

Database Systems

CS 353

**Design Report**

Group 6

Mustafa Hakimuddin Motani

Cheol Woo Park

Waqaas Rehmani

Syed Sarjeel Yusuf

URL to the webpage:

https://cwp312.github.io/CS353/

## 

## 

**Table of Contents**

[**ER Diagram**](#_wvjtdvbef492)[**3**](#_wvjtdvbef492)

[**Relational Schemas**](#_kuoiopy87ld5)[**4**](#_kuoiopy87ld5)

[**Functional Components**](#_cwtyv91uq80r)[**6**](#_cwtyv91uq80r)

[Class Diagram:](#_lr5ypjep1u9a) [6](#_lr5ypjep1u9a)

[Driver](#_ivd0o446b28a) [6](#_ivd0o446b28a)

[UI\_Manager](#_d12mhba2istk) [6](#_d12mhba2istk)

[DB\_Manager](#_ewkp1od85p6y) [6](#_ewkp1od85p6y)

[Use Case Model](#_3pld5dy35enj) [7](#_3pld5dy35enj)

[**Table Creation SQL**](#_ch6k8rsbmhf)[**8**](#_ch6k8rsbmhf)

[Achievements](#_z1azq1qtdmx5) [8](#_z1azq1qtdmx5)

[Chats\_in](#_pzm1jxlm5p3t) [8](#_pzm1jxlm5p3t)

[Consists](#_1s6lvnf67rar) [8](#_1s6lvnf67rar)

[Friends](#_w42ghy5i3u15) [8](#_w42ghy5i3u15)

[Game](#_ptnbifcexbw1) [8](#_ptnbifcexbw1)

[Gifts](#_hu3w184tehuh) [9](#_hu3w184tehuh)

[Gives](#_kvup99gba8po) [9](#_kvup99gba8po)

[Group](#_6htm5bqrj84r) [9](#_6htm5bqrj84r)

[Has](#_2duol1c09bb7) [10](#_2duol1c09bb7)

[Login](#_7q9v229z232m) [10](#_7q9v229z232m)

[Message](#_c5eb5lfn3v8x) [10](#_c5eb5lfn3v8x)

[Payment\_info](#_yhelzmj6ewdd) [10](#_yhelzmj6ewdd)

[Sends](#_gytl67j5v9qk) [10](#_gytl67j5v9qk)

[Review](#_wsta9pe9mryo) [11](#_wsta9pe9mryo)

[Unlocks](#_44a9b5d3knpe) [11](#_44a9b5d3knpe)

[User](#_1gnxw8wi89k5) [11](#_1gnxw8wi89k5)

[Writes](#_k6idwiuqhfea) [11](#_k6idwiuqhfea)

[**User Interface**](#_j8yg4kaz5of7)[**12**](#_j8yg4kaz5of7)

[Login](#_fikd46ydbt98) [12](#_fikd46ydbt98)

[Main Screen](#_cuua1fatdyz0) [13](#_cuua1fatdyz0)

[Friend List](#_3grs3uk0rvn) [15](#_3grs3uk0rvn)

[Game Information](#_t8g89z9fpspl) [16](#_t8g89z9fpspl)

[ChatBox](#_fj1p4h8dt8tl) [17](#_fj1p4h8dt8tl)

[Game List](#_m24plwo4gxkd) [18](#_m24plwo4gxkd)

[Achievement List:](#_36k1hosv6gk7) [19](#_36k1hosv6gk7)

[Settings](#_z52my9y51myb) [21](#_z52my9y51myb)

[Group Chat](#_r86542lx2u2p) [22](#_r86542lx2u2p)

[Payment Information](#_j2hk93swfn2m) [23](#_j2hk93swfn2m)

[Create New User](#_f0lm5m7hp4ou) [24](#_f0lm5m7hp4ou)

[Gift to send](#_utzc7c9p237f) [24](#_utzc7c9p237f)

[**Advanced Database Components**](#_e7683l2b0m0q)[**25**](#_e7683l2b0m0q)

[View to get list of friends of a User](#_v2k2k44yw1i3) [25](#_v2k2k44yw1i3)

[View to get list of ratings of a Game from Reviews](#_ja26dr12hhzk) [25](#_ja26dr12hhzk)

[View to get all members of a group](#_tlle1b8fsmcc) [25](#_tlle1b8fsmcc)

[Trigger to set default prices of a game](#_rjobnqs7ij3b) [25](#_rjobnqs7ij3b)

[Trigger to set default rating when new game entered](#_r6or1039gza) [25](#_r6or1039gza)

## 

## 

## ER Diagram

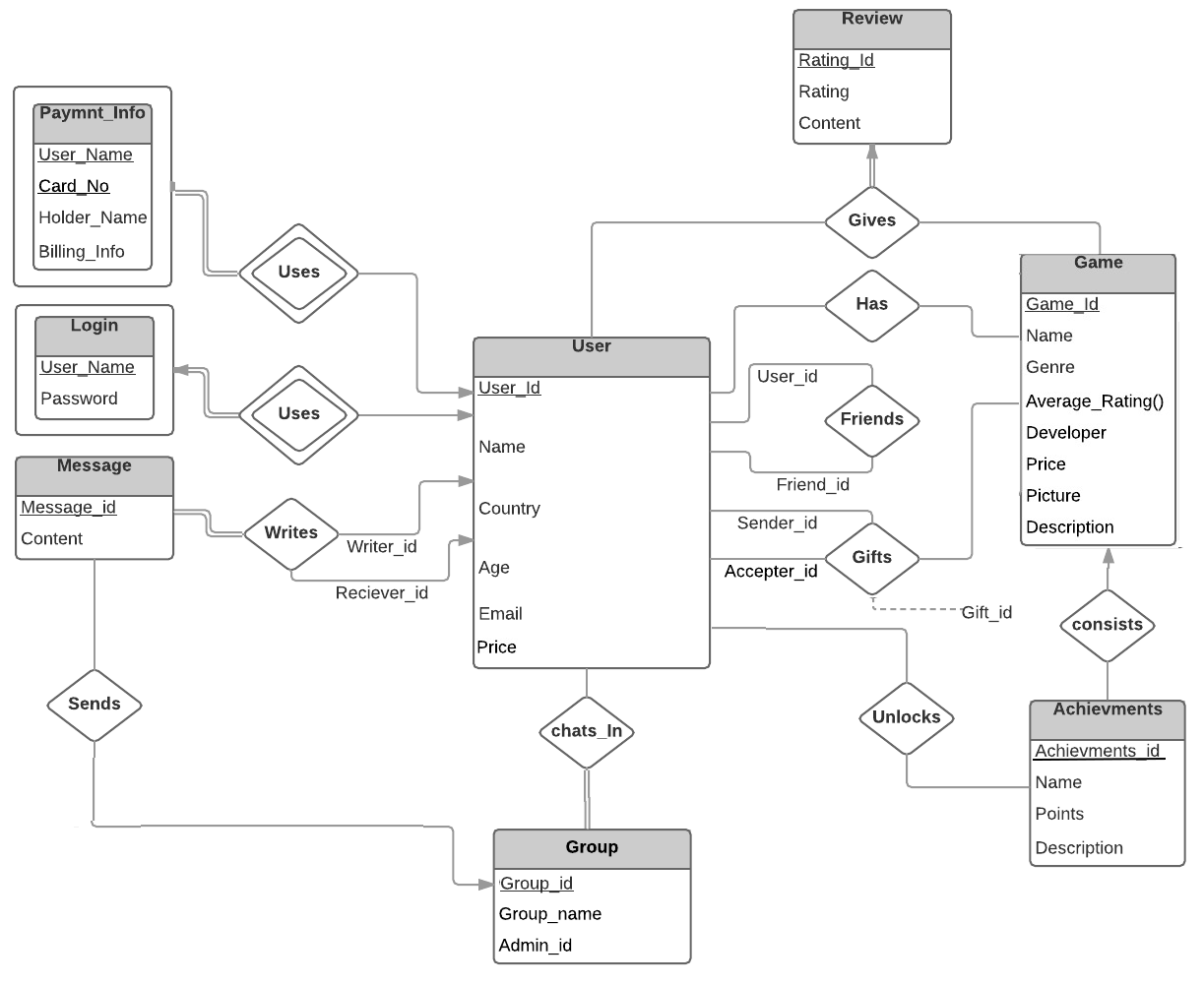


Figure 1( ER-Diagram)

## 

## 

## Relational Schemas

1. **Achievements**( achievements\_id, name, points, description)

~Domain of ‘Point ’attribute limits ranges from 0 to 10.

1. **Chats\_in**( group\_id, User\_id)
2. **Consists**( game\_id, achievement\_id)
3. **Friends**( user\_id, friend\_id)

~’user\_id’ is a foreign key referenced from ‘User’ entity

~’friend\_id’ is a foreign key referenced from ‘User’ entity the ‘user\_id’ attribute

1. **Game**( game\_id, name, genre, type, average\_rating(), developer, price, description, price, description, picture)

~Domain of genre is as listed below:

● Adventure games

● Action games

● Virtual reality games

● Sports games

● Casual games

● Indie Games

● RPG

● Strategy games

● Simulation games

● Massively Multiplayer Online games

~Average rating is a derived function as the average of all the rating pertaining to the game in the ‘Review’ entity.

1. **Gifts**( sender\_id, accepter\_id, game\_id, gift\_id)

~’sender\_id’ is a foreign key referenced from ‘User’ entity the ‘user\_id’ attribute

~’accepter\_id’ is a foreign key referenced from ‘User’ entity the ‘user\_id’ attribute

~’game\_id’ is a foreign key referenced from ‘Game’ entity

1. **Gives**( user\_id, rating\_id, game\_id)

~’user\_id’ is a foreign key referenced from ‘User’ entity

~’rating\_id’ is a foreign key referenced from ‘Review’ entity

~’game\_id’ is a foreign key referenced from ‘Game’ entity

1. **Group**( group\_id, group\_name, admin\_id)

1. **Has**( user\_id, game\_id)

~’user\_id’ is a foreign key referenced from ‘User’ entity

~’game\_id’ is a foreign key referenced from ‘Game’ entity

1. **Login**( user\_id, user\_name, password)

~’user\_id’ is a foreign key referenced from ‘User’ entity

1. **Message**( message\_id, content)
2. **Payment\_Info**( user\_id, card\_no, holder\_name, billing\_adress)

~’user\_id’ is a foreign key referenced from ‘User’ entity

1. **Sends**( message\_id, group\_id, group\_chat\_id )
2. **Review**( rating\_id, rating, content)

~Domain of ‘rating’ is from 0 to 5

1. **Unlocks**( user\_id, achievement\_id)
2. **User**( userId, name, country, birth\_date, email, picture)

~Domain of ‘country’ attribute limits options to all recognised nations.

1. **Writes**( chat\_id, writer\_id, reciever\_id)

~’writer\_id’ is a foreign key referenced from ‘User’ entity the ‘user\_id’ attribute

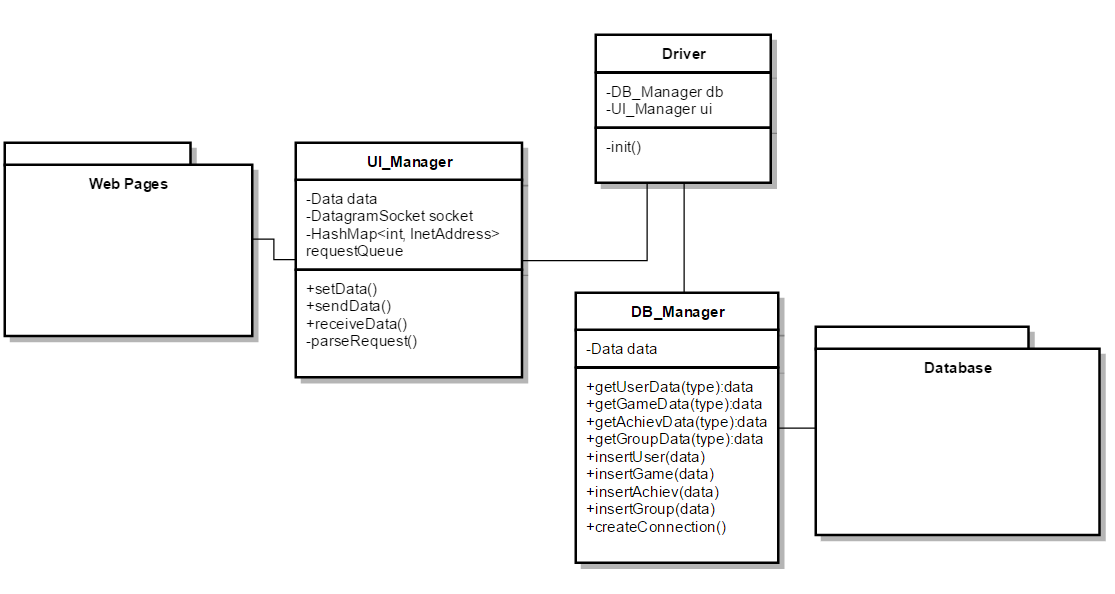
~’reciever\_id’ is a foreign key referenced from ‘User’ entity the ‘user\_id’ attribute

~’chat\_id’ is a foreign key referenced from ‘Chat’ entity the ‘user\_id’ attribute

## 

## Functional Components

### Class Diagram:



#### **Driver**

The Driver class is basically the upper layer class which contains the UI manager (web interface class) and the DB manager (database). The driver class will have ongoing loop until the program terminates which involves the program actively listening for any incoming data requests, parsing it and then sending the requested data back to the web pages.

#### **UI\_Manager**

The UI Manager class is in charge of interacting with the web pages we will setup in different language. The interactions primarily being sending/receiving data packets from them, and parsing the data into meaningful requests that the database manager can handle. This manager will contain probably the most complex data structure of the system as it is required for manager to keep track of every individual client’s request and keep track of their progress. For that, we will use a modified HashMap (as default cannot get key from value) to keep track of who requested what.

#### **DB\_Manager**

The DB Manager class, as the name implies, is in charge of anything that involves database interaction, which includes: sending queries, creating tables, deleting tables and etc.

As most of the sorting will be done while data is being fetched, there is no real need for any complex sorting algorithms.

### Use Case Model

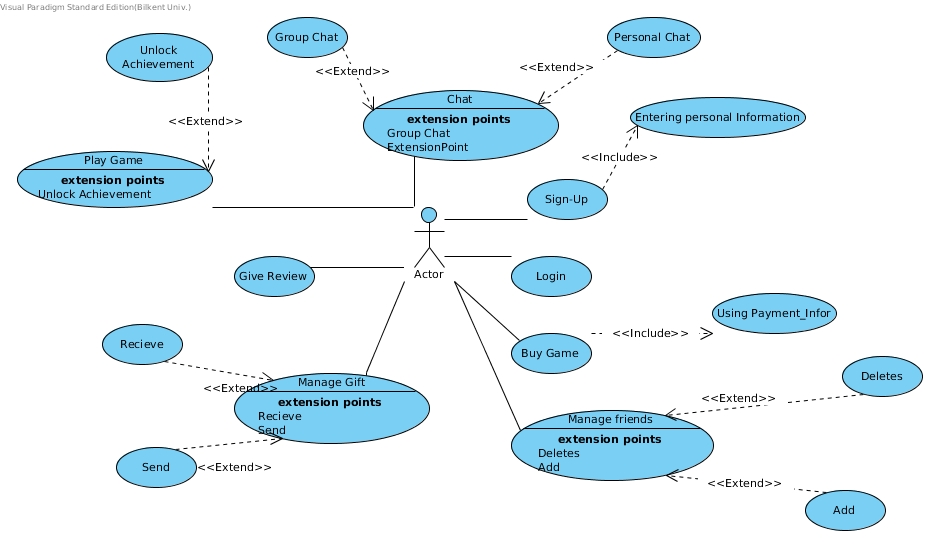


Figure 3.1: USe Case Model

## Table Creation SQL

### Achievements

Create table Achievements (

Achievements\_id int,

Name varchar(20) not null,

Points int,

Description varchar(500),

Primary key (Achievements\_id));

### Chats\_in

Create table Chats\_in (

group\_id int,

User\_id int,

Primary key (group\_id, user\_id),

Foreign key (group\_id) references Group,

Foreign key (user\_id), references User);

### Consists

Create table Consists (

Game\_id int,

Achievement\_id int,

Primary key (game\_id, achievement\_id),

Foreign key (game\_id) references Game,

Foreign key (achievement\_id) references Achievement);

### Friends

Create table Friends (

User\_id int,

Friend\_id int,

Primary key (User\_id, Friend\_id),

Foreign Key (User\_id) references User,

Foreign Key (Friend\_id) references User);

### Game

Create table Game (

Game\_id int,

Name varchar(50) not null,

Genre varchar(50),

Type varchar (20)

Average\_rating() float(3)

Developer varchar(50)

Price int,

Description varchar(500),

Picture varchar(100),/ **\* Saving picture as a URL address\*/**

Primary key (game\_id)

Check (genre in (‘Action’, ‘Adventure’, ‘Virtual Reality’, ‘Sports’, ‘Casual’, ‘Indie’,

‘RPG’, ‘Strategy’, ‘Simulation’, ‘Massively Multiplayer Online’ )

Check (type in (‘Single Player’, ‘Multiplayer’));

### Gifts

Create table Gifts (

Sender\_id int,

Accpeter\_id int,

Game\_id int,

Gift\_id int,

Primary key (sender\_id, accepter\_id, game\_id, gift\_id),

Foreign key (sender\_id) references User,

Foreign key (Accepter\_id) references User.

Foreign key (game\_id) references Game);

### Gives

Create table Gives (

User\_id int,

Rating\_id int,

Game\_id int,

Primary key (user\_id, rating\_id, game\_id),

Foreign key (user\_id) references User,

Foreign key (rating\_id) references Review,

Foreign key (game\_id) references Game);

### Group

Create table Group (

Group\_id int,

Group\_name varchar(30),

Admin\_id int not null,

Primary key (group\_id),

Foreign key (admin\_id) references User);

### Has

Create table Has (

User\_id int,

Game\_id int

Primary key (user\_id, game\_id),

Foreign Key (user\_id) references User,

Foreign key (game\_id) references Game);

### Login

Create table Login (

User\_id int,

User\_name varchar(20),

Password varchar(20),

Primary key (user\_id, user\_name),

Foreign key (user\_id) references User);

### Message

Create table Message (

Message\_id int,

Content varchar(500),

Primary key (message\_id));

### Payment\_info

Create table Paymen\_info (

User\_id int,

Card\_no char(16),

Holder\_name varchar(50),

Billing\_address varchar(200),

Primary key (user\_id, card\_no),

Foreign key (user\_id) references User);

### Sends

Create table Sends (

Message\_id int,

Group\_id int,

Group\_chat\_id int,

Primary key (message\_id, group\_id, group\_chat\_id),

Foreign key (message\_id) references Message,

Foreign key (group\_id) references Group,

Foreign key (group\_chat\_id) references Chats\_in );

### Review

Create table Review (

Rating\_id int,

Rating int,

Content varchar(500),

Primary key (rating\_id),

Check (rating > 0),

Check (rating <= 5) );

### Unlocks

Create table Unlocks (

User\_id int,

Achievements\_id int,

Primary key (user\_id, achievements\_id),

Foreign key (user\_id) references User,

Foreign key (achievements\_id) references Achievements );

### User

Create table User (

User\_id int,

Name varchar(20),

Country varchar(20),

Birth\_date char(8),

Email varchar(30),

Picture varchar(100),/ **\* Saving picture as a URL address\*/**

Primary key (user\_id) );

### Writes

Create table Writes (

Chat\_id int,

Writer\_id int,

Receiver\_id int,

Primary key (chat\_id, writer\_id, receiver\_id),

Foreign key (chat\_id) references Chats\_in,

Foreign key (writer\_id) references User,

Foreign key (receiver\_id) references User)

## User Interface

**N.B: Also the text in between the ‘<>’ brackets is simply to label what is displayed or used and will not be present on actual implementation.**

### Login

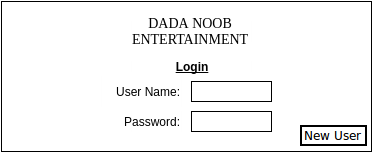


Figure 5.1: Login Page

The Login page of the system will be very simple with a simple text boxes in which the user will input his username and password.

**Tasks:**

1. Check input password and username match in database:

Select (\*)

From User

Where user\_id = “<input>” And password = “<input>”;

### Main Screen

Figure 5.2: Main ScreenMain Page.png

The main screen will be shown after the user successfully logs into the system. It can be seen that the main screen is divided into three main parts, which includes the list of games the user has in the middle, the chat bar in the right and the user details in the left. In the user details section it can be seen that the user has two buttons, one which displays a list of his friends and another which allows him to see the payment details associated to his account. In the chats bar, the user is presented all the conversations that he is participating in, in chronological order of chat that has occurred. Personal chats have a Friend List.png image whereas the other image seen indicates a group chat. Also on the right top corner of the user details section there is a settings button which will allow the user to change certain details about his account such as the user name, profile picture, country. In the game section there is a button which allows the user to see all the games available on the system, and thus buy more games that he may not currently have, which would otherwise be listed in the middle section. Also, in the chat section, there is a button with a plus sign. This button creates a new chat, and gives the option of whether it is to be a group chat or personal chat. Also if the user clicks on the game then he will start playing the game.

**Tasks**

1. Display the games of all the users:

Select G.name

From Has as H, Game as G, User as U

Where U. user\_id = H.user\_id And H.game\_id = G.game\_id;

1. Get details of user:

Select (\*)

From user

Where user\_id = “<input>”;

1. Sort games displayed by Name:

Select G.name

From Has as H, Game as G, User as U

Where U. user\_id = H.user\_id And H.game\_id = G.game\_id;

Order By G.name Desc;

Select G.name

From Has as H, Game as G, User as U

Where U. user\_id = H.user\_id And H.game\_id = G.game\_id;

Order By G.name Asc;

1. Sort games displayed by Developer:

Select G.name

From Has as H, Game as G, User as U

Where U. user\_id = H.user\_id And H.game\_id = G.game\_id;

Order By G.Developer, G.name Desc;

Select G.name

From Has as H, Game as G, User as U

Where U. user\_id = H.user\_id And H.game\_id = G.game\_id;

Order By G.name, G.Developer Asc;

1. Sort games displayed by Rating:

Select G.name

From Has as H, Game as G, User as U

Where U. user\_id = H.user\_id And H.game\_id = G.game\_id;

Order By G.name, G.rating Desc;

Select G.name

From Has as H, Game as G, User as U

Where U. user\_id = H.user\_id And H.game\_id = G.game\_id;

Order By G.name, G.rating Asc;

1. Sort games by genre:

Select G.name

From Has as H, Game as G, User as U

Where U. user\_id = H.user\_id And H.game\_id = G.game\_id;

Order By G.name, G.genre Desc

Select G.name

From Has as H, Game as G, User as U

Where U. user\_id = H.user\_id And H.game\_id = G.game\_id;

Order By G.name, G.genre Asc;

1. Search:

Select G.name

From Has as H, Game as G, User as U

Where U. user\_id = H.user\_id And H.game\_id = G.game\_id And G.name = “<input>”;

### Friend ListFriend List.png

Figure 5.3: Friend List

The friend list is displayed when the user clicks on the see friends button in the user section of the main page as seen in figure 5.2. The friend list does not display the names of the users but their user names on the system. It also provides three options for each user which includes :

Send Gift ~ Allows the user to send a gift to his friend

Chat ~ Allows the user to start a personal chat with the friend

Unfriend: ~ Deletes the friend from the user’s friend list

It must be noted that the chat bar is still present in the left of the screen.

**Tasks**

1. Display all friends of the user:

Select U.name

From Friends F, User U

Where user\_id = “<input>” and F.Friend\_id = U.user\_id

1. Unfriend**:**

Delete from Friends

Where user\_id = “<input1>” and friend\_id = “<input2>”

1. Search for friend:

Select friend\_id

From Friends

Where user\_id = ”<input1>” and friend\_id = “<input2>”

### Game Information

Game.png

Figure 5.4: Game Information

The game information page displayed if the user clicks on a game title. The game information page is split into three sections which includes the game information in the middle, the user reviews in the left and the chat bar in the right. Each review has a series of stars out of 5. The value of each rating is used to calculate the average value of the game rating which is displayed in at the bottom of the image. Moreover, the price of the game is also at the bottom and a button next to it to purchase the game. If the game is already bought then the buy button is grey with a little notification at the bottom which indicates that the player already has bought the game. Also next to the name of the game, there will be a clickable textbox which will tell the user how many of his friends have the game. The user will see a list of the friends who have the game when he clicks on the text box.

**Tasks**

1. Display information of the game:

Select (\*)

From game

Where game\_id = “<input>”;

1. Display all user reviews of the game:

Select R.(\*)

From gives as G, Review as R

Where G.game\_id = “<input>” And G.review\_id = R.review\_id;

1. **Get number of friends who play the game:**

Select count(\*)

From has As H, friends\_of As F

Where F.

1. Check if the game exists in the user’s game list:

Select (\*)

From has As H, User As U

Where U.user\_id = “<input>” And H.game\_id = “<input2>” And H.user\_id =

H.user\_id;

1. Buy game:

Insert Into Has Values (<id of user>, <id of Game>)

### ChatBox

Chat Box.png

Figure 5.5: Chat Window

**Tasks**

1. Personal chat:

Select M.Content

From Message M, Writes W, User U

Where W.Message\_id = M.Message\_id And W.Writer\_id = “<input1>” And

W.Receiver\_id = “<input2>”

### Game List

gamw list.png

Figure 5.6: Game List

The game list view shows all the games that are in the system and next to the game list are the ratings which are the average ratings derived from the ratings posted in the reviews. The games listed can be sorted using the sort option above or using the filters.

**Tasks**

1. Display all the games in database:

Select distinct name

From Game

1. Check which of the games the user has:

Select G.name

From User U, Game G, Has H

Where U.user\_id = “<input>” and H.user\_id = U.user\_id and H.game\_id =

G.game\_id

1. Sort games displayed by Name:

Select name

From Game

Order by name

1. Sort games displayed by Developer:

Select name

From Game

Order by Developer

1. Sort games displayed by Rating:

Select name

From Game

Order by Rating

1. Sort games by genre:

Select G.name

From Has as H, Game as G, User as U

Order By G.name, G.genre Desc

1. Apply Filter

Select G.name

From Game

Where **\*\*\*\*\*\* Here the attribute selected by the user from the top left combo box will be used to equal to the specific value of the attribute chosen as the filter**

e.g : Where developer = “<selected developers>”

### Achievement List:

Achievement list.png

Figure 5.7: Achievement List

Only the achievements of the games owned by the user shall be displayed in the list. They can be sorted using the sort option seen in the top right corner of the middle pane. Also filters can be used as seen by the option in the top left pane.

**Tasks**

1. Display all the achievements of the games owned by the user:

Select G.name as Game\_name, A.name as Achievement\_name

From Achievements A, Consists C, Game G, Has H

Where H.user\_id = “<input>” and H.game\_id = G.game\_id and G.game\_id =

C.game\_id and C.Achievement\_id = A.Achievement\_id

Order by G.name

1. Display all the achievements of the games owned by the user that the user hasn't unlocked:

Select G.name, Game\_name, A.name as Unlocked\_Achievement

From Achievements A, Unlocks U, Has H, Consists C, Game G

Where H.user\_id = “<input> and H.game\_id = G.game\_id and C.game\_id =

G.game\_id and U.user\_id = H.user\_id and U.achievement\_id =

A.Achievement\_id and C.Achievement\_id = A.Achievement\_id

Order by G.name

1. Display all the achievements unlocked by the user:

Select G.name as Game\_name, A.name as Achievement\_name

From Achievements A, Consists C, Game G, Has H

Where H.user\_id = “<input>” and H.game\_id = G.game\_id and G.game\_id =

C.game\_id and C.Achievement\_id = A.Achievement\_id

Order by G.name

except

Select G.name, Game\_name, A.name as Unlocked\_Achievement

From Achievements A, Unlocks U, Has H, Consists C, Game G

Where H.user\_id = “<input> and H.game\_id = G.game\_id and C.game\_id =

G.game\_id and U.user\_id = H.user\_id and U.achievement\_id =

A.Achievement\_id and C.Achievement\_id = A.Achievement\_id

Order by G.name;

### Settings

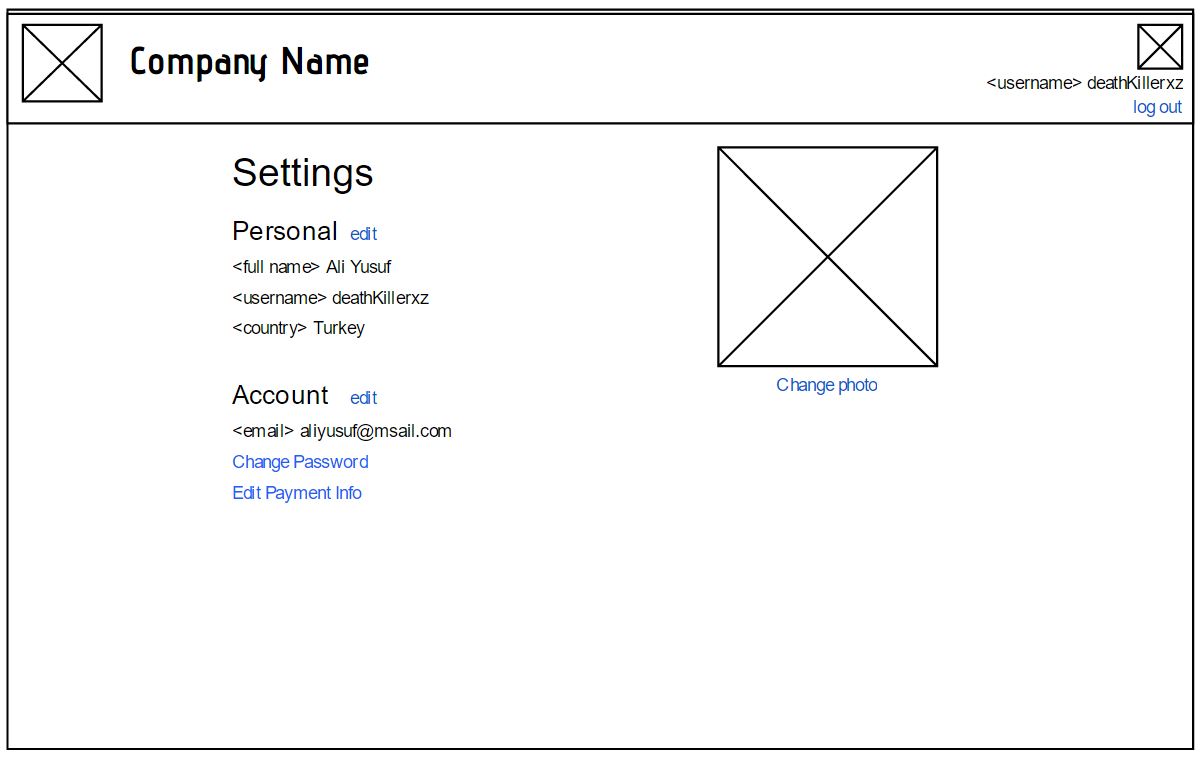


Figure 5.8: Settings Window

**Tasks:**

1. Update User Details:

Update user

Set name= <input1>

Set Country = <input2>

Set Age = <input3>

Set Email = <input4>

Where user\_id = “<input user\_id>”;

1. Update Login Info:

Update login

Set user\_name= <input1>

Set password = <input2>

Where user\_id = “<input user\_id>”;

### Group Chat

group chat window.png

Figure 5.9: Group Chat Window

The group chat window resembles the personal chat window but the only difference is that each message received has the name of the member who wrote the message.

**Tasks**

1. Users receive message in group chat:

Select M.content

From message As M, sends As S, group As G, writes As W

Where G.group\_id = S.group\_id And S.message\_id = M.message\_id, And

Writes.message\_id = M.message\_id, And reciever\_id = ( Select member\_id

From member\_of );

1. Group chat where user sends message:

Insert Into writes (message\_id, writer\_id, reciever\_id)

Where writer\_id = “<input>” And reciever\_id = ( Select member\_id

From member\_of );

**\*Member\_of explained in section 5 of report**

### Payment Information

15239394_10211456448065951_1004182320_n.png

Figure 5.10: Payment Information

**Tasks**

**1.** Add new payment details:

Insert Into Payment\_Info Values (<id of user>, <input0>, <input1>, <input3>)

**2.** Delete Payment details:

Delete from Payment\_Info

Where user\_id = <id of user>

**3.** Edit Payment details:

Update payment\_info

Set holder\_name = <input1>

Set Billing\_info = <input2>

Where card\_no = <input card number> And user\_id = <id of the user>;

**4.** View all payment details that belong to a user:

Select(\*)

From payment\_info

Where user\_id = “<input>”

### Create New User

**Tasks**

1. Insert new user into system:

Insert Into User Values (<id of user>, <name of user>, <country>, <age>, <email>);

### Gift to send

**Tasks**

1. Send a friend a gift:

Insert Into Gifts Values (<id of user>, <id of friend>, <game\_id>, <gift\_id>);

## 

## Advanced Database Components

### View to get list of friends of a User

Create view as Friends\_of ( firend\_id) as (

Select U.name

From Friends F, User U

Where user\_id = “<input>” and F.Friend\_id = U.user\_id)

### View to get list of ratings of a Game from Reviews

Create view as Ratings\_of as (

Select R. rating

From review As R, Gives as G

Where G.game\_id = “<input>” And R.game\_id = G.game\_id And R.review\_id = G.review\_id)

### View to get all members of a group

Create view as Members\_of (member\_id) as (

Select R. User\_id

From Chats\_in as R, Group as G

Where G.group\_id = R.group\_id And G.group\_id = “<input>”)

### Trigger to set default prices of a game

create trigger setdefaultPrice\_trigger before insert on Game

referencing new row as nrow

for each row

when (nrow.price = null)

begin atomic

set nrow.price = 0;

End;

### Trigger to set default rating when new game entered

create trigger setdefaultRating\_trigger before insert on Game

referencing new row as nrow

for each row

when (nrow.Average\_rating() = null)

begin atomic

set nrow.Average\_rating() = 0;

End;