Database Systems Project

CS - 353 - Section 1

Social Gaming Marketplace-Mist

Alptuğ Albayrak - 21501970
Alperen Erkek - 21501265
Barış Eymür - 21502879
Deniz Şen - 21502997

1.0 INTRODUCTION

This report aims to explain Mist's functionality and integrations. The project will heavily depend on a database system and its specifications will be underlined in this report. Additionally, description, requirements, limitations and finally entity-relation diagram will be given in this report.

Project description will be underlined first. It aims to explain goal of the project and its purpose. Then whether we need a database or not will be questioned and how we can utilize a database for our best interest will be explained.

Later on, Mist's requirements will be examined. We'll investigate this aspect under 3 different categories-that is: functional, non-functional and pseudo requirements. Functional requirements will explain the capabilities of user, scope of the system and properties of it. Non-functional requirements will explain criteria that are essential to a stable, responsive and secure program. In pseudo, requirements technical aspects of the project will be given.

In limitations, we will be talking about what system and user can't do. In other words, we will be talking about boundaries of the system. Finally, an Entity-Relation diagram will be given. This diagram is designed for our requirements.

2.0 PROJECT DESCRIPTION

Mist is a web-based online application for buying video games and interacting with other video gamers around the world. In this application, the users will be able to choose the videos games which are appropriate for them by checking different categories of games and getting information about them from store. If a user likes a specific game and decides to purchase it from store, then that game will be added to the list of games which are owned by the user called "Library" and the cost of the game will be withdrawn from the balance of the user in the system. After this, the user will be available to play the game.

The goal of this system is to create a social platform for the gamers. There will be discussion platforms about every game and in these platforms, the gamers will be able to share their opinions and critics with other gamers, even with the developer of the game. Other gamers will able to respond to this message and even upvote or downvote them. Moderation of this platform will be handled by admins that will be selected by developers. It's necessary that users share their experience so that platform can grow as a social place.

Mist will be supported by a friendship system. This means that each user will have a unique friend list. They will be able to broaden their friend list by searching their friends in the system and adding them. Of course once a user adds another one, they will not establish friendship immediately. This is for security reasons. An invitation will be send to the receiver and if the receiver user agrees, their friendship will be established. Finally, users will be able

to see what games their friends are playing and how much hours did they play. Additionally, if a user plays a multiplayer game, they will be able to send invitation to the their friends.

The users will have activities-which are social and gaming. A social activity can be a friendship between two distinct users. Similarly, a gaming activity can be purchase of a new game by a user. Each user will have unique activity list which is consisted of activities of their friends. These activities and activity lists have crucial impact on the sense of achievement users can feel.

2.1 Why Do We Need a Database for Mist?

Mist will depend on multiple entities that has big pile of data. This data will be accessed and modified frequently. For example, number of users will be very high and each user will have properties like friend list, games that differs for each user. This amount of data can not be managed by a normal system. Therefore we need a system that does this job effectively and efficiently.

2.2 How Do We Use a Database for Mist?

Entities that has big pile of data-such as “Users”, “Store”, “Friend List” etc.- will be stored in our database. By this way, we will be able to reach our data and modify them if needed in fast manner. It is not just speed that we will require from our database system. Management of this huge set of data will be really easy and safe to use as well. Additionally, they might be storing sensitive information that needs to be backed up.

3.0 FUNCTIONAL REQUIREMENTS

3.1 User

- Every user should login to the system using a username and a password.
- Users should be able to search other people and add them as friends.
- Users can change their passwords.
- Each user’s friend list and activity list should be unique. User’s friends should be able see their recent activities.
- Users can search and buy games from the store.
- Users can post to the forum, they can read and reply to already existing posts from other users.
- A user can be a regular user, an admin or a developer.

3.2 Admin

- The admin can manipulate the posts in the forum by editing and deleting them.
- The admin can ban a user from the forum and unban an already banned user.

3.3 Developer

- The developer can publish games to the store.

4.0 NONFUNCTIONAL REQUIREMENTS

4.1 Authentication & Security

- In order to use the system, users should login using their username and password.
- Passwords will be encrypted to protect the privacy of the users.

4.2 User Friendliness

- The system will have quick response times. The system should have a structure such that it will continue working properly when the data grows larger.
- The system should not fail. Even if a fail occurs, the recovery time should be minimal.
- The interface of the platform will be user-friendly and simple. It will help the user how to navigate through the platform.

4.3 Performance

- The system will be quick enough to respond multiple requests at a time.
- The users will not encounter stuttering issues while using the marketplace.

4.4 Scalability

- The system will be available for further innovations such as new entities, new sections, more advanced entity relations, etc.

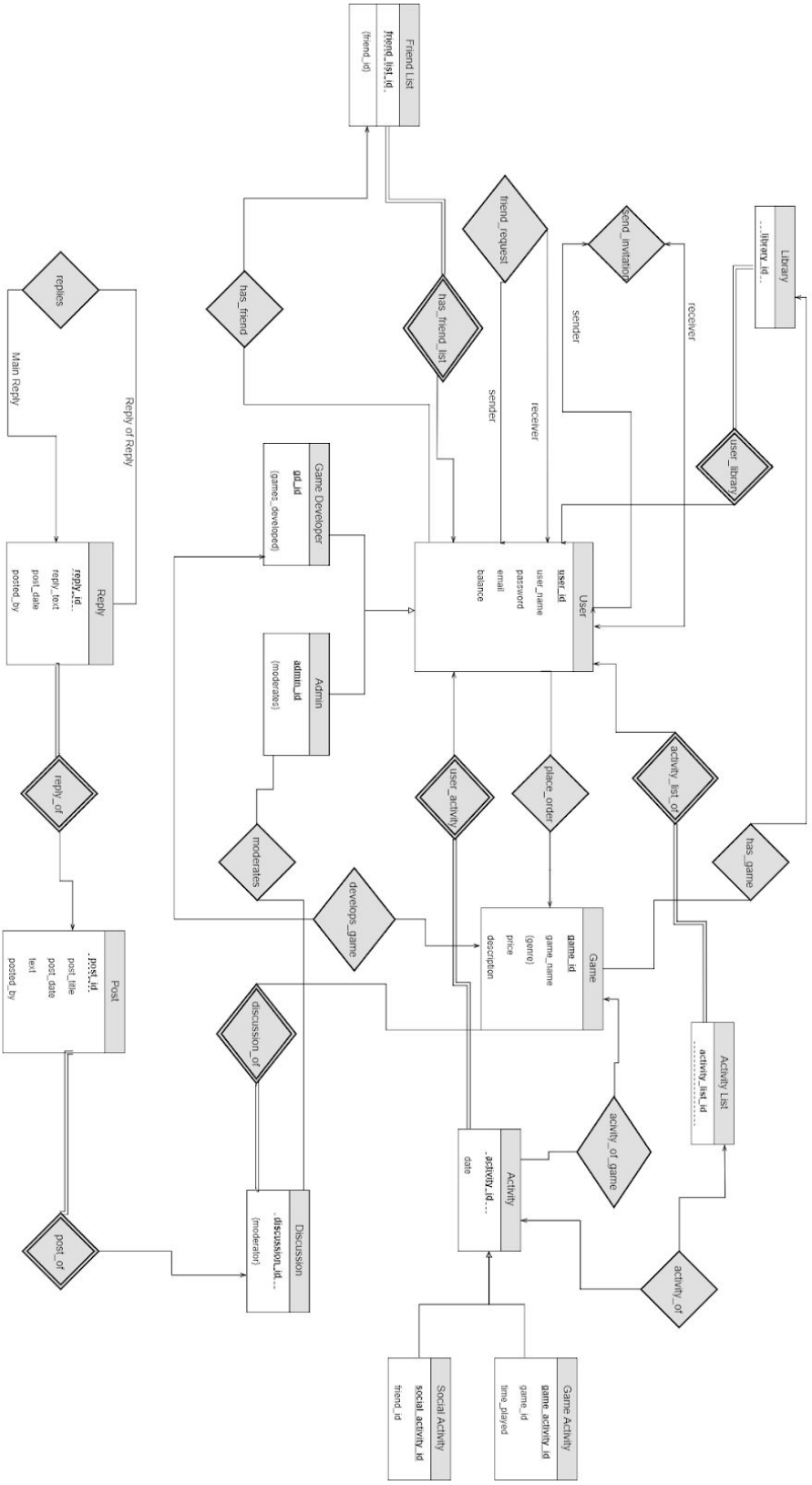
5.0 PSEUDO REQUIREMENTS

- The database will be implemented using MySQL.
- Back-end of our website will be implemented using PHP.
- HTML, CSS, JavaScript will be used for the front-end of the website.

6.0 LIMITATIONS

- A user will only be able to see the activities of his/her friends.
- If a user is banned from discussions, then he/she will not be able to send posts to any discussion.
- A user will be able to buy a specific game only once.
- Usernames and names of the games in the platform should be unique.

7.0 ENTITY-RELATIONSHIP DIAGRAM



8.0 CONCLUSION

Mist is a system that users are able to share their games in store and gamers can buy their favorite games. Users can share their thoughts in game's discussion place as comments and there will be admin(s) who will moderate these forums. Users will have friend list and they can find their friends and invite them for friendship. They will see their friends' activities and invite them to multiplayer games that they will be playing.

Description of the project is underlined in project description. Necessary of the database and how we can benefit from it has been talked in this section. Our project will have lots of entities that needs to be stored. For this reason we need a database system that will handle these requirements effectively.

This report described the requirements of the projects under three categories: functional, non-functional, pseudo. In functional requirements, scope of the system and user-end functionalities are explained. Capacity of Mist can be found here. In nonfunctional requirements we explained some of the criteria that gives user the best experience. We believe that in these criteria, the most important one is cost/time efficiency. In pseudo requirements technical details of the system are given.

Then boundaries that are mentioned in "Limitations" part can be improved in the future. Finally we provided a entity-relationship diagram that explains the structure of our database. We believe this diagram represents our requirements in the best manner.

9.0 PROJECT LINK

https://github.com/theGuiltyMan/CS-353