

}