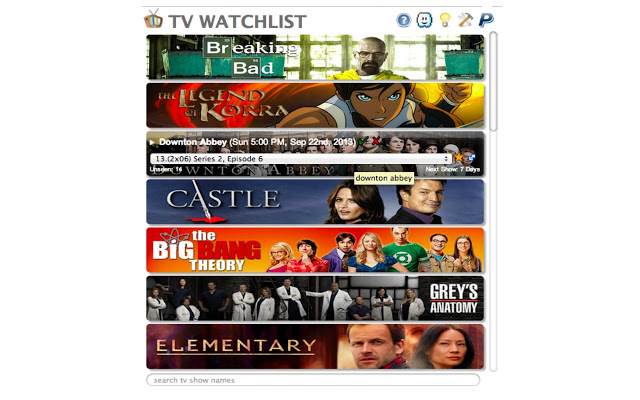
**Proposal Report-DatabaseDesign-Group16**



**Name Of Group Members**

**Özgün Özkan**

**Can Avcı**

**Emir Özbek**

**WebSite:** https://github.com/Ozgun137/CS353

**Introduction**

In our CS353 project, we are assigned to project named “TV Show Tracking System” . In this report our main purpose is to demonstrate a proposed design for our database system. In the introduction section we will provide a brief description related with our Project. Aim of this Project is to design a system in which users can track Tv shows and interact with eachother.

**USAGE OF THE SYSTEM**

In our system, users are going to have some facilities such as tracking Tvshows, rating shows and making comment on these shows . In addition users will be interacting with eachother, keep in touch with other users by making comment on their activities as well as becoming friends.

**DATABASE SYSTEM**

In our system we will have lots of datas which will be kept in our database. These datas will be kept as entity sets and and more detailed information related with tables and attributes are given in the database design section.

**REQUIREMENTS**

**Functional Requirements:**

**Login**

Before tracking the tv shows and using some facilities of the system such as commenting on other users and tvshows, user has to log in.

**Making Comments**

User can comment on either users or series.

**Rate**

Users can rate series and episodes.

**MarkWatched**

Users will be able to marked as watched.

**Adding To FriendList**

User can become friends.

**Non Functional Requirements**

**Usability:**

Usage of the system has to be clear and easy. Basic concepts which will help users to use the application has to be straightforward.

**Reliability:**

In order to improve the reliability, choosing a safe environment which provides error checking is essential.

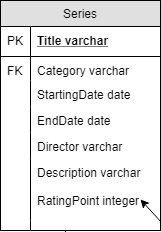
**Performance:**

System has to be coded wisely so that runtime performance is convenient.

**LIMITATIONS**

Increased number of users and datas might make us make some optimizations on our database system.

**DatabaseDesign**

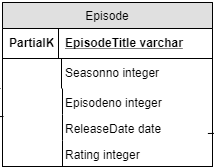


“Series entity keeps details of a tv show”

Title:Name of the show

Category: Category of tv show  
StartingDate: The fist day show was aired  
EndDate: If show has ended/canceled the date of it  
Director: Director’s name  
Description: a textfield which gives info about the Show

RatingPoint:Rates given by users to specific series.



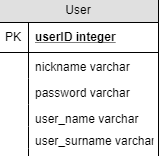
“Episode entity keeps valuable info about the eposide”

EpisodeTitle: Name of the eposide  
Seasonno: Season number of the eposide  
Eposideno: Eposide number of the eposide  
ReleaseDate: The day eposide was released  
Rating: Average of rates to episodes given by the users



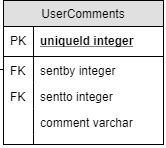
“Category is for listing tv shows by categories”

CategoryName: Name of show categories



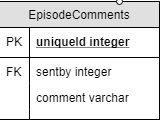
“User keeps user account details”

userID: Unique user id autoincrement  
nickname: A nickname for user which will shown in comments  
password: Password of the user  
user\_name: Name of the user  
user\_surname: Surname of the user



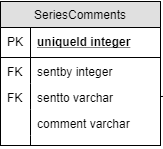
“UserComments keeps comments made to users”

uniqueID: unique comment id auto incremented  
sentby: User id who sent the comment  
sentto: Userid who recieves the comment  
comment: Comment inputed by the user



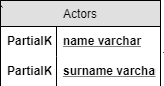
“EpisodeComments keeps comments made to eposides”

uniqueID: unique comment id auto incremented  
sentby: User id who sent the comment  
sentto: Eposide id which recieves the comment  
comment: Comment inputed by the user



“SeriesComments keeps comments made to series”

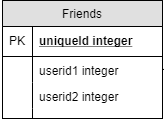
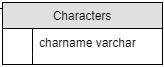
uniqueID: unique comment id auto incremented  
sentby: User id who sent the comment  
sentto: Series title which recieves the comment  
comment: Comment inputed by the user



“Actors entity keeps info about actors”

Actor\_ID: unique id of actors auto incremented  
ActorName: Name of the actor  
ActorSurname: Surname of the actor  
Birthdate: Birthday of the actor

“Characters entity keeps name of the characters playing in shows”



“Friends entity keeps friendlist who are added by friends by users”.

uniqueId:We used unique id to distinguish friendlists for each specific user.

Userid1:User who adds other user as friends

Userid2:User who is added to the friendlist



“Watched Episodes” entity keeps the episodes which are watched by users.”

UserID: We use userId attribute to distinguish watched episodes table for each specific user.



“Watched Series” entity keeps series which are watched by users.”

UserID: We use userId attribute to distinguish watched series table for each specific user.

**ER DIAGRAM OF THE SYSTEM**

