Semester-Long Project

(Initial Document)

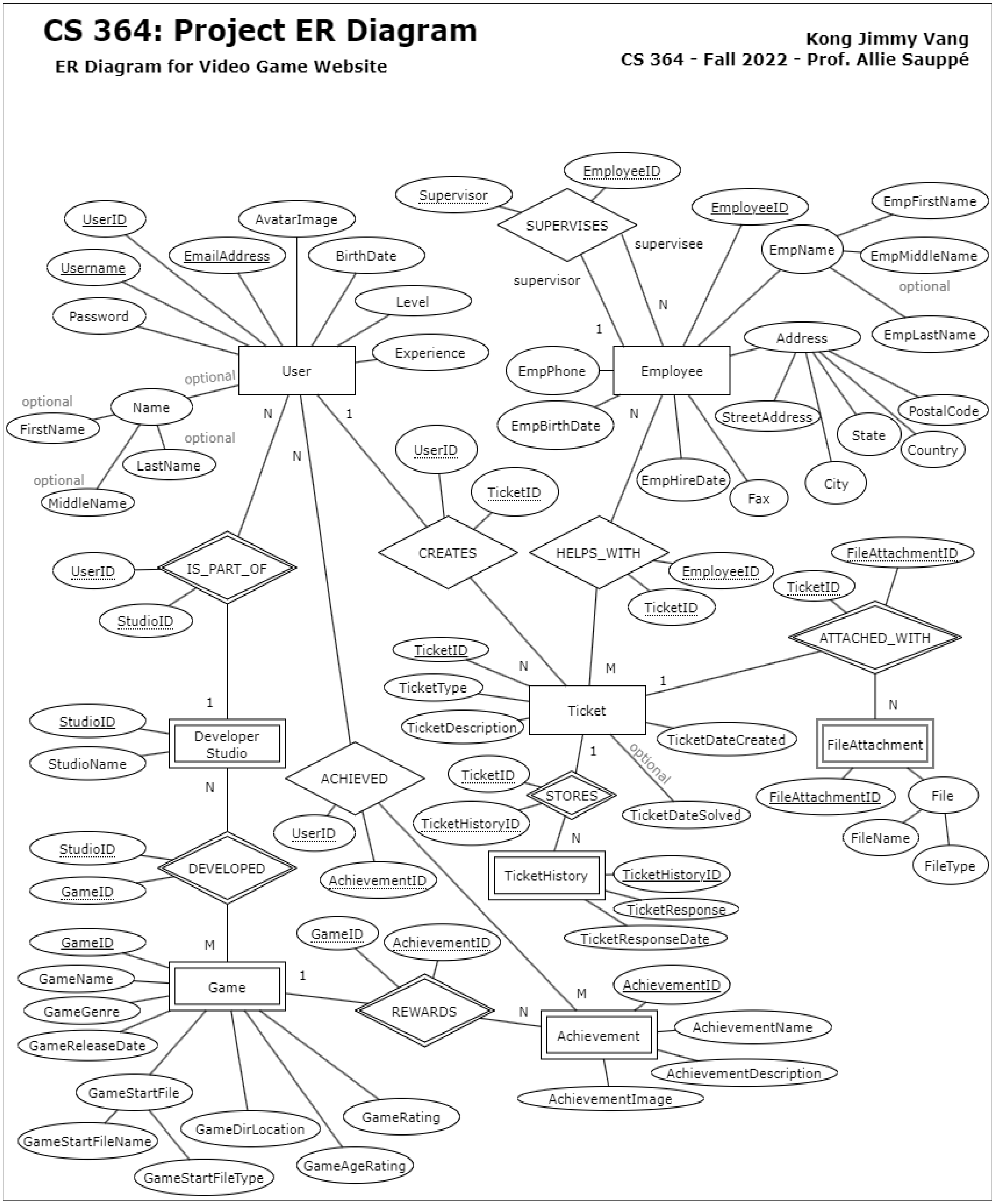
by Kong Jimmy Vang

Abstract

This project will be based on a Free-To-Play (F2P) game hosting website named Kongregate (https://www.kongregate.com/). This website allows developers to create games and upload and host their games on the website for users to play. Developers are allowed to make money on their games through monetization, but their games are required to be free to play. In the past, the site allowed uploading of SWF files or Adobe Flash games. Nowadays, many browsers have deprecated the use of Flash. Most newer games on the site now use Unity WebGL, or other forms of integrated web application software. The site does allow for backwards compatibility with older games, but they require the use of Third Party software that are similar to Adobe Flash. The site allows searching for games to play. The site shows top new games and the top rated games if the button is clicked on. At the bottom of each web application, comments are allowed. Comments can be upvoted or downvoted, they can also be replied to. Users are allowed to send messages to each other via PM or DM.

The site allows users to play without an account. They are also allowed to create an account, sign in, and play games. Users that sign in have information stored about each game they play. Users are allowed to chat with others during each game they play. Each game has rooms that players can socialize in during gameplay. Each game has it's own acheviement that the user can earn by playing the game. They are only earned if you own a user account. Overall, my project will be based on a this type of design and attempt to store data in a database relevant to running a website similar to Kongregate.

Database:


CS 364: Project ER Diagram
ER Diagram for Video Game Website
Kong Jimmy Vang
CS 364 - Fall 2022 - Prof. Allie Sauppé
User
UserID
Username
Password
EmailAddress
AvatarImage
BirthDate
Level
Experience
Name
FirstName
LastName
MiddleName
optional
optional
optional
optional
SUPERVISES
Supervisor
EmployeeID
supervisor
supervisee
1
N
Employee
EmployeeID
EmpName
EmpFirstName
EmpMiddleName
EmpLastName
optional
EmpPhone
EmpBirthDate
EmpHireDate
Fax
Address
StreetAddress
City
State
Country
PostalCode
N
1
N
N
CREATES
UserID
TicketID
HELPS_WITH
EmployeeID
TicketID
IS_PART_OF
UserID
StudioID
1
Developer Studio
StudioID
StudioName
N
DEVELOPED
StudioID
GameID
M
Game
GameID
GameName
GameGenre
GameReleaseDate
GameStartFile
GameStartFileName
GameStartFileType
GameDirLocation
GameAgeRating
GameRating
ACHIEVED
UserID
AchievementID
REWARDS
GameID
AchievementID
1
N
M
Achievement
AchievementID
AchievementName
AchievementDescription
AchievementImage
Ticket
TicketID
TicketType
TicketDescription
TicketDateCreated
TicketDateSolved
optional
N
M
1
STORES
TicketID
TicketHistoryID
1
N
TicketHistory
TicketHistoryID
TicketResponse
TicketResponseDate
ATTACHED_WITH
TicketID
FileAttachmentID
N
FileAttachment
FileAttachmentID
File
FileName
FileType

Tables/Attributes:

- **User** - This table will store user information relevant to the F2P game hosting website. Users can send a Ticket, which Employees can review. Users can also be part of a Developer Studio and develope Games together.
    - **UserID** - (**PRIMARY**) Unique User ID.
    - **Username** - (**PRIMARY**) Unique Username.
    - **EmailAddress** - (**PRIMARY**) The Email Address of the User.
    - **Password** - Login password.
    - **Name** - (**OPTIONAL COMPOSITE**) Real name of the user. Optional for privacy reasons.
        - **FirstName** - (**OPTIONAL**) Real first name. Optional for privacy reasons.
        - **MiddleName** - (**OPTIONAL**) Real middle name. Optional for privacy reasons.
        - **LastName** - (**OPTIONAL**) Real last name. Optional for privacy reasons.
    - **AvatarImage** - A jpg, png, or gif file that shows the user's avatar icon.
    - **BirthDate** - To verify if the user is 13+ years old.
    - **Level** - The level of the user from earning experience points.
    - **Experience** - The experience points earned from playing games.
    - **StudioID** - (**FOREIGN**) Studio ID.
    - **TicketID** - (**FOREIGN**) Ticket ID.
- **DeveloperStudio** - This table will store information about a developer studio team. The Developer Studio consists of multiple Users and can Develop multiple different games. Many studios can work on one game together also.
    - **StudioID** - (**PRIMARY**) Unique Studio ID.
    - **StudioName** - The Developer Studio's name.
    - **UserID** - (**FOREIGN**) User ID.
    - **GameID** - (**FOREIGN**) Game ID.
- **Game** - This table will store information about each game a Developer Studio has created. Each game will have it's own Achievements.
    - **GameID** - (**PRIMARY**) Unique Game ID.
    - **GameName** - The game's title/name.
    - **GameGenre** - The primary genre of the game.
    - **GameReleaseDate** - The date that the game was released on.
    - **GameStartFile** - (**COMPOSITE**) The game's start file.
        - **GameStartFileName** - The name of the start file.
        - **GameStartFileType** - The type (file extension) of the start file.
    - **GameDirLocation** - The server directory where the game is stored.
    - **GameAgeRating** - The ESRB age rating for a game.
    - **GameRating** - The star rating for a game (0.0 stars being the worst, 5.0 stars being the best).
    - **StudioID** - (**FOREIGN**) Studio ID.
    - **AchievementID** - (**FOREIGN**) Achievement ID.

- **Achievement** - This table will store achievements for each Game and store a many-to-many relationship from the ACHIEVED table to the User table.
  - **AchievementID** - (**PRIMARY**) Unique Achievement ID.
  - **AchievementName** - The name of the achievement.
  - **AchievementDescription** - The description of the achievement.
  - **AchievementImage** - A jpg, png, or gif file that shows what the achievement looks like.
  - **GameID** - (**FOREIGN**) Game ID.
  - **UserID** - (**FOREIGN**) User ID.
- **Ticket** - This table will store Support Ticket information related to User and Technical Support. Tickets can be responded to by Employees
  - **TicketID** - (**PRIMARY**) Unique Ticket ID.
  - **TicketType** - The subject of the ticket.
  - **TicketDescription** - The description about the subject of the ticket.
  - **TicketDateCreated** - The date the ticket was created on.
  - **TicketDateSolved** - The date the ticket was solved on.
  - **TicketHistoryID** - (**FORIEGN**) Ticket History ID.
  - **UserID** - (**FORIEGN**) User ID.
  - **EmployeeID** - (**FORIEGN**) Employee ID.
  - **FileAttachmentID** - (**FORIEGN**) File Attachment ID.
- **FileAttachment** - This table will store file attachments that are related to the Ticket table.
  - **FileAttachmentID** - (**PRIMARY**) Unique File Attachment ID.
  - **File** - (**COMPOSITE**) The attachment file.
    - **FileName** - The name of the attachment file.
    - **FileType** - The type (file extension) of the attachment file.
  - **TicketID** - (**FORIEGN**) Ticket ID.
- **Employee** - This table will store Employee information. They are mainly to help with Ticket Requests. Everything else in this table is for company use.
  - **EmployeeID** - Unique Employee ID.
  - **EmpName** - (**COMPOSITE**) Employee full name.
    - **EmpFirstName** - Employee first name.
    - **EmpMiddleName** - (**OPTIONAL**) Employee middle name.
    - **EmpLastName** - Employee last name.
  - **EmpPhone** - Employee phone number
  - **EmpBirthDate** - Employee birth date.
  - **EmpHireDate** - Employee hire date.
  - **Address** - (**COMPOSITE**) Full address of employee.
    - **StreetAddress** - Street address location.
    - **City** - City location.
    - **State** - State.
    - **Country** - Country of residence.
    - **PostalCode** - Postal code or ZIP code.
  - **Fax** - The fax number of the employee
  - **Supervisor** - (**FORIEGN**) Supervisor's ID.
- **DEVELOPED** - This table stores a many-to-many relationship between the Developer Studio table and the Game table.
  - **StudioID** - (**FORIEGN**) Studio ID.
  - **GameID** - (**FORIEGN**) Game ID.

- **ACHIEVED** - This table stores a many-to-many relationship between the User table and the Achievement table.
  - **UserID** - (**FORIEGN**) User ID.
  - **AchievementID** - (**FORIEGN**) Achievement ID.
- **HELPS_WITH** - This table stores a many-to-many relationship between the Employee table and the Ticket table. Many employees can help with and respond to many different tickets.
  - **EmployeeID** - (**FORIEGN**) Employee ID.
  - **TicketID** - (**FORIEGN**) Ticket ID.

Functionality:

Main Functions of the Application

1. Allow Developer Studios to upload games.
2. Allow Anonymous Users to play games.
3. Allow the creation of a User account to play games.
4. Allow Users to customize their Account Settings, Profile Settings, and Profile Picture.
5. Allow Developer Studios to add achievements to their games so they can be earned by User.
6. Show appropriate games to the appropriate age group based on the ESRB rating system.
7. Allow Users to level up their accounts by earning points through rating games and completing Achievements.
8. Allow Employees to modify website information.
9. Allow Customer Service agents to be contacted.
10. Allow for a User Support ticket system.

Examples of Envisioned Advanced Queries

Query 1: Find the top 10 newest games.

```
SELECT GameName, GameReleaseDate
  FROM Game
  ORDER BY GameReleaseDate DESC, GameName ASC
  LIMIT 10;
```

Query 2: Find the top 11-20 rated games.

```
SELECT GameName, GameRating
  FROM Game
  ORDER BY GameRating DESC, GameName ASC
  LIMIT 10 OFFSET 10;
```

Query 3: Find the user with the most Achievements.

```
SELECT Username, count(*) AS NumOfAchievements
  FROM User
    NATURAL JOIN ACHIEVED
    NATURAL JOIN Achievement
  GROUP BY UserID, Username
  ORDER BY count(*) DESC
  LIMIT 1;
```

Query 4: Find the Developer Studio who has the highest average rating across all of their games.

```
SELECT StudioName, max(AvgRating) AS HighestAvgRating
  FROM (SELECT avg(GameRating) AS AvgRating
          FROM DeveloperStudio
            NATURAL JOIN DEVELOPED
            NATURAL JOIN Game
          GROUP BY StudioID, StudioName) AS TableAvgRating;
```

Query 5: Find Ticket IDs that have been solved with the help of three or more Employees.

```
SELECT TicketID, count(*) AS NumOfEmployeesSolved
  FROM Ticket
    NATURAL JOIN HELPS_WITH
    NATURAL JOIN Employee
  WHERE TicketDataSolved IS NOT NULL
  GROUP BY TicketID
  HAVING count(*) >= 3;
```

Query 6: Find Games that are from the 'RPG' genre with a rating of 4.0 or more stars ordered from the highest rated to the lowest rated game. Order the game's name from A-Z last.

```
SELECT GameName, GameGenre, Rating
  FROM Game
  WHERE GameGenre = 'RPG' AND Rating >= 4.0
  ORDER BY Rating DESC, GameName ASC;
```

Stakeholders:

The users of the my program (website) will be 13 years old or older. For 13 to 16 year olds, they will be able to use the site to socialize with friends and play games together. Some may play singleplayer games for fun, or are just looking for leisure time. For 17+ year olds, I expect it to be similar. Otherwise, 17+ year olds may want more mature rated games (based on the ESRB rating system) and the site should be expected to not filter games rated for 17+ year olds.

Technological Requirements:

The platform will mainly be the web. I anticipate the type of languages I will use to be HTML, Unity WebGL, Node.js, and JavaScript. The expect the database I plan to use will be MySQL. Any code I share will be through GitHub. I have good experience with Web Applications, mainly through HTML and Unity WebGL games. I expect that I will have to study and learn how to integrate all the anticipated languages to work together and complete the main functions of the application. Node.js and JavaScript are new to me, but I believe with enough research and learning, I can complete this project down to the requirements. I expect to not complete all of the functions that I envision for the website, but I hope to complete all the requirements for this project when it's due.