CLASS DIAGRAM
for
Soccer Live

Version: 1.0
Prepared by:

- Usman Awan (22F-3378)
- Muhammad Houd (22F-3376)
- Mutassim Billa (22F-8772)

Organization: FAST NUCES CFD

Date: 8th October, 2024

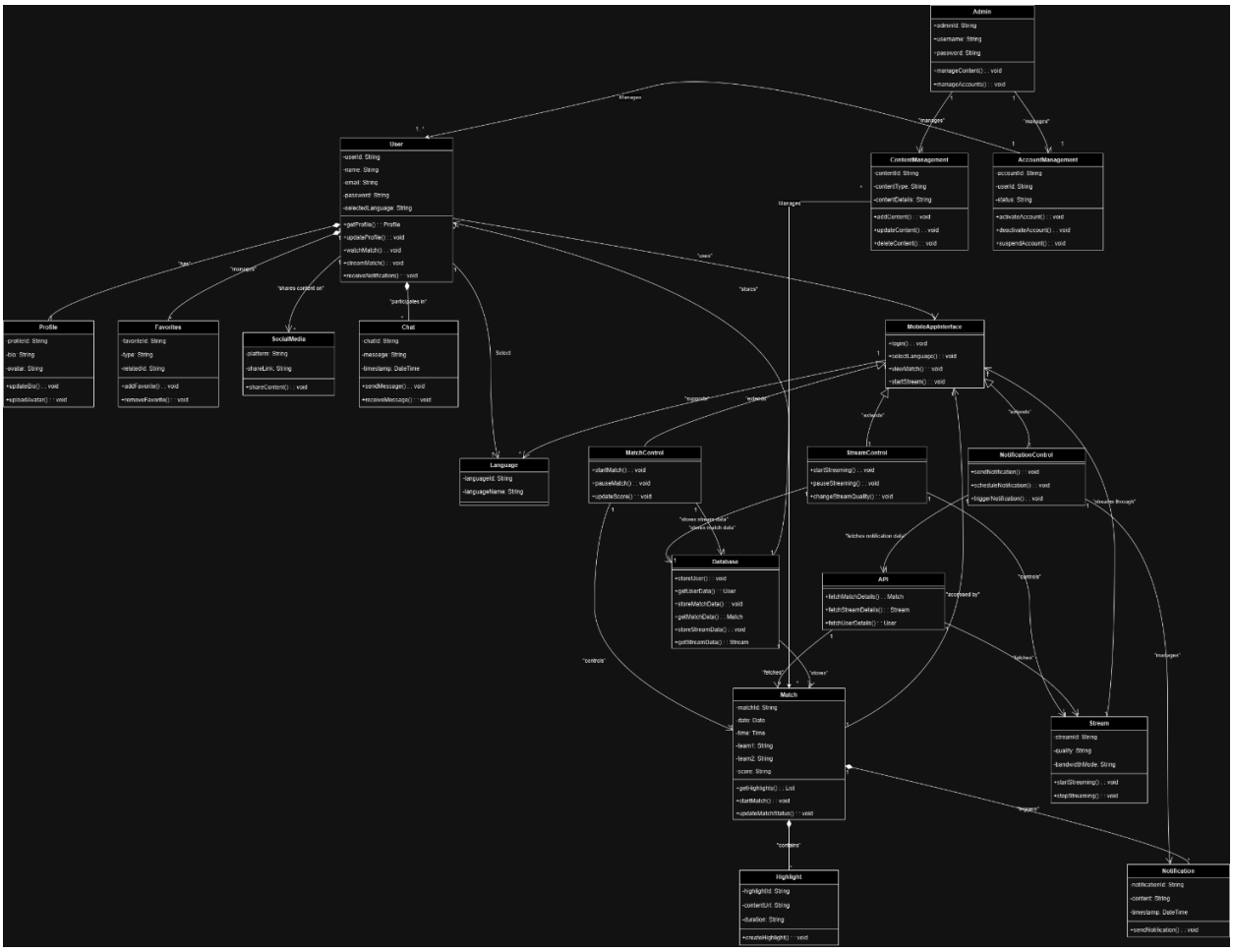
1. **Logical Models (Class Diagram):**

Here are some assumptions based on the provided class diagram. These will help to define the system's behavior and clarify the relationships and responsibilities of each class:

1. **User Registration and Profile Management**:
    - Users are required to register with unique userId, email, and password to access the platform.
    - Each user has a Profile containing information like bio and avatar, which they can manage independently.
    - Users select their preferred Language upon registration or in their settings, which customizes the interface language.
2. **Administrator Roles**:
    - Administrators have unique privileges, allowing them to manage system content, user accounts, matches.
    - Administrators can create, update, and delete content in ContentManagement,
    - Administrators are also responsible for activating, deactivating, or modifying user accounts, ensuring proper access control and account security.
3. **Match Streaming and Quality Control**:
    - Users can stream live Match events through the system's streaming service, which offers different quality options (HD, SD, etc.) based on bandwidthMode.
    - The StreamControl class handles the logic for stream quality adjustments, such as reducing quality in low-bandwidth mode, to optimize user experience.
4. **Notifications**:
    - Notifications are generated for various events (e.g., match updates, upcoming matches, administrative announcements).
    - Users receive notifications related to matches they follow or important system updates. These notifications are stored with timestamps and can be marked as read or dismissed by users.
5. **Favorites and Highlights**:
    - Users can add matches, teams, or highlights to their Favorites for quick access, which are specific to the user and do not affect other users.
    - Each Match can contain multiple Highlights (e.g., goal replays), which users can view separately from the live stream.

6. **Data Storage and API Integration**:
    - A Database class is responsible for storing all user, match, and streaming data, ensuring data persistence and accessibility.
    - An external API class may be used to fetch data about matches or live statistics, extending the system's capabilities with third-party integrations.
7. **Language and Multi-Regional Support**:
    - The system supports multiple languages, which can be selected by users to enhance accessibility.
8. **Interface and Control Classes**:
    - The MobileAppInterface serves as the primary user interface, facilitating access to core features like match streaming, chat, and notifications.
    - Control classes (StreamControl, MatchControl, NotificationControl) handle the business logic, managing how users interact with matches, streams, and notifications.

These assumptions form the basis for system functionality, explaining how various classes and entities will interact and operate within the platform.


Admin
+adminId: String
+username: String
-password: String
+manageContent() : void
+manageAccounts() : void
"manages"
"manages"
"manages"
User
+userId: String
+name: String
-email: String
-password: String
-selectedLanguage: String
+getProfile() : Profile
+updateProfile() : void
+watchMatch() : void
+streamMatch() : void
+receiveNotification() : void
ContentManagement
-contentId: String
-contentType: String
-contentDetails: String
+addContent() : void
+updateContent() : void
+deleteContent() : void
AccountManagement
-accountId: String
-userId: String
-status: String
+activateAccount() : void
+deactivateAccount() : void
+suspendAccount() : void
"has"
"manages"
"shares content on"
"participates in"
"uses"
"stores"
Profile
-profileId: String
-bio: String
-avatar: String
+updateBio() : void
+uploadAvatar() : void
Favorites
-favoriteId: String
-type: String
-relatedId: String
+addFavorite() : void
+removeFavorite() : void
SocialMedia
-platform: String
-shareLink: String
+shareContent() : void
Chat
-chatId: String
-message: String
-timestamp: DateTime
+sendMessage() : void
+receiveMessage() : void
Select
MobileAppInterface
+login() : void
+selectLanguage() : void
+viewMatch() : void
+startStream() : void
"supports"
"interacts"
"accesses"
"accesses"
Language
-languageId: String
-languageName: String
MatchControl
+startMatch() : void
-pauseMatch() : void
+updateScore() : void
StreamControl
+startStreaming() : void
+pauseStreaming() : void
+changeStreamQuality() : void
NotificationControl
+sendNotification() : void
+scheduleNotification() : void
-triggerNotification() : void
"stores stream data"
"queries match data"
"fetches notification data"
"sends through"
Database
+storeUser() : void
+getUserData() : User
-storeMatchData() : void
-getMatchData() : Match
+storeStreamData() : void
+getStreamData() : Stream
API
+fetchMatchDetails() : Match
+fetchStreamDetails() : Stream
+fetchUserDetails() : User
"accessed by"
"controls"
"controls"
"fetches()"
"shows"
"displays"
"manages"
Match
-matchId: String
-date: Date
-time: Time
-team1: String
-team2: String
-score: String
+getHighlights() : List
+startMatch() : void
+updateMatchStatus() : void
Stream
-streamId: String
-quality: String
-bandwidthMode: String
+startStreaming() : void
+stopStreaming() : void
"triggers"
"contains"
Highlight
-highlightId: String
-contentUrl: String
-duration: String
+createHighlight() : void
Notification
-notificationId: String
-content: String
-timestamp: DateTime
+sendNotification() : void