Bilkent University

2017-2018 Spring Semester

CS 353 Database Systems Final Report

BilPlay - Group 22

Furkan Bacak

Enes Emre Erdem

Ikbal Kazar

Samil Iraz

https://github.com/ikbalkazar/CS353-Group-22

CONTENTS

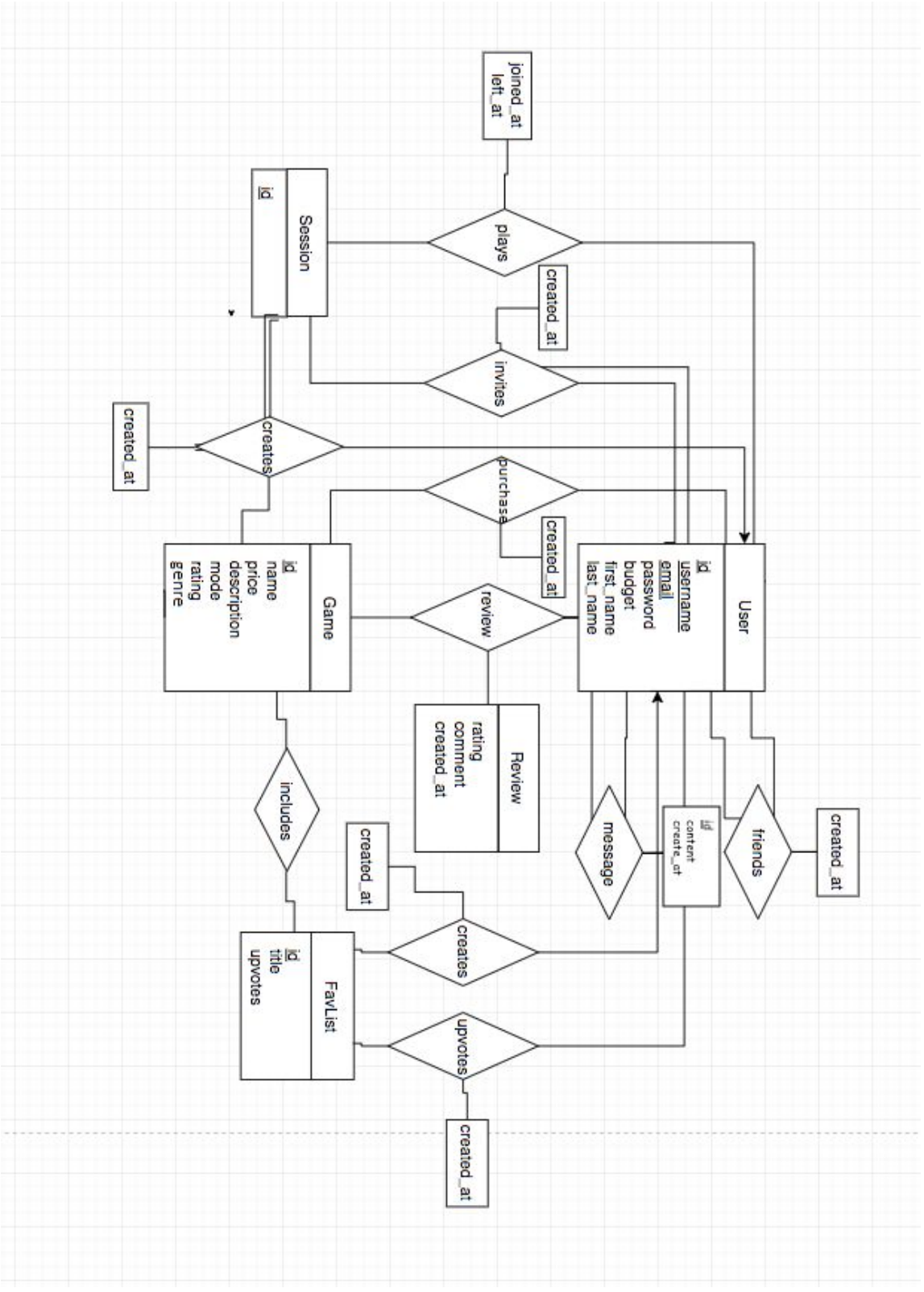
1. Description of Application

BilPlay is an ultimate gaming platform where users can buy, play and discuss games. With many games from different genres, BilPlay is appealing to every user. To provide security, users have to have an account in order to be part of BilPlay. When a user signs up to BilPlay, s/he can login to his/her account anytime, using the unique username and a password. Once logged in, users can buy games from the store with enough money in their account. If they do not have enough budget, they can add funds to their BilPlay wallet. Another feature of BilPlay is that, it provides users a social network like platform in which users can add other users as their friends. Besides playing games, users can chat with their friends live. Furthermore, in order to keep track of users in the database properly, usernames have to be unique to every user as mentioned above. However, users are able to add their first name, last name to their profile or whatever they like, so that their friends can recognize them as who they are.

BilPlay is a gaming platform as much as it is a social network. The main utility of BilPlay is gaming. It has a big game inventory which users choose whichever the game they want and add it to their own library. Users can find games for every taste by defining their criteria. Store will display many games that are fit to the criteria given. Are you not sure about whether you should buy a game or not? In such a case, one does not simply trust on reviews of random people which you don't even know whether they played the game before or not. Don't worry. We know that a gamer's taste is quite different that the common so called "authorities". You can find many reviews written by other games who has that game. You can take a look at the reviews and have an idea about whether it is nice game worth buying or not.

What would be different from using BilPlay than buying games separately if there would be no multiplayer? Users can create game sessions to play certain games with their friends. This is indeed what a social gaming platform is !

2. Final E/R Model



3. Relation Schemas

1. **user**

Relational Model:

user(id, username, email, password, budget, first_name, last_name, is_admin, banned_by)

FK: banned_by ref. user(id)

2. **friend**

Relational Model:

friend(user_id, friend_id, created_at)

FK: user_id ref. user(id)

FK: friend_id ref. user(id)

3. **invite**

Relational Model:

invite(id, user_id, friend_id, session_id, created_at)

FK: user_id ref. user(id)

FK: friend_id ref. user(id)

FK: session_id ref. session(id)

4. **user_session**

   Relational Model:

   user_session(user_id, session_id, joined_at, left_at)

   FK: user_id ref. user(id)

   FK: session_id ref. session(id)

5. **game**

   Relational Model:

   game(id, name, price, mode, description, rating)

6. **session**

   Relational Model:

   session(id, creation_time, creator_id, game_id)

   FK: creator_id ref. user(id)

   FK: game_id ref. game(id)

7. **genre**

   Relational Model:

   genre(name)

8. **game_genre**

   Relational Model:

   game_genre(game_id, genre_name)

FK: game_id ref. game(id)

FK: genre_name ref. genre(id)

9. **purchase**

   Relational Model:

   purchase(<u>user_id</u>, <u>game_id</u>)

   FK: user_id ref. user(id)

   FK: game_id ref. game(id)

10. **review**

    Relational Model:

    review(<u>user_id</u>, <u>game_id</u>, rating, comment, created_at)

    FK: user_id ref. user(id)

    FK: game_id ref. game(id)

11. **message**

    Relational Model:

    message(<u>id</u>, user_id, friend_id, content, created_at)

    FK: user_id ref. user(id)

    FK: friend_id ref. user(id)

12. **favlist**

    Relational Model:

    favlist(<u>id</u>, creator_id, title, created_at)

    FK: creator_id ref. user(id)

13. **favlist_upvote**

Relational Model:

favlist_upvote(user_id, favlist_id, created_at)

FK: user_id ref. user(id)

FK: favlist_id ref. favlist(id)

14. **favlist_game**

Relational Model:

favlist_game(favlist_id, game_id)

FK: favlist_id ref. favlist(id)

FK: game_id ref. game(id)

4. Implementation Details

We are using MySQL to implement our designed database with InnoDB engine. We implemented a REST API in Java and connected to database from our Java code with JDBC interface which provides prepared sql statements and connection to MySQL. We are using the Dropwizard framework which includes Jersey for REST endpoints and JDBI-3 to interact with MYSQL. We used two Docker containers with Docker-Compose, one for the Dropwizard application and one for the MYSQL server. Containerized environment allows us to replicate the development environment in different computers easily. It also makes deploying the application to cloud convenient. We used a github public repository for code revision maintenance. For our front-end views we are using Apache Freemaker html templates.

There was a compatibility issue with java-mysql-connector MYSQL driver and MYSQL version 8, we tracked down the issue and found out that it was an open bug. Hence we decided to downgrade our mysql version to 5.

When we were designing the chat feature, we first decided to use web-sockets to push received messages to the client browser. However, dropwizard didn't have built-in support for web-sockets hence we decided to reload the chat page periodically instead. We wrote some javascript to check if user was writing a message and delayed the refresh to make sure we didn't reloaded the page while user in the middle of writing a message.

5. Advanced DB Features

5.1 Reports

- **Number of reviews of each game**

```
SELECT name AS game_name, count(*) AS number_reviews
    FROM game NATURAL JOIN review
    GROUP BY name;
```

game_name	number_reviews
DOTA	3
CS GO	2
HEARTHSTONE	5

- **Top 10 Games according to number of players**

```
SELECT name AS game_name, count(*) AS number_players
    FROM game g JOIN purchase p ON g.game_id = p.game_id
    GROUP BY name
    ORDER BY number_players DESC LIMIT 10;
```

game_name	number_players
DOTA	43

CS GO	22
HEARTHSTONE	13

- **Budget of users in decreasing order**

  SELECT * FROM user **ORDER BY** budget **DESC**;

- **List of users in decreasing order of number of friends.**

  SELECT u.username**, COUNT(*) AS** friend_cnt **FROM** user **u JOIN** friend **f ON** u.user_id = f.user_id

  GROUP BY u.user_id

  ORDER BY friend_cnt **DESC;**

username	friend_cnt
john	7
doe	3
bob	2

5.2 Views

- **Game with rating View**

  CREATE VIEW game_with_rating **AS**

  SELECT g.*, **AVG**(r.rating) **AS** rating

  FROM game g **JOIN** review r **ON** g.game_id = r.game_id

  GROUP BY g.id;

5.3 Triggers

- **Total budget of all users**

  CREATE TRIGGER total_budget **BEFORE UPDATE ON** user

FOR EACH ROW SET @total_bud = @total_bud + NEW.budget - OLD.budget**;**

6. User's Manual

1. **Sign-Up Page:** User will sign up to application to take advantage of bilplay such as playing games with friends, having some ideas about games before purchasing by looking reviews of gaming society.
2. **Login Page:** After signing up the application, user will use the "login in" page with username and password to use application other times that s/he wants to use application.
3. **Store Page:** To take a look at the new games and their reviews or purchase new games, user will use the store page. In this page, user can search games according to some criterias such as genres, ratings, costs.
4. **Game Page:** in this page users can get informations about game such as costs. They can read the description which is added by creators of game of game and look at the reviews about game of other users who have owned the game. They also can go to purchase page to buy game from this page.
5. **Purchase Page:** In this page, users can see game's description, their balances and the cost of game. Users can purchase game in this page if they have enough money.
6. **Library Page:** Users can see their games in library page. In this page, there will be some informations such as playing time, photograph of games and also, there will be box to send review of user about game. Users can select the game from the games list in the left side of the page. To play game both singleplayer and multiplayer games, users will use the button which is located in the library page.
7. **Multiplayer Page:** By trying to play a multiplayer games, users will join the game session page from the library page. In this page, users can invite only their friends who have the game to play together. All players that joined same game session

page, can see other players. There also will be a button to leave from this game session page. This page will keep the beginning time of creation.

8. **Profile Page:** Profile page mainly provides some options for users. They can set their first name and last name as they wish. In addition they can also change their password by providing the old password and the new. Moreover, users have to come to this page when they want to add funds to their wallet.
9. **Friends Page:** In this page, users can add new friends. Adding process is one side process like twitter follow process. Users also can start a chat with their friends in this page. To start chat with another user, both sides have to be added as friend. In addition to these features, users can remove their friends.
10. **Chat Page:** After starting a chat with a friend, users are redirected to this page. In this page, users can receive and send messages live. Users can leave chat whenever they want.