DAT602, Database Application Development

Milestone 3 – Final Report

Submitted to: Todd Cochrane

Submitted on: 27th June 2019

Submitted by: Achuthanand Vasudevan

Table of Contents

Si. No	Content		Page. No
1	Introduction		1
2	Storyboard & Description		2
	2.1	Login page	3
	2.2	Registration Page	4
	2.3	Game Dashboard – Player	5
	2.4	Game Dashboard – Admin	6
	2.5	Game room	7
	2.6	Manage Player(s)	8
	2.7	Settings	9
	2.8	Help	9
3	Design Choices		10
4	Logical Entity Relationship Diagram		11
5	CRUD Matric		14
6	Usage Scenario		16
7	Data Definition Language – DDL		19
8	MYSQL Procedures & Functions		20
9	Implemented Procedures & Functions		22
10	Support Over Multi-player access		25
11	ACID Properties		26
	11.1	Atomicity	26
	11.2	Consistency	27
	11.3	Isolation	28
	11.4	Durability	30

12 Error Handling 31

13 Implementing GUI 33

13.1 Implemented Forms 34

13.2 Implemented Classes 35

14 Important Links 36

15 References 36

1. Introduction

The game 'Roll Dice Win Game' is all about rolling a dice and according to the resultant the player token will move to respective blocks in the game board. The game is originally inspired from the classical 'Snake & Ladder' game. The game is designed in a way that there is no big rules or movements which makes this game to play by even small children. It is designed to support multiple players where each player can make their move in the order they joined. Since the game board is small enough to occupy only small group I insist to play with maximum of 4 and minimum 2 players. By insisting this restriction, it helps to avoid the congestion in the game. The fundamental goal of this protocol game application is to enable multiple players to play same game same time and the database should not fail to manage the data in real time.

The main elements of game room are a shared game board, a dice and a token for each player to make the movements. Each user will have each token with different color. The game board consists of 25 blocks including Start and Finish points. Apart from the First and Finish block, each block has obstacles which meant to be for the player to take extra movement according to the obstacle purpose. The main objective is to finish the game first regardless of how many players playing the game. The first player who finish the game will be the winner of game.

The game is basically designed using Visual Studio 2019 in alliance with MySQL database. The player should have login credential in-order to enter the game. The player needs to register or ask admin to add in-order to get the login credentials for the game. The username and email are unique and cannot be used again for another player. The system will record the player details and each game associated with that game. The score detail is displayed in the online players score list.

The game will be fully controlled by the administrators who has certain privileges to manage games and players. For the security purpose, invalid log-in tries of more than 5 times will lead the user to be stayed lock. The player then needs to contact the admin for retrieving locked account or could create new one by his own.

2. Storyboard & Description

Storyboard is a graphical organizer of the application which needs to be developed. These graphical representations can be visualized in the form of illustrations or images in a sequence which happens in the application. It helps the developers to initiate the development process with the basic design which can be maintained throughout the development. This gives a visual idea on how the game is playing and thus the developers will be able to develop the application real time without having confusion in the design part.

It is a quick way of getting a perspective into what the game will look like prior to the production even before a prototype is developed and tested. These storyboards will help to see where there are gaps in the gameplay actions which will help to give richer experience for the game. While giving the storyboard it is very important to organize it in the proper way so that it will depend on the precise goals of storyboard session.

With storyboard it is equally important to describe what is happening in the storyboard. Storyboard includes the story background of game and the elements required for the game progress. These elements are initiated with the help of user actions. Hence it is also essential to describe what will happen when the user initiates the action.

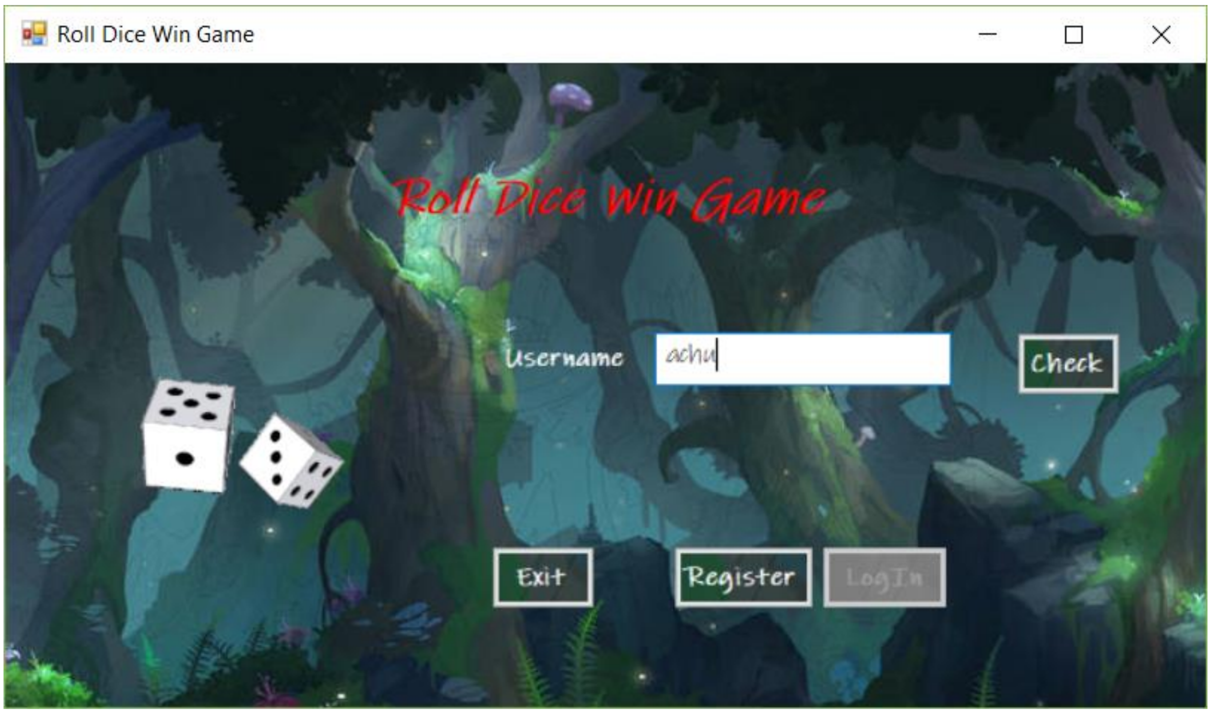
Advantages of storyboard and description are:

1. Will get an overview of the application just with a look.
2. Can improvise the design ideas.
3. Provides concise information.
4. Clear idea about the system before its development

Below explains the story board of game application and the descriptions of user actions related to the game.

2.1 Login Page

Board No: 1

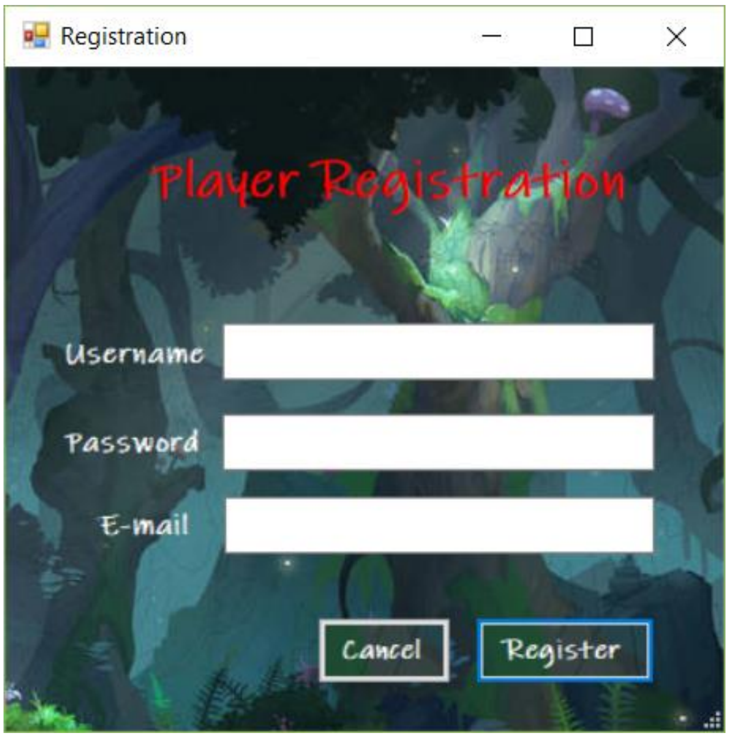


The player or admin will be needed to login before start playing or manage the game. The login page for both admin and normal players will be same. The system will automatically analyze the user type and will organize the further game pages.

Action	What happens?	Where to?
Click Check	1. Check if user exists or not?	
	1.1 If not, throw an error saying so	Go to board 2
	1.2 If exists, then allow the user to input password	
Validate username & password	1. Validate the username and password	
	1.1 If validation success	Go to board 3
	1.2 If admin login success	Go to board 4
	1.3 If the validation fails, throw error message and stay in the same login page. *Note: If the validation fails consecutively 5 times, the account will be locked, and an info dialog will show with an email contact of admin.*	
Click Exit	Exits the application	
Click Register	The registration from will open	Go to board 2

2.2 Registration Page

Board No: 2

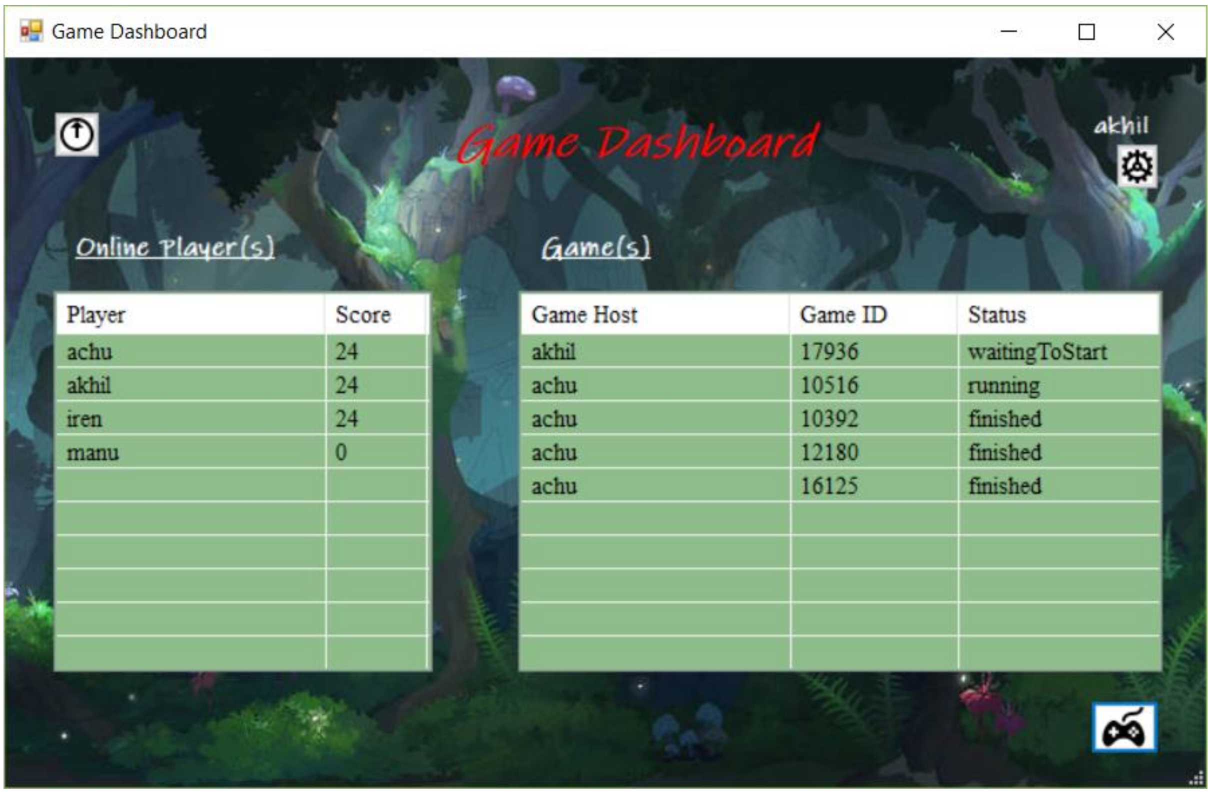


The registration page allows the user to register themselves or the admin can add players. The user cannot take the username which already exists. The email should be unique to every player.

Action	What happens?	Where to?
Click Register	1. Check all the fields are filled	
	1.1 If so then continue	
	1.2 If not then throw error	
	2. Check the username is already taken or not. If so then throw error.	
	3. Check the e-mail is a valid one or whether it is already taken or not.	
	3.1 If so then throw error	
	3.2 If valid then register successful	Go to board 1/6
Click Cancel	The registration form will close	Go to board 1/6

2.3 Game Dashboard – Player

Board No: 3



Player	Score
achu	24
akhil	24
iren	24
manu	0

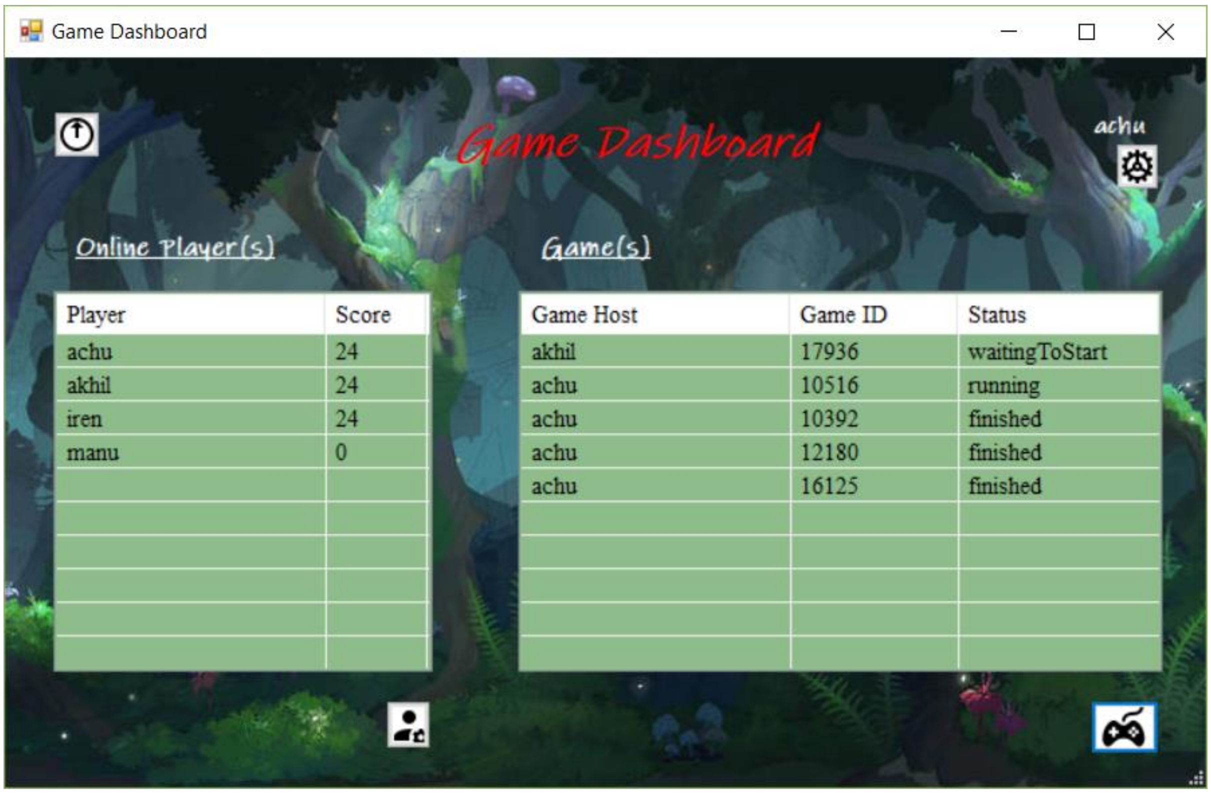
Game Host	Game ID	Status
akhil	17936	waitingToStart
achu	10516	running
achu	10392	finished
achu	12180	finished
achu	16125	finished

The dashboard consists of list of online players and games. The player can start a game or join live game. The status of live game is either ‘running’ or ‘waitingToStart’. The player can use Start button to start a new game. Right click on the available game will show option to join the game.

Action	What happens?	Where to?
Click Logout	The player will be out of the current session	Go to board 1
Click Settings	The player can reset the password	Go to board 8
Click Join	The Join option will be available only for ‘running’ or ‘waitingToStart’ games.	Go to board 5
Click Start	New game will start. *Note: The host need to start the game once the game room is available and atleast one more player joined.*	Go to board 5

2.4 Game Dashboard – Admin

Board No: 4



Player	Score
achu	24
akhil	24
iren	24
manu	0

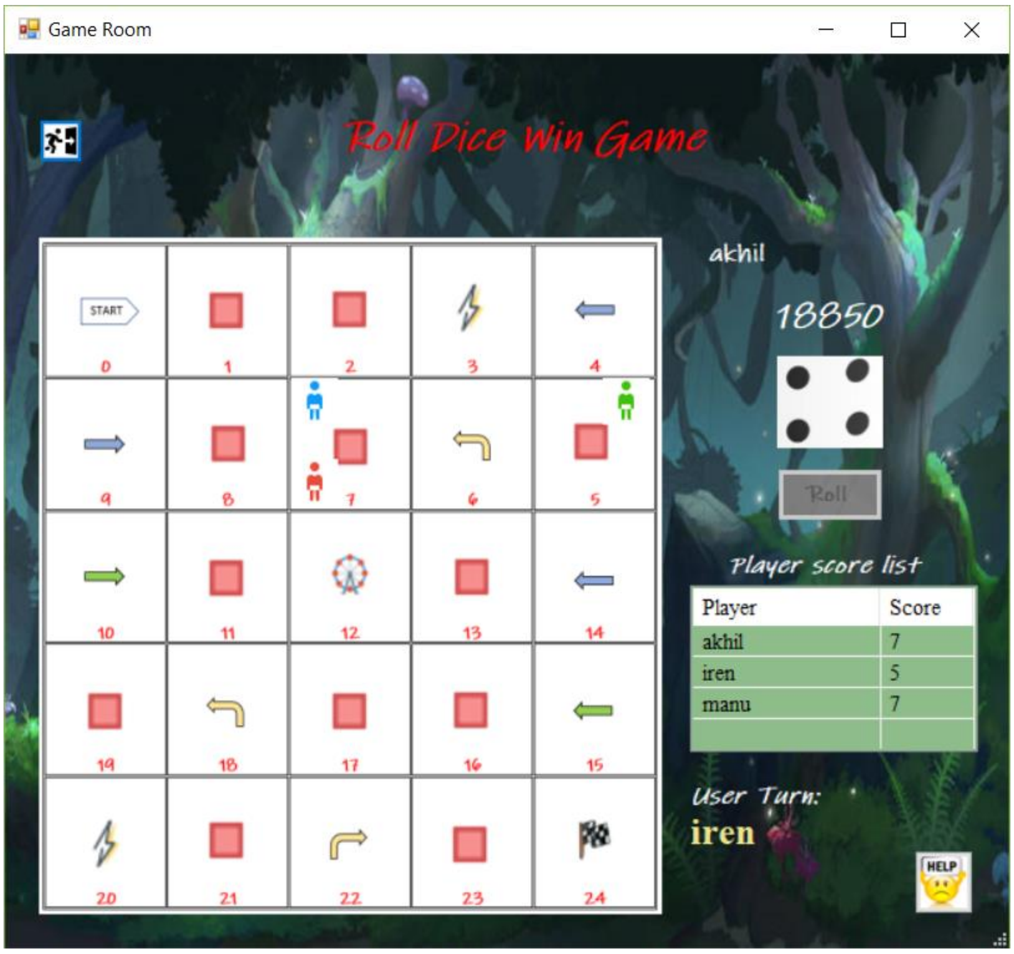
Game Host	Game ID	Status
akhil	17936	waitingToStart
achu	10516	running
achu	10392	finished
achu	12180	finished
achu	16125	finished

The dashboard for admin will have one extra button to manage the players and will have one option of terminating running games when right click in the game shown in the game list.

Action	What happens?	Where to?
Click Logout	The player will be out of the current session	Go to board 1
Click Settings	The player can reset the password	Go to board 8
Click Join	The Join option will be available only for 'running' or 'waitingToStart' games.	Go to board 5
Click Start	New game will start. *Note: The host need to start the game once the game room is available and atleast one more player joined.*	Go to board 5
Click Manage Player(s)	The admin can add, remove, unlock and update password of players.	Go to board 6
Click Terminate	The selected game will be terminated, and players will come out of the game.	

2.5 Game Room

Board No: 5



akhil

18850

Player	Score
akhil	7
iren	5
manu	7

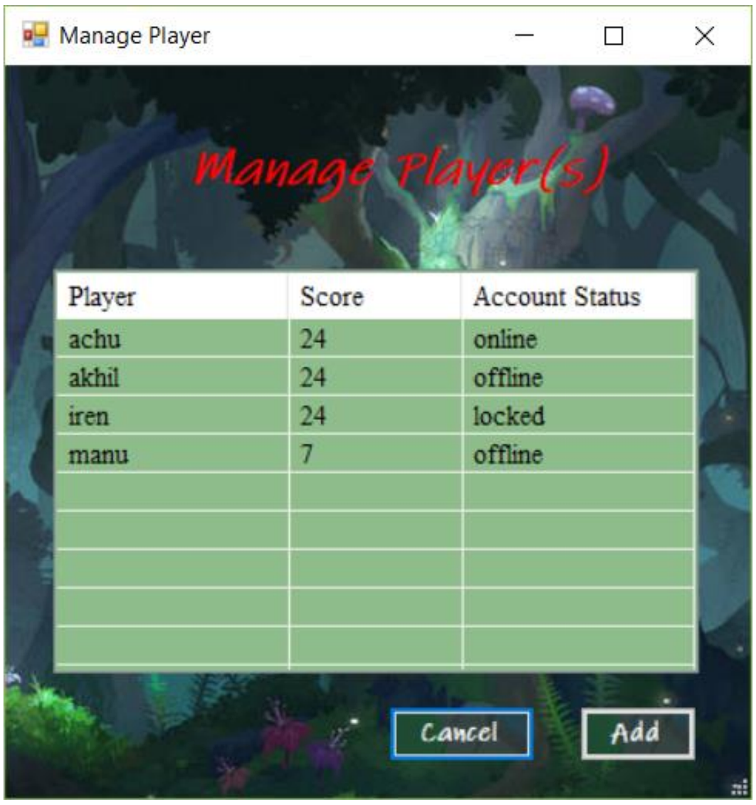
User Turn:
iren

The game room consists of game board, live score list, next user turn, dice to roll and button to roll the dice. When the player turn comes the player can roll the dice and according to it the token will move.

Action	What happens?	Where to?
Click Roll	The dice will roll and shows the resultant side. According to the result the movement of corresponding player token will happen.	
Click Help	Another form will show regarding the details of game symbols	Go to board 8
Finish game	The game will finish when one of the players reaches FINISH and the control will go back to dashboard	Go to board 3/4
Click Exit	The user will be exited from the game and cannot return to the same game again	

2.6 Manage Player(s)

Board No: 6



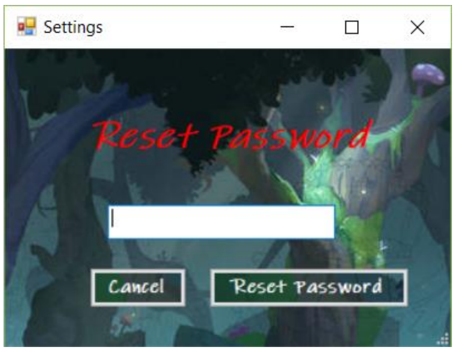
Player	Score	Account Status
achu	24	online
akhil	24	offline
iren	24	locked
manu	7	offline

The options to unlock, delete and reset password will be available according to the status of user when right click on the user row. The admin cannot insist any operation on online players. Unlock option will be available for only locked users. The options to delete and reset password will be available for offline and locked players.

Action	What happens?	Where to?
Click Reset Password	The selected player password will reset to new given password.	Go to board 8
Click Un-lock	The selected locked player will be unlocked	
Click Add	The admin can add new player through registration page	Go to board 2
Click Delete	The selected player will be deleted from the game system	
Click Cancel	The control will go back to the Admin dashboard	Go to board 4

2.7 Settings

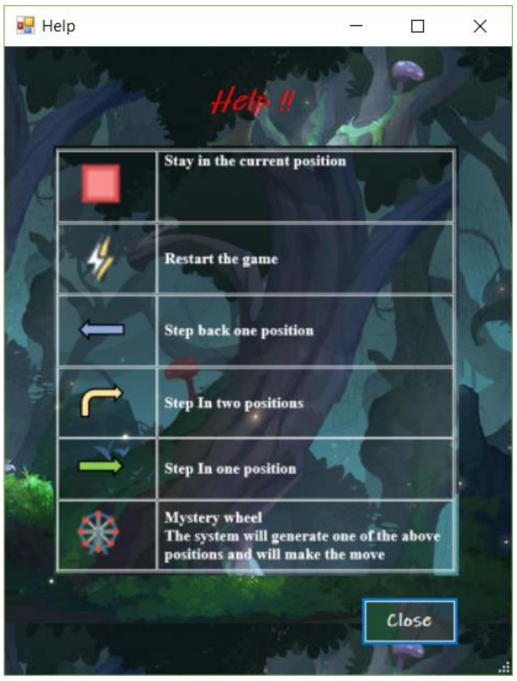
Board No: 7



Action	What happens?	Where to?
Click Reset Password	1. The password will reset to new given password.	Go to board 3/4/6
	2. If the given password is old password throw error	
Click Cancel	The control will go back to the Admin dashboard	Go to board 3/4/6

2.8 Help

Board No: 8



🟥	Stay in the current position
⚡	Restart the game
←	Step back one position
↱	Step In two positions
→	Step In one position
✳	Mystery wheel The system will generate one of the above positions and will make the move

Clicking Close button will close the help page and control go backs to game room.

3. Design Choices

The design of the game application is optimized to a maximum level from the design sketches in the Milestone 1 submission. The design is mainly optimized in terms of removing unnecessary forms, button etc. In the case of managing games by admin instead of another form the option to terminate a game is provided in the context menu options of game list. This 'terminate' option will be available only to admin users. For normal user only the option 'Join' will show.

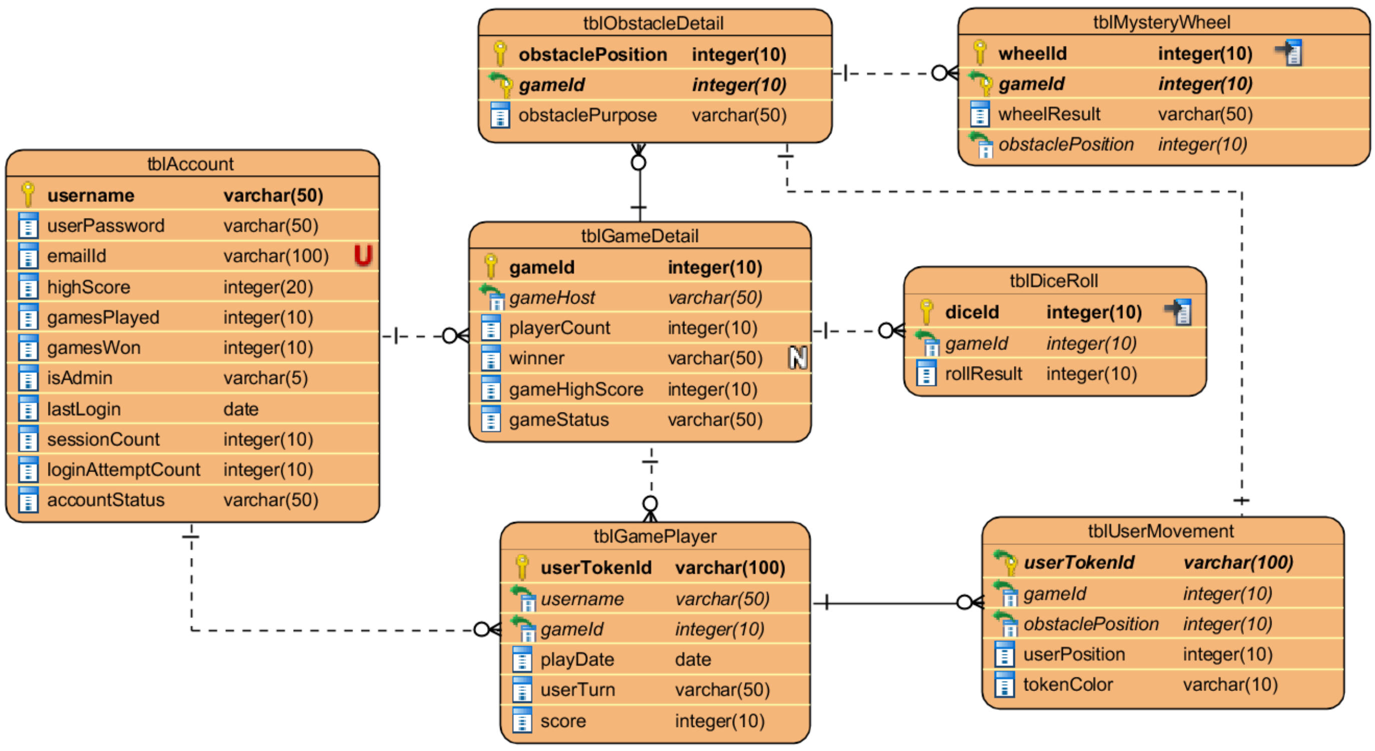
Similarly, in the case of managing players instead of implementing buttons for delete, unlock and reset password, these options are provided in the context menu of the player list. The implementation of context menu in list view of game and player helps to avoid lot of error checking and conditions. This makes the design efficient.

Since this is a protocol game application the movement of tokens in the game room is not animated. Hence the token will move to destination in one movement like disappearing from the initial position and will appear in the destination position.

Design of dashboard for normal player and admin can be designed in 2 ways. Considering the view of admin options, the options can be made available only to admin by make use of visible property of component. Another way is to make use of inheritance concept where making a form which has common elements for both normal players and admin and from it designing new form for normal player and admin will implement an effective and scalable design. In the current situation since only one button and one context menu options are different in both dashboard I use the visible property for button and dynamic implementation for context menu options.

One of the issues I faced was the size of background image used for the applications. The first version has a better-quality image but while implementing the functionalities it makes the application little slower to buffer. Hence I reduced the size of the image without much compromising the quality.

4. Logical Entity Relationship Diagram



The following table represents the data which has actual datatype which is different from the Logical ERD since some of the datatype options are not available in visual paradigm.

Actual datatype & datatype in Logical ERD

Si. No	Table/Entity	Column	Actual datatype	Datatype in ERD
1	tblAccount	lastLogin	DateTime	Date
2		accountStatus	Enum	Varchar
3		isAdmin	Enum	Varchar
4	tblGameDetail	gameStatus	Enum	Varchar
5	tblGamePlayer	playdate	DateTime	Date

Logical Entity Relationship Diagram which is short formed as Logical ERD is logical data model which describes a specific problem domain expressed independently of certain database management technology. But there are described using terms such as relational tables & columns, object-oriented classes or xml tags.

The above shown figure is the Logical ERD of my game application. In-order to manage the data related to game it is essential to design a storage system. Thus, the role of database comes and the need of effective and scalable design. The Logical ERD helps the database designer to create a Logical representation of database tables by illustrating the relation between tables, constraints imposed etc. Sometimes this may have slight changes when the development in progress, but it is recommended to design perfect instead of updating on the go. In-between alters in the table results in-consistent data and further issues while scaling the system.

- For the efficient and concise running of the system 7 tables are designed which focuses on the player details, game details, player actions and the decisions needed to run the game.
- tblAccount is maintained to store the player details including the login details and play details. This login details will be used to check the authentication of user in the game.
- tblGameDetail is related to tblAccount such that each account can host one or more game. Hence one to many relations. tblGameDetail will hold the high-level details of each game.
- tblGamePlayer is maintained to store the details each player related to a game. Each game will have multiple players and hence the tblGameDetail will be related to tblGamePlayer with one to many relationships. Likewise, each account will be related to many games other than host relationship, thus between tblAccount and tblGamePlayer one to many relations exists.
- tblDiceRoll is used to get the result when a dice is rolled. Each game will have atleast one dice and a maximum of ‘m’. Each time the dice is rolled this table will have new entry related to the game. Hence each game has multiple dices with one to many relations.
- tblObstacleDetail is used to maintain the game board detail such as Obstacle purpose and position. Each game will have multiple Obstacle in the game board and hence One to may relations with the tblGameDetail.
- tblMysteryWheel is used to generate the Obstacle when the user movement in the board results in Mystery Wheel Obstacle (obstacle position 12).

- tblUserMovement is maintained to track the user movements in the game board. This tracking of movement is essential since it is required to calculate the next movement when the player is rolled the dice. It is related to tblObstacleDetail since the calculation is dependent with the Obstacle purpose which can be known using the Obstacle position. Each user movement will have to analyze an obstacle; hence the relation is One to One.

Each time when a dice is rolled the outcome is calculated using the table tblDiceRoll. Each game will have multiple dices which rolls. In accordance with the table tblDiceRoll to calculate the user movement table tblObstacleDetail is required since each position in the game board has purpose which need to execute. This purpose can be retrieved from tblObstacleDetail by comparing the user position + outcome of the dice rolled with the obstacle position. Table tblMysteryWheel is maintained to auto generate the obstacle purpose if the user position + dice outcome is 12.

5. CRUD Matrix

CRUD matrix is a mechanism which help the developers to identify the tables and operations imposed to those tables required to run the application successful. CRUD stands for:

1. C – CREATE
2. R – INSERT
3. U – UPDATE
4. D – DELETE

The below table represents the CRUD matrix for the game application. It includes the tables required for the game and the operations involved for those table by the user actions in the game. These CRUD operations may affect the records in the database fully or partially. Some delete operations will delete the data not only from the mentioned table but also from the tables it has reference through foreign key. But for only the foreign becomes null but the row will be kept for future references.

Entity / Action	tblAccount	tblGameDetail	tblObstacleDetail	tblMysteryWheel	tblGamePlayer	tblDiceRoll	tblUserMovement
Username Check	R						
Login Check	R						
Login Success	U						
Fail Login	U						
Register	C, R						
Start a Game	U	C	C		C		C
Playing Game			R	C, R	U	C, R	R, U
Quit Game	U				U		
After Game Finish	U	U			U		
Join a Game	U	R, U			C		C

Admin	Reset password	U						
	Un-lock player	U						
	Delete a player	D						
	Terminate a game	U	U					
Log Out		U						

These tables are designed for the saving and manipulation of data related to the game application. These includes the user data, game data and auto generated data which is essential for the running of game. The tables are designed is such a way that no duplication of data will happen and relation between the tables helps the system to retrieve and update the data. Each table has its own responsibility in the proper and efficient running of game application. Below explain the purposes of each table used in the application.

Table purposes

tblAccount	:	1. Store the information of each players registered in the system. 2. Used to check the login and locking process.
tblGameDetail	:	Store the information of each game including host.
tblObstacleDetail	:	Store the information of obstacle position including the purpose.
tblMysteryWheel	:	Used to generate the result of mystery wheel for each game.
tblGamePlayer	:	Stores the information of each player related to each game.
tblDiceRoll	:	Used to generate the result when a dice is rolled for each player in each game.
tblUserMovement	:	Used to track the movement of tokens of each player related to each game.

It is very essential to implement these CRUD operations efficiently since it will affect the performance of the game. The operations are built upon the system according to the application performs. The data retrieval and updates should be made effective so that no delay in response time will happen.

6. Usage Scenarios

Usage Scenarios helps to describe a real time process of how the users are interacting with the system and for each user actions what is happening. It describes the steps, events, actions, results which occur during the interaction between system and user. It can be a detailed explanation exactly how a user works with the system.

The usage scenarios can be applied or described in different ways considering the system and designer. The basic strategy of usage scenario is to identify the system navigation through use cases and describe the user scenario for the interaction. The below given details explains the user scenario of game application I designed. It is explained in a way such that what will happen for each user actions and resultant system navigations.

Usage Scenario

1. User/Player Login

1.1. Enter the username.

1.2. Click Check.

1.3. If the username exists then the password input will be shown and can login.

1.4. If the username doesn’t exist then the user needs to register.

1.5. Maximum of 5 invalid login is allowed and the next invalid try locks the account.

1.6. It is essential to contact the admin to unlock the account once it is locked.

1.7. Successful login navigates the user to dashboard

1.8. The dashboard view may vary according to the user privilege.

2. User registration

2.1. Enter the user details.

2.2.The username should not be the one which is already taken.

2.3. The email ID should be unique to each player account.

2.4. All the input fields should be filled with valid data.

2.5. Clicking Register results the new user in the system.

2.6. The new user can use Login page to login into system.

3. Starting a Game

3.1. Click Start to start a game.

3.2. The host will be navigated to the game room.

3.3. The game list in the dashboard will update with ne game details

3.4. The host can start the game once atleast one more player joined.

3.5. The Quit button will help to quit from the game.

3.6 Once quit the player cannot enter the same game again.

4. Joining a Game

4.1 Right click on the live game

4.2.Click Join.

4.3 The corresponding game room will open.

4.4. The Quit button will help to quit from the game and cannot return to it.

5. Reset Password

5.1. Click settings.

5.2. The user can reset the password.

5.3. If the new password is existing one, then throw error.

5.4. If the new password is exactly new then the newer one overrides the old.

6. Logout

6.1. Clicking logout button will end the user session.

6.2. Successful end of session will navigate to the login page of system.

7. Admin Privileges

7.1. Manage Games

7.1.1. Right click on the running game.

7.1.2. Terminate option will be shown for only running games.

7.1.3. Click terminate.

7.1.4. The game will terminate.

7.2. Manage Players

7.2.1. If click Add then the system will navigate to Registration screen

7.2.2. Delete player

7.2.2.1 Right click on the player.

7.2.2.2 Delete option will be available only to offline users.

7.2.2.3. Click delete.

7.2.2.4. The player will be removed from the system.

7.2.3. Un-lock player

7.2.3.1. Right click on the player.

7.2.3.2. Unlock option will be available only to locked users.

7.2.3.3. Click unlock.

7.2.3.4 The player will be unlocked.

7.2.4. Reset password

7.2.4.1. Right click on the player.

7.2.4.2. Reset password option will be available only to offline users.

7.2.4.3. Click reset password.

7.2.4.4. Setting page will open.

7.2.4.4. Process same as case 5. Reset password.

8. Game Room

8.1. Play Game

8.1.1. The user token position will start from the START block

8.1.2. Clicking the Dice button will roll the dice

8.1.3. According to the outcome of dice rolled the user token will move. For an example if the outcome of dice roll is 6, the token will move to the 8th block. Then the system will analyze the obstacle in the 6th block. The obstacle purpose is 'stepInTwice', then the token will move further 2 steps more. Thus, in result the token reaches 8th block when the outcome of dice is 6.

8.1.4. The same process will continue and the player who reaches the FINISH block first will be the winner.

8.2. Help button will provide the details of each obstacle in the game board.

8.3. Exit button will quit the game and cannot enter to the same game again.

Note:-

- *Once the game is created by a player, other players can join the game by clicking the 'Join' option available in the context menu options of game list in the dashboard. The context menu will open when the player right clicks on the 'running' or 'waitingToStart' game.*
- *Admin privileges will be only available to admin only. The admin for the game has below credentials.*

  Username :- admin

  Password :- admin

7. Data Definition Language – DDL

A data definition or data description language (DDL) is a language similar to computer programming language for defining and manipulating the data structures of a database system. It is a standard for commands that define the different structures in a database. These statements help the database designers and architects to create, modify and remove database objects such as tables, indexes and users.

The most commonly used DDL statements are

1. CREATE – to create table
2. ALTER – to update table structure
3. DROP – to remove a table

The DDL queries related to the game application I am developing is given below. Since it is easier to analyze and view the queries in its original format I am sharing a link to the file. It can be downloaded or view to get the best view.

Click to get the DDL file.

The above file contains the DDL commands for Create database, Use database and a procedure to create all the related tables to the game application. The Create table queries will create required tables with imposing constraints to the data and columns.

8. MySQL Procedures & Functions

A procedure in SQL is same as a method in Object Oriented programming where the method won't have any return values. Procedures allows the developers to implement task independent of other tasks. It also allows the developers to reuse the existing procedures in multiple occasions which removes the duplication of code. The parameters can be defined as input-inly, output-only or both. It refers that a procedure can pass values back to the caller by using output parameters. A procedure can be accessed by using the "CALL" keyword.

Syntax of Procedure:

```
CREATE
    [DEFINER = user]
    PROCEDURE procedure_name ([proc_parameter[,...]])
    [characteristic ...] routine_body

proc_parameter:
    [ IN | OUT | INOUT ] param_name type

characteristic:
    COMMENT 'string'
  | LANGUAGE SQL
  | [NOT] DETERMINISTIC
  | { CONTAINS SQL | NO SQL | READS SQL DATA | MODIFIES SQL DATA }
  | SQL SECURITY { DEFINER | INVOKER }

routine_body:
    Valid SQL routine statement
```

Syntax of calling a procedure:

```
CALL procedure_name([parameter[,...]])

CALL procedure_name[()]
```

A function is as same as procedure in the working, only difference is that it must return atleast one return value as a result of the operation it executed also atleast one return statement within the function body to return the value to the called function. A function can act as an in-built functionality alike Object-Oriented programming. A function is invoked by “SELECT” keyword.

Syntax of function:

```
CREATE
    [DEFINER = user]
    FUNCTION function_name ([func_parameter[,...]])
    RETURNS type
    [characteristic ...] routine_body

func_parameter:
    param_name type

type:
    Any valid MySQL data type

characteristic:
    COMMENT 'string'
  | LANGUAGE SQL
  | [NOT] DETERMINISTIC
  | { CONTAINS SQL | NO SQL | READS SQL DATA | MODIFIES SQL DATA }
  | SQL SECURITY { DEFINER | INVOKER }

routine_body:
    Valid SQL routine statement
```

Syntax of calling a function:

```
SELECT function_name([parameter[,...]])

SELECT function_name[()]
```

9. Implemented Procedures & Functions

Following procedures and functions are required for the running of game application. There are 22 procedures and 4 functions implemented.

Procedures used:

1. CheckUserExistance

   This procedure check whether the input username exists in the system or not. If not then the system will ask the user to register. If the username exists then it will check whether the user is locked or not. If so then pop an information dialog suggesting the user to contact the admin. If not then the input box to provide password will be enabled.
2. RegisterNewUser

   This procedure allows the user to register themselves as well as for the admin to add players. There exist 2 main conditions for the successful registration. First one, the username should not be an already taken one. Second one is that the mail-id should be valid and unique. Once these conditions are passed with a strong password the player will be registered successfully.
3. CheckUserAuthentication

   This procedure checks whether the given access details are valid or not. Valid access details will navigate the user to dashboard according to the user privilege. If the access details invalid the system will keep the fail attempts and maximum allowed fail attempts is 5. When it becomes 5 the user will be locked, and the admin will be responsible to unlock those users once they requested.
4. GetOnlinePlayerList

   This procedure will retrieve the list of online players to display in the user dashboard.
5. GetGameList

   This procedure will retrieve the basic detail from the system ordered by gameStatus in ascending.
6. GetPlayerListForAdmin

   This procedure will retrieve all the users for the admin to implement the admin privileges. The admin can add, remove, unlock players as well as reset the password.

7. GetGameListForAdmin

   This procedure retrieves all the high-level details of games for the admin to terminate and follow the details. The procedure is different from the GetGameList is that the columns retrieved is different.
8. ResetPassword

   This procedure will reset the password for the selected user. This procedure is shared by normal users and admins.
9. DeleteUser

   This procedure allows the admin to remove the user from the system. Only players who is locked and offline cam be removed.
10. UnLockUser

    This procedure allows the admin to unlock the locked players.
11. TerminateSession

    This procedure terminates the running game. Only admin can do so.
12. LogOutSession

    This procedure **allows** the user to log out from the system. The navigation goes to Login.
13. CreateNewGame

    This allows the user to create a new game. It will create and new game will provide details such GameId, User token color and User turn.
14. InsertObstackeDetail

    This procedure inserts game related obstacle details when a game is created for the calculation of movements in game.
15. JoinGame

    This procedure allows the user to join the existing game. Using gameId the players can join the game. The maximum allowed users are 4 and minimum is 2. A player cannot join the game which he was playing and exits.
16. GenerateDiceRollOutput

    This procedure allows the user rolls the dice and shows the output of roll in the screen.
17. UpdatePlayerPosition

    This procedure will calculate the movement of user when a dice is rolled, and all the related tables will get updated. It provides details such as new position and next user turn to roll.

18. UpdateWhenGameFinish

    This procedure is used to update the user details including score and user turn when game is running as well as finished.
19. ExitGame

    This procedure will update the user details as well as game details when a user exits the game.
20. GetGameStatus

    This procedure is used to check the status of game when the control is in game room and acts according to the status. If the status is ‘finished’ or ‘terminated’ then the control will throw an information message to player and will close the game room.
21. GetGameBoardDetails

    This is used to update the game board periodically to show the player movements to every player playing the same game.
22. GetGameRoomScoreList

    This is used to update the live score list in the game room periodically and will update the turn to which player need to roll the dice.

Functions Used:

1. GenerateGameId

   This function generated random 5-digit number for gameId. The function will make sure that generated gameId is already taken.
2. CalculateNewPosition

   This function calculates the new position according to the output of dice rolled. This will make use of obstacle details and mystery wheel concept.
3. GetNextUserTurn

   This function calculated the which user has the next turn when a user rolled the dice. This functionality requires atleast 2 players and maximum of 4 players to calculate the exact turn.
4. GetTokenColor

   This function is used to provide color for the token when a player joined the game.

Click here to get the SQL file for procedures and functions.

10. Support over Multi-player access

The fundamental goal of this game applications is to allows multiple players to play a same game. As a protocol game application, I imposed a limitation of maximum 4 players. I did this because of 2 reasons. First is to avoid the complexities and second one is, for my game board more than 4 players will be a congested like structure which won't give a good gaming experience.

The game supports multiple players to play game at the same time. This facility is mainly allowed because of the database structure designed. The table are structured in a way that each player related to a game such that the same player can be related to another game and the details regarding these won't conflict each other.

Table tblGameDetail is used to store the game details but the table is not dealing with the user details such which are all the user are playing this game. At the same time the gameId which is the primary key of tblGameDetail will be used as a foreign key over table tblGamePlayer to store the player details related to that game. This table has unique userTokenId which is the primary key here has a value of "gameId_username". This unique format makes sure that the same player can have different records over this table with different gameId. Same relation is made to store the movement of user tokens in table tblUserMovement. This relation made between the tables makes sure that each user data won't get override unless it is of same game.

In terms of transactions I used the isolation level of READ COMMITTED which makes sure that the data read by a session transaction is the updated data of another session transaction committed. This makes sure that the game will run with exact data expected during the execution of transactions. This isolation level will create a lock on Read accessibility when the same record is being Write by another transaction. Since a maximum of only 4 players is allowed here this lock won't affect the performance of database.

11. ACID Properties

Before starting to discuss ACID properties, it is essential to understand what a transaction in MySQL is (or in any RDBMS), because ACID properties are meant for RDBMS to be transaction safe.

A transaction can be considered as a block of SQL/NoSQL queries that is to be independently executed for data retrievals and updates. This block may contain one or more statements to execute. A transaction starts with the first SQL statement and finish when it is committed or roll back. This commit and roll back is defined by the keywords COMMIT and ROLLBACK consecutively.

When a transaction imposes multiple changes in the database it may affect the data when the transaction is committed, or it won't affect the data when the transaction is rolled back. When a transaction is rolled back the imposed updates in the database are undone.

MySQL is a RDBMS which full satisfies the ACID requirements for a transaction-safe system. ACID is acronym for:

- Atomicity
- Consistency
- Isolation
- Durability

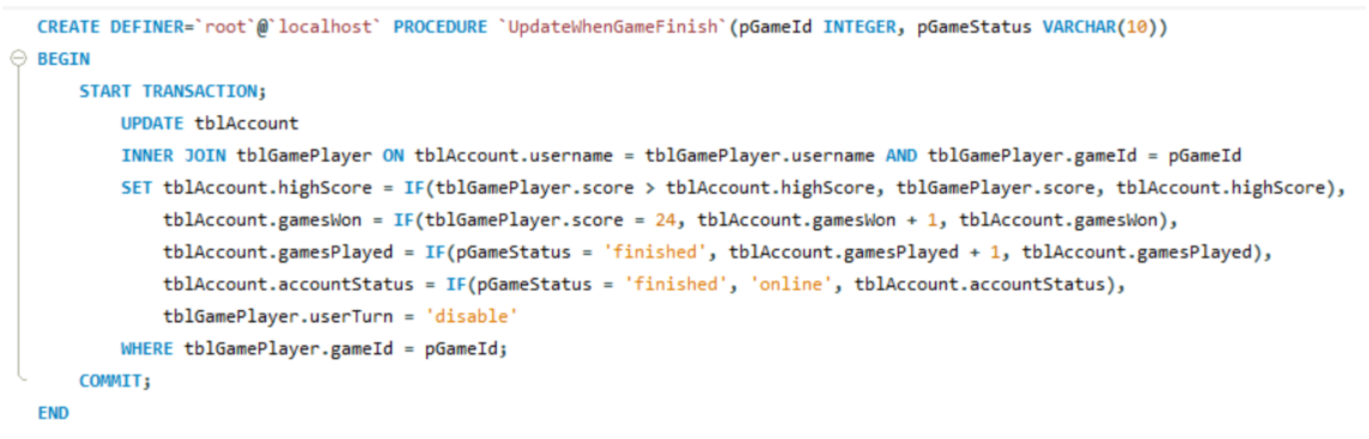
11.1 Atomicity

The concept Atomicity mainly involves the transactions in the database. As in any programming languages database transactions can be split into different blocks according to the logic it executes. In database context atomicity "refers to the integrity of the entire database transaction". I.e., when one part of the transaction fails to execute the remaining part will also fail as a result and vice versa. Simply a transaction must complete, committed or rolled back without impacting any changes in the data. Essentially it follows an "all" or "nothing" rule.

The technique behind Atomicity is the transaction will store the modified result in a memory buffer and will write to disk and binary log only when the transaction is committed. This maintains the independent nature or transaction and the result will appear only collectively or not at all.

In the game application I implement transaction where I am executing multiple impacts on the data which may fails during any unknown conditions. In such situation use of transaction with other ACID properties such as Isolation will help the data to roll back.

In MySQL the default settings of autocommit is 1. START TRANSACTION will basically SET autocommit=0 until COMMIT or ROLLBACK happens.



```sql
CREATE DEFINER=`root`@`localhost` PROCEDURE `UpdateWhenGameFinish`(pGameId INTEGER, pGameStatus VARCHAR(10))
BEGIN
    START TRANSACTION;
        UPDATE tblAccount
        INNER JOIN tblGamePlayer ON tblAccount.username = tblGamePlayer.username AND tblGamePlayer.gameId = pGameId
        SET tblAccount.highScore = IF(tblGamePlayer.score > tblAccount.highScore, tblGamePlayer.score, tblAccount.highScore),
            tblAccount.gamesWon = IF(tblGamePlayer.score = 24, tblAccount.gamesWon + 1, tblAccount.gamesWon),
            tblAccount.gamesPlayed = IF(pGameStatus = 'finished', tblAccount.gamesPlayed + 1, tblAccount.gamesPlayed),
            tblAccount.accountStatus = IF(pGameStatus = 'finished', 'online', tblAccount.accountStatus),
            tblGamePlayer.userTurn = 'disable'
        WHERE tblGamePlayer.gameId = pGameId;
    COMMIT;
END
```

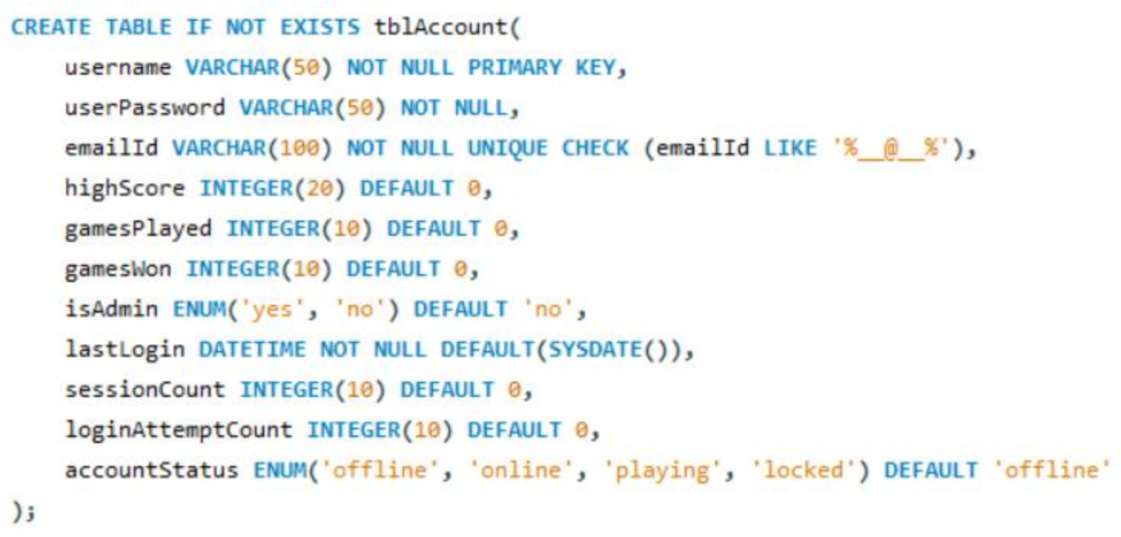
Figure 1 Using transaction for updating the user and game data

11.2 Consistency

Consistency maintains the database to protect data and from crashes when a constraint fails while updating the table or database. It won't allow to violate the integrity of database. Any interrupted changes make sure that it is rolled back to its prior state to the change. Basically, to maintain the integrity of database it must follow various appropriate validation rules. In the case of transaction, when the results don't follow these rules it will rolled back to previous state which passes the rules. As a result, a successful transaction keeps the database consistent with the rules it maintains.

In the game when adding data or updating data to the related tables certain constraints which applied to the columns should meet in-order to maintain the consistency between column data.

For example, the column gameId only accepts numbers which is an imposed constraint on the column. Other type of data will lose the constraint over the column. Also, in tblAccount for email 3 constraints is applied. One is it should not have null value, second it should be unique and 3rd it should follow the valid structure.



```
CREATE TABLE IF NOT EXISTS tblAccount(
    username VARCHAR(50) NOT NULL PRIMARY KEY,
    userPassword VARCHAR(50) NOT NULL,
    emailId VARCHAR(100) NOT NULL UNIQUE CHECK (emailId LIKE '%__@__%'),
    highScore INTEGER(20) DEFAULT 0,
    gamesPlayed INTEGER(10) DEFAULT 0,
    gamesWon INTEGER(10) DEFAULT 0,
    isAdmin ENUM('yes', 'no') DEFAULT 'no',
    lastLogin DATETIME NOT NULL DEFAULT(SYSDATE()),
    sessionCount INTEGER(10) DEFAULT 0,
    loginAttemptCount INTEGER(10) DEFAULT 0,
    accountStatus ENUM('offline', 'online', 'playing', 'locked') DEFAULT 'offline'
);
```

Figure 2 Constraints imposed on columns for tblAccount

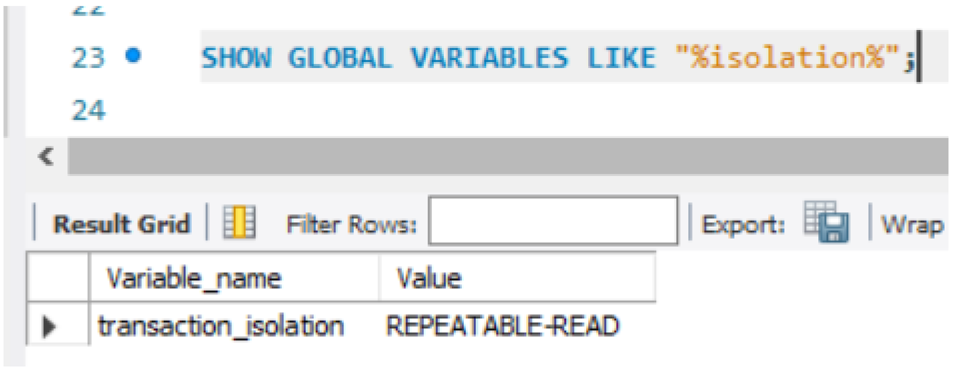
11.3 Isolation

Isolation is implemented to keep the isolation of different transactions without interrupting each other even they are referring of updating same tables or rows. “A transaction in context should not affected by any other on-going transactions”. This isolation technique avoids the collision of transactions. This aspect is very essential when multiple users happening to use the same system same time which is common today. When multiple users are accessing the system simultaneously a huge amount of transactions will run on the database. Hence isolation keeps the database to process these multiple transactions concurrently in a way it doesn’t affect other

There are mainly 4 type of isolation levels. The user can check which isolation level is used by executing the below command.

SHOW GLOBAL VARIABLES LIKE “%isolation%”;

The result will show which isolation level is configured in the current context. In the case of MySQL default isolation level is REPEATABLE-READ.



```
SHOW GLOBAL VARIABLES LIKE "%isolation%";
```

Variable_name	Value
transaction_isolation	REPEATABLE-READ

Figure 3 Checking isolation level

- REPEATABLE READ

  This isolation level impose isolation like it will return the data that was read regardless of whether that data was changed or not by another transaction in action. This isolation will help to return the result to the requested user at the when the user is requested the data. In between another transaction may have updated the data. But it won't affect the data read by the requested user. This won't fit in the situation where online shopping is happening with limited number of products available.

  To set this isolation level:

  ```
  SET GLOBAL TRANSACTION ISOLATION LEVEL REPEATABLE READ;
  ```

- READ UNCOMMITTED

  This setting enables a transaction to retrieve the data from a table where another transaction is updating. This makes sure that even though the update data by second transaction is not committed it will be available for the first transaction. This causes some inconsistency for the user when a updated data is shown and in between may it is reverted, but it allows the highest concurrency level in the server via many transaction can read the data from the server without the risk of any lock in the data of writing transactions.

  To set this isolation level:

  ```
  SET GLOBAL TRANSACTION ISOLATION LEVEL READ UNCOMMITTED;
  ```

- READ COMMITED

  Read committed will work exactly opposite to the read uncommitted. The data won't be available to a transaction until the second updating transaction updates the data and committed. This isolation level is little restrictive and at the same time it offers the best consistency for the data in the system. This isolation level is not an effective one when considering the system has lot of users because it will affect the performance of the server.

  To set this isolation level:

  ```
  SET GLOBAL TRANSACTION ISOLATION LEVEL READ COMMITTED;
  ```

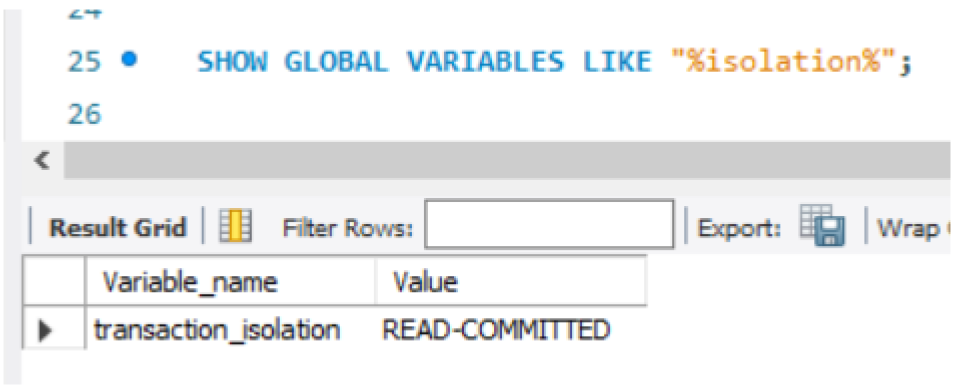
- SERIALIZABLE

  This isolation level is the most restrictive one compared to others. This will create a lock over the rows that is being read or write by a transaction. When the lock becomes active these rows cannot be read or write by any other transactions. There are mainly side effects exists for this isolation level. 1st one is many locks for both read and write operations. 2nd one is, even when a transaction is reading a data is not allowed when the same data is reading by another transaction which is not actually a required locking situation. This configuration will seriously affect the configuration of server because of the decrease concurrency and increased locks.

To set this isolation level:

```
SET GLOBAL TRANSACTION ISOLATION LEVEL SERIALIZABLE;
```

In the game application I would like to keep my database isolation be Read Committed. I believe it helps the game data to be consistent and the rows won't get override by other similar transactions. Since I am using Read Committed the players will be updated with latest details. Since as a protocol game application and only limited number of players the configuration won't affect the server performance.



```
SHOW GLOBAL VARIABLES LIKE "%isolation%";
```

```
Variable_name           Value
transaction_isolation   READ-COMMITTED
```

Figure 4 Setting isolation to Read Committed

11.4 Durability

Durability represents durability of the database to maintain the data even the system power goes down or crashes. Technical failure is a common issue in the technology. Even the technology advances this issue exists, but as technology advances there will be lot of ways to reduce the impact of these issues over data. Durability of a database, the fundamental goal is to make those failures invisible to the end-users. The data should not loose when a transaction is completed, even if the power outage or system failure happens. Scenarios such as in e-commerce site the durability has high impact over the data. Hence transaction durability is a must have.

MySQL durability features.

12. Error Handling

As in Objected oriented programming it is important to handle the exception or error that may happen while running the stored procedures. This will help the developer to continue with the execution by showing the error to the user and thus the application won't stop suddenly without any information. Even though the application can be restarted, this situation may cause inconsistency over data and while running the same procedure again may it won't work again as it is expected. HANDLER concept in MySQL provide the ability to handle these exceptions as well as error and thus the developer can make sure that the inconsistency in the data won't happen even if anything wrong happens.

There are mainly 2 types of handlers EXIT HANDLER & CONTINUE HANDLER. Exit handler will be used when an exception is thrown the handler will catch it and exits the procedure which stops executing further statements in that procedure. At the same Continue handler will catch the exception and the remaining statements continue their execution. If the later statements are depended on the statement which cause exception then it is recommended to use Exit handler.

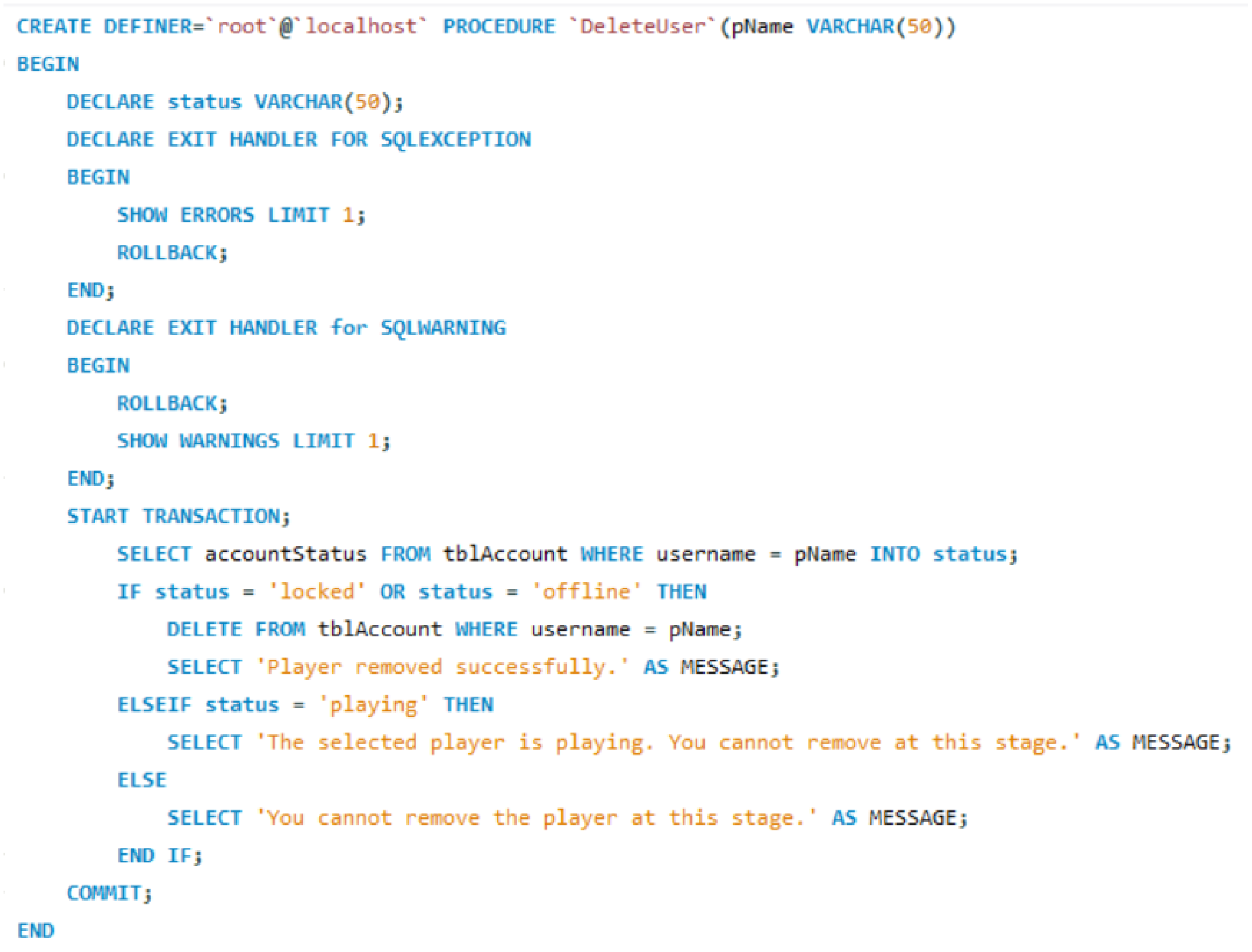
Syntax of Declaring handler:

```
DECLARE handler_action HANDLER
    FOR condition_value [, condition_value] ...
    statement

handler_action: {
    CONTINUE
  | EXIT
  | UNDO
}

condition_value: {
    mysql_error_code
  | SQLSTATE [VALUE] sqlstate_value
  | condition_name
  | SQLWARNING
  | NOT FOUND
  | SQLEXCEPTION
}
```

In the game application I am using handlers in every procedure with roll back property so that if any exception happens it will roll back the updated state to previous state thus, the consistency of the data maintains.



```
CREATE DEFINER=`root`@`localhost` PROCEDURE `DeleteUser`(pName VARCHAR(50))
BEGIN
    DECLARE status VARCHAR(50);
    DECLARE EXIT HANDLER FOR SQLEXCEPTION
    BEGIN
        SHOW ERRORS LIMIT 1;
        ROLLBACK;
    END;
    DECLARE EXIT HANDLER for SQLWARNING
    BEGIN
        ROLLBACK;
        SHOW WARNINGS LIMIT 1;
    END;
    START TRANSACTION;
        SELECT accountStatus FROM tblAccount WHERE username = pName INTO status;
        IF status = 'locked' OR status = 'offline' THEN
            DELETE FROM tblAccount WHERE username = pName;
            SELECT 'Player removed successfully.' AS MESSAGE;
        ELSEIF status = 'playing' THEN
            SELECT 'The selected player is playing. You cannot remove at this stage.' AS MESSAGE;
        ELSE
            SELECT 'You cannot remove the player at this stage.' AS MESSAGE;
        END IF;
    COMMIT;
END
```

Figure 5 Handler usage

The above figure shows the usage of handler in one of my procedure “DeleteUser”. I implemented Exit handler for SQLException and SQLWarning.

13. Implementing GUI

Graphical User Interface also called as GUI is a very important aspect in the application development since this is what the real user is going to see and interact with. It helps the users to interact with applications through a series of icons and visual indicators. GUI makes the user easy to interact with the system since the above-mentioned icons and visual indicators are implemented in comparison with the real-life indications. Hence the users need not to memorize what need to do to have some functionality working like in command line applications. In GUI implemented applications these visual indications are activated by using devices such as mouse and keyboard. In the current technical advancements even finger on a touch screen works smoothly.

Well implemented GUI helps any one with no technical knowledge can deal the system without facing any trouble. That is the fundamental goal of implementing GUI. While considering the GUI, developer needs to consider that the user is like a baby to the system. The user doesn’t know anything about the system, and hence it is essential that the system should be implemented in a way that even a starter can use it as a professional.

In my protocol game application, I tried to implement the GUI simple but little game oriented since I am doing a game. I needed to do my GUI in some dark mode, hence the background of all game pages is little darker. I need to make my background little catchy so that the user will investigate every functionalities of the system.

I used visual studio 2019 to design the windows forms and GUI for the game applications. MySQL is connected with the visual studio using MySQL.dll file. In the dashboard I user Icons on buttons instead of text and did tooltips to understand what the button is for. But I believe without the help of tooltip itself the user can understand what action button do. I used icons similar to the functionality it has. Like an icon of wheel for the settings, joystick for the start game button etc.

In-order to load the player list and game list in dashboard, live score list in game room and player list in manage player page I used ListView component. Even though there is no dataSource property for ListView it has a nice view compared to the DataGrid. I was able to adjust the properties of ListView by matching with my game background. When I tried to use DataGrid, the view was not good. It is not adjusting its height with the background where rows present to the blank area of it. This looks awkward in the screen. I used ContextMenuStrip component above list view since efficient use of ContextMenuStrip avoid the usage of buttons and unnecessary conditions.

Considering the game room, I used PictureBox Component to load my game board. The game board is designed simply by using MS Word. I took advantage of table designs and insert picture options in MS word to design the game board. Also, I used pictures of various sides of dice to show when a dice is rolled. For this also I used PictureBox component.

Since the goal is to achieve multi player game the game room and dashboard need to update whenever a player login or a game movement is happened in the game room consecutively, I used the concept of Timer. It enables the screen to refresh periodically so that all the details in the screen stayed update.

Components I used to design GUI:

1. Label
2. TextBox
3. Button
4. Timer
5. ListView
6. ContextMenuStrip
7. PictureBox

13.1 Implemented Forms

1. FrmLogin

   This form allows the user to login and prompts Register page if needed.
2. FrmRegister

   This form allows the user to register themselves as well admin can add new users.
3. FrmPlayerDashboard

   Once the user login successful, dashboard will open. It provides the list of online players and game. The user can join any running game. Also, it allows the user to start a new game. If the login user is admin then the admin will be able to navigate to manage player page as well as terminate running games.

4. FrmManagePlayer
   This allows the admin to Add, Remove, unlock existing users as well as resetting their passwords.
5. FrmSettings
   This allows the user to reset their password. Admin will make use of this page to reset password other players as well as admin password also.
6. FrmGameRoom
   This is where the game runs. It consists of Game board, Dice to roll and live score list.
7. FrmHelp
   This helps the player to know about what the meaning of each obstacle in the game board is.

13.2 Implemented Classes

1. ClsConnectDbGetData – It will make connection with MySQL and has a method ‘ExecuteProcedure’ to call the procedure associated to the intended call and will return the result of procedure it called.
2. ClsExtractData – It will help to extract required data from DataSet returned by the ‘ExecuteProcedure’ method in the ClsConnectDbGetData class.
3. ClsConstants – It defines all the constants used in the game applications. Instead of hardcoding the literals I implemented a class for constants. It has static variables which has literal values and can be called without creating instance to the class.
4. ClsGameBoard – It has the variables required to update the Game board.
5. ClsGameList – It has the variables required to load the game list in dashboard.
6. ClsScoreList – It has the variables required to load the score list in dashboard and in game room.
7. ClsPlayerRedPosition – It has a list which contains the co-ordinate of red token.
8. ClsPlayerGreenPosition – It has a list which contains the co-ordinate of green token.
9. ClsPlayerBluePosition – It has a list which contains the co-ordinate of blue token.
10. ClsPlayerBlackPosition - It has a list which contains the co-ordinate of black token

14. Important Links

Click here to open Journal

Click here to download the SQL file for procedures and functions.

Click here to download the Game Application.

15. References

1. *MySQL Acid Properties*. Retrieved from https://logicalread.com/mysql-acid-properties-mc13/#.XQAZk8TVKMo
2. *Definition: ACID*. Retrieved from https://searchsqlserver.techtarget.com/definition/ACID
3. *What is meant by ACID?*. Retrieved from https://www.essentialsql.com/what-is-meant-by-acid/
4. *Definition: Transaction*. Retrieved from https://www.techopedia.com/definition/16455/transaction
5. *ACID Compliance means Care*. Retrieved from https://www.clustrix.com/bettersql/acid-compliance-means-care/
6. *MySQL ACID*. Retrieved from https://dev.mysql.com/doc/refman/5.6/en/mysql-acid.html