MINOR PROJECT: DIGITAL VIDEO LIBRARY MANAGEMENT SYSTEM

Submitted By:
Niranjan Joshi 160001026
Kanishkar J 160001028

1. **Need for digital video library management system:**
   - All videos can be found at a single place.
   - Videos can be searched efficiently because they are classified according to their categories.
   - Channels are helpful for organizing videos belonging to one user. People who want to see all the videos uploaded by the channel's owner can visit the channel. They will find all the videos at one place.
   - The playlists inside the channel are the group of videos of similar type. Refined search to get relevant videos is possible due to channels and playlists.
   - Special options like commenting on the video, liking the video, etc. enhance user experience and also provides community review.
   - People from different parts of the world can view what's happening around them.
   - Anyone can upload the video, watch the video, download it anytime, anywhere.
   - Digital platform to keep all the videos reduces the manual book-keeping, need for CD s, storage space and it also makes the availability of videos free of cost. Management is done through software reducing human errors. Also the efficiency as well was the capacity of storing the videos increases.
   - The digital management can be helpful for students because they can undertake distance learning.

2. **Users of the system:**
   - There are two types of users: guest and account holders. Account holders have certain privileges over the guests.
   - To get the extended facilities given to account holders the user must create account.

3. **View Videos:**
   - There are two types of users: guests and account holders.

- Both can view videos by searching for title or description tags.
- Tags help to search a video. They contain text which has keywords. These keywords match the input search text.

4. **Creating an account:**
   - Any person can create his/her account by signing up.
   - This includes filling details like Name, email Id, Password for his login.
   - The person can now sign in using his/her account username and password.

5. **Uploading Videos:**
   - Only account holders can upload a video. For uploading user first requires to sign in.
   - The entry of the new video will be done in the Video table. User needs to specify title of the video, description, the URL and its category.
   - User must specify the tags so that video can be searched efficiently.

6. **Removing Videos:**
   - Only users who have uploaded the video have permission to delete their video only.
   - They will have to sign in for this.

7. **Creating new channels:**
   - Any account holder can create his/her own channel where he can upload the videos.
   - A channel is a place where user can upload the videos of his/her interest. It help to organize videos all at one place.
   - To create a channel one must provide the name, description of the channel. Description includes gaming, art, documentary, comedy, informative, etc.
   - Playlists can be created within a channel to club the videos of similar types. In this way the channel becomes more organized.

8. **Subscribing to channels:**
   - Any signed in user can subscribe the channels so that he/she will be able to view all the channel's videos at one place.

9. **Liking and commenting on the video:**
   - Users can tag any video as favorite by pressing a button on the screen.
   - This system can be viewed as the review/feedback.

9. Adding tags to the video:

- Tags contain text which is used in searching the video. Many tags can be given to a single video. Related keywords should be put in the tags.

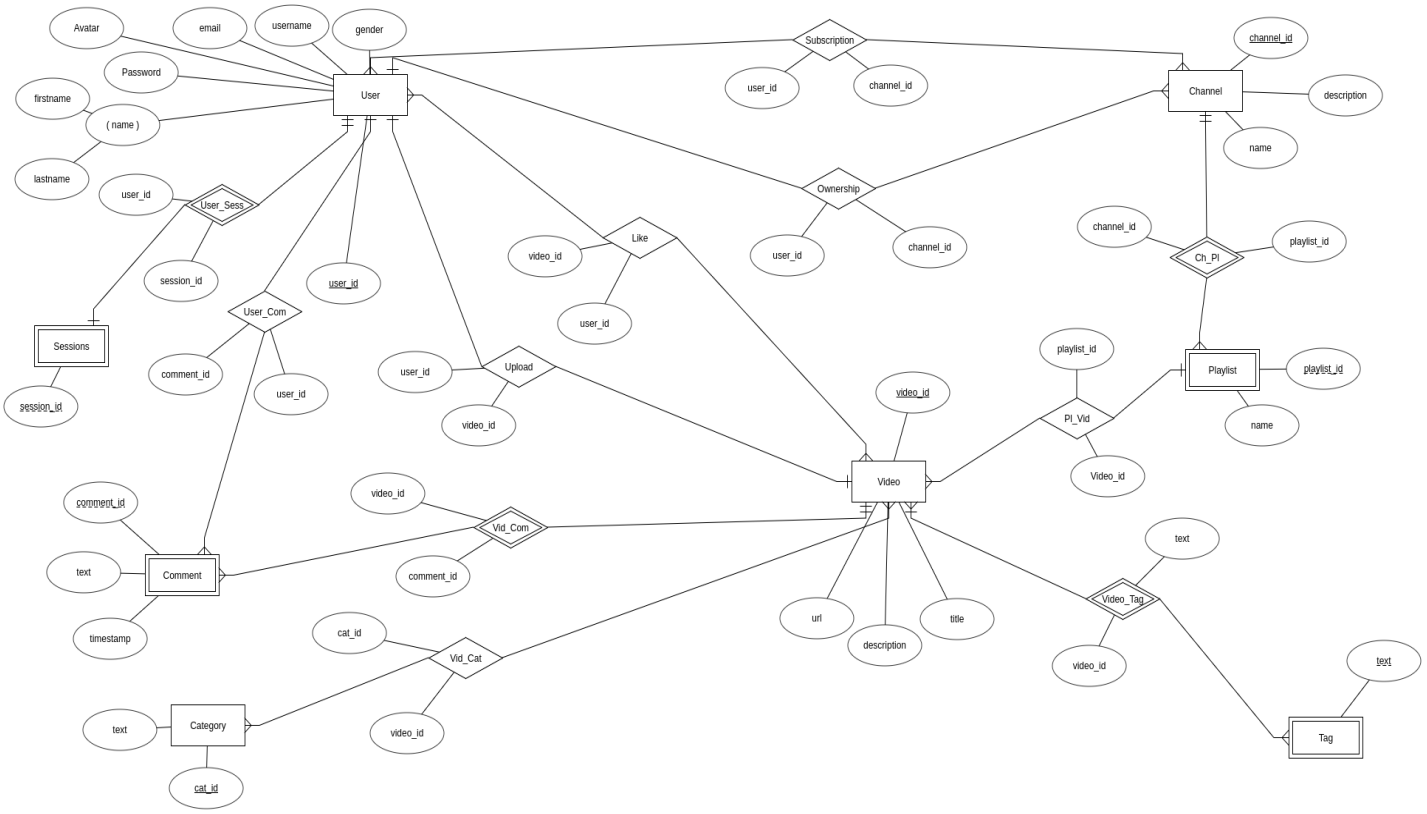
ER DIAGRAM:

ENTITY SET:

- Video
- Channel
- Category
- User
- Tag: Weak entity set Existentially Dependent on Video
- Comments: Weak entity set existentially Dependent on Video
- Playlist: Weak entity set Existentially Dependent on Channel

ENTITY RELATIONSHIPS:

- Subscription (User-Channel)
- Ownership (User-Channel)
- Upload (User_Video)
- Like (User_ Video)
- Video_Comment
- Belongs (Video_Channel)
- Video_Category
- Video_Tag
- Playlist_Video
- Playlist_Channel



FUNCTIONAL DEPENDENCIES / CONSTRAINTS:

User: (user_id) → (name, email, password)

Channel: (channel_id) → (name, description, User.user_id, User.name, User.email, User.password)

(channel_id) → (name, description)

Video: (video_id) → (title, url, description)

(url) → (title, description)

Category: (cat_id) → (name)

Tag: (video_id, text) → (text)

Comment: (video_id, Comment_id) → (text, timestamp)

Playlist: (channel_id, playlist_id) → (name)

(video_id , playlist_id) → (name)

TABLES :

User(user_id, username, email, password)

Profile(user_id, firstname, lastname, gender, avatar)

foreign key : *user_id references User.user_id*

Channel(channel_id, name, description, *user_id*)

foreign key : user_id references User.user_id

Video(video_id, url, title, description, *user_id*)

foreign key : user_id references User.user_id

Tag(text , video_id)

foreign key : *video_id references Video.video_id*

Category(cat_id, text)

Playlist(playlist_id, name, channel_id)

foreign key : channel_id references Channel.channel_id

Comment(comment_id, text, timestamp, *video_id, user_id*)

foreign key : *user_id references User.user_id*

video_id references Video.video_id

Subscription (user_id, channel_id)

foreign key : user_id references User.user_id

channel_id references Channel.channel_id

Like(video_id, user_id)

foreign key : user_id references User.user_id

video_id references Video. video_id

Pl_Vid(video_id, playlist_id)

foreign key : video_id references Video. video_id

playlist_id references Playlist.playlist_id

Vid_Cat(cat_id, video_id)

foreign key : video_id references Video.video_id

cat_id references Category.cat_id

Sessions(user_id, session_id)

foreign key : *user_id references User.user_id*