DAT602 Final handin

Sean Cavill

Contents

[Game Description 1](#_Toc75726826)

[Story Board 1](#_Toc75726827)

[1. Login Screen 1](#_Toc75726828)

[2. Menu 2](#_Toc75726829)

[3. Game 3](#_Toc75726830)

[4. Admin Tools 4](#_Toc75726831)

[5. Edit user 6](#_Toc75726832)

[6. New User 6](#_Toc75726833)

[Logical diagram 8](#_Toc75726834)

[Crud Table 8](#_Toc75726835)

[Refined Database 10](#_Toc75726836)

[Crud Updates 11](#_Toc75726837)

[Procedures and Transactions 13](#_Toc75726838)

[Basic User Functionality 13](#_Toc75726839)

[FindUser 13](#_Toc75726840)

[RegisterPlayer 13](#_Toc75726841)

[LoginPlayer 13](#_Toc75726842)

[LogoutPlayer 13](#_Toc75726843)

[GetOnlineplayers 13](#_Toc75726844)

[Admin Commands 13](#_Toc75726845)

[AdminAccess 13](#_Toc75726846)

[EditUser 13](#_Toc75726847)

[DeleteUser 14](#_Toc75726848)

[DeleteGame 14](#_Toc75726849)

[DeactivateGame 14](#_Toc75726850)

[GetAllGames 14](#_Toc75726851)

[GetAllPlayers 14](#_Toc75726852)

[Game Creation 14](#_Toc75726853)

[CreateGame 14](#_Toc75726854)

[Game Interaction 14](#_Toc75726855)

[GetActiveGames 14](#_Toc75726856)

[JoinGame 14](#_Toc75726857)

[MovePlayer(Handles picking up item) 14](#_Toc75726858)

[GetActiveGamePlayers 15](#_Toc75726859)

[LeaveGame 15](#_Toc75726860)

[GetItems 15](#_Toc75726861)

[ConsumeItem 15](#_Toc75726862)

[Chat 15](#_Toc75726863)

[SendMessage 15](#_Toc75726864)

[GetChat 15](#_Toc75726865)

[ACID 15](#_Toc75726866)

[Atomicity 16](#_Toc75726867)

[Consistency 16](#_Toc75726868)

[Isolation 16](#_Toc75726869)

[Durability 16](#_Toc75726870)

[Milestone 3 17](#_Toc75726871)

[Login Screen 18](#_Toc75726872)

[LobbyForm 20](#_Toc75726873)

[Admin Form 22](#_Toc75726874)

[User Form 24](#_Toc75726875)

[Game Form 26](#_Toc75726876)

[Conclusion 37](#_Toc75726877)

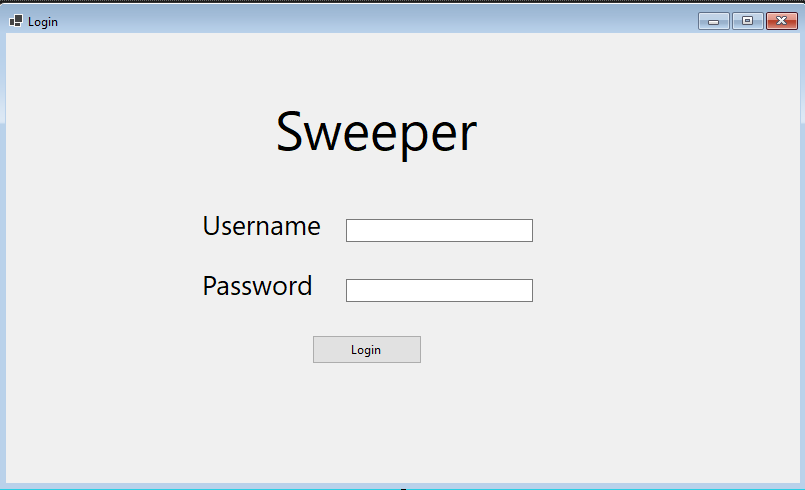
[REFERENCES 38](#_Toc75726878)

# Game Description

The idea for my game is a multiplayer top-down square grid map where players move between neighbouring 8 tiles (except when they are against the wall of the map) to collect items and gain score. All players are going to start on a ‘home tile’ that they will not be able to move back to once they leave. All the other tiles on the map will only be able to be moved to if another user is not already on them at that time. The map will be a ‘minesweeper’ style where there are numbers indicating where nearby traps are. Like minesweeper, the tiles will not be visible until one of the game users have moved to that tile, or an area that is not close to traps is opened up. If a player moves to a hidden tile that happens to have a trap on it, that player will be given a debuff (slow speed, loss of score, lose items). There will also be tiles on the map that contain items that will give users buffs (increase score, other user debuffs, powerup items) that will aid them in the game. Once every tile without a trap on has been cleared by the players of that game, the game will end and the user with the highest score wins.

# Story Board

## Login Screen



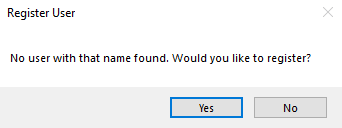
1.3

1.2

1.1

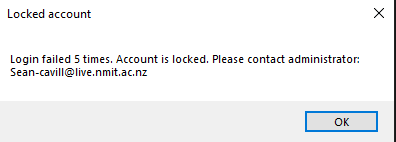
On the login screen the user will enter a username and a password. If the login fails 5 times the associated account will be locked and they will be prompted to contact an administrator.

* 1. – User enters a user name here
  2. User enters a password here
  3. Login button. If username exists in the database it will go on to the menu screen. If not the following dialogue box appears.

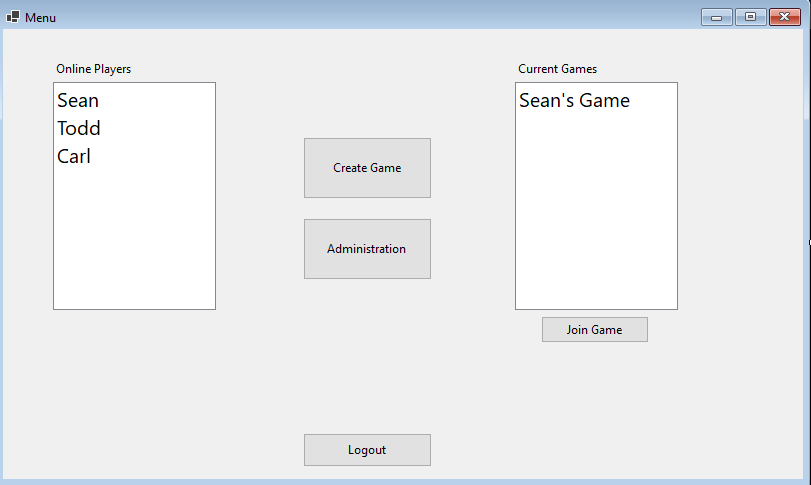


If the user clicks yes it will register the new user in the database and move on to the menu screen.

If the user fails to log in with the correct password 5 times the account will be locked and can only be unlocked by an administrator.



## 2. Menu



2.6

2.5

2.4

2.3

2.2

2.1

This is the menu screen where the user can see the online players, join a current game, create a game, and go to the administration screen if they have the privileges.

2.1 – List of online players

2.2 List of joinable games

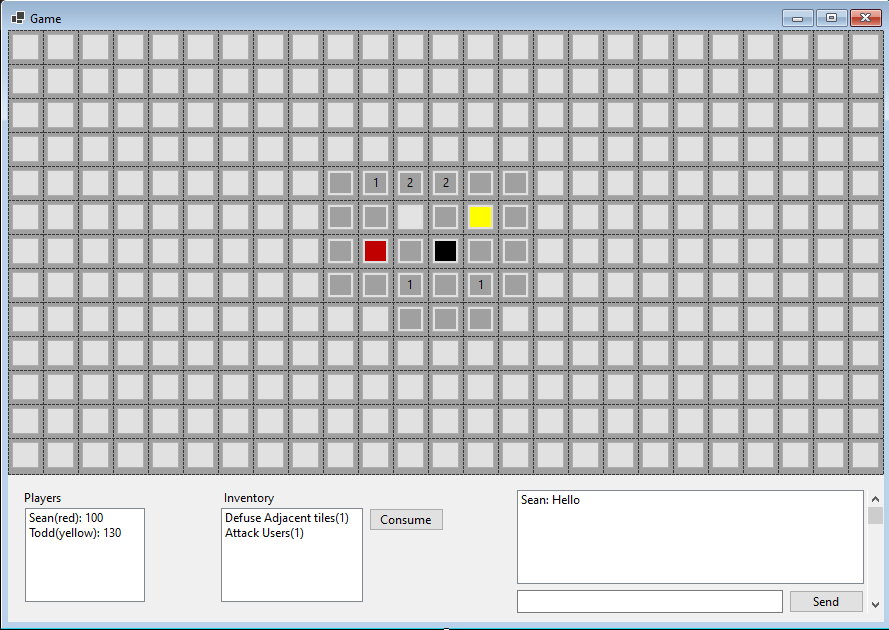
2.3 – joins game if a game on the list is selected. Goes to the Game Story board.

2.4 – Create a game goes to the game story board.

2.5 – goes to administration tools(only available to admins)

2.6 – Logs the user out and returns to the login screen.

## 3. Game



3.4

3.3

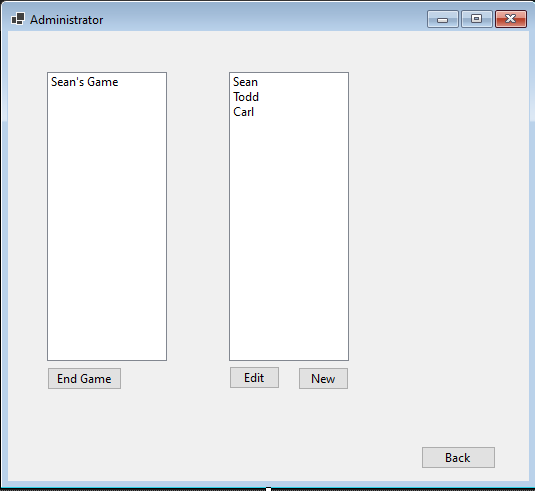
3.2

3.1

Game screens where users will play against each other. The map will show numbers near where trapped tiles are. There will be assets on the map users need to double click in order to add to their inventory. Other users are shown on the map as a coloured blob.

* 1. – Home tile where all the users will start.
  2. – Shows players, scores, and their colours.
  3. – Shows users inventory of items they can consume
  4. - Chat where users can send messages to each other.

## 4. Admin Tools



4.3

4.5

4.4

4.2

4.1

If an admin clicks on the administrate button on the home screen they will get to this administration page.

4.1 – Shows a list of active games

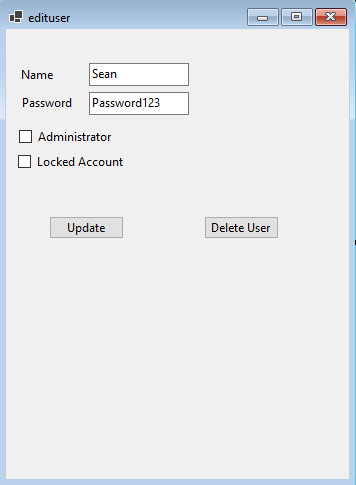
4.2 – Shows a list of registered users

4.3 ends currently highlighted game

4.4 Edits selected user goes to edit user screen

4.5 Goes to create new user screen

## 5. Edit user



5.6

5.5

5.4

5.3

5.2

5.1

Edit user screen that is filled from the selected user. The admin can change the name, password, admin status, and locked status of the account.

5.1 – Name of the user

5.2 – Password of the account that can be changed.

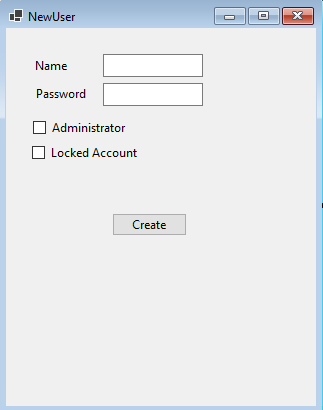
5.3 – admin status of the user that if checked gives them administrator privileges.

5.4 – locked status of the user. Can be used to lock or unlock the account.

5.5 – updates the user record with the new information in the fields. Takes admin back to administration screen.

5.6 – deletes the user permanently takes admin back to administration screen.

## 6. New User



6.5

6.4

6.3

6.21

6.1

Screen for the admin to create a new user.

6.1 – Name of the user

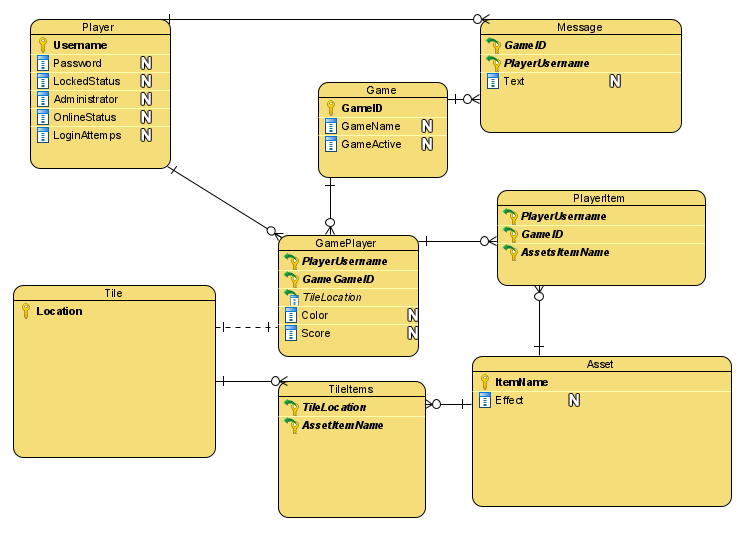
6.2 – Password of the account

6.3 – admin status of the user that if checked gives them administrator privileges.

6.4 – locked status of the user. Can be used to lock or unlock the account.

6.5 – Creates the user and takes the admin back to the Administration screen.

## Logical diagram

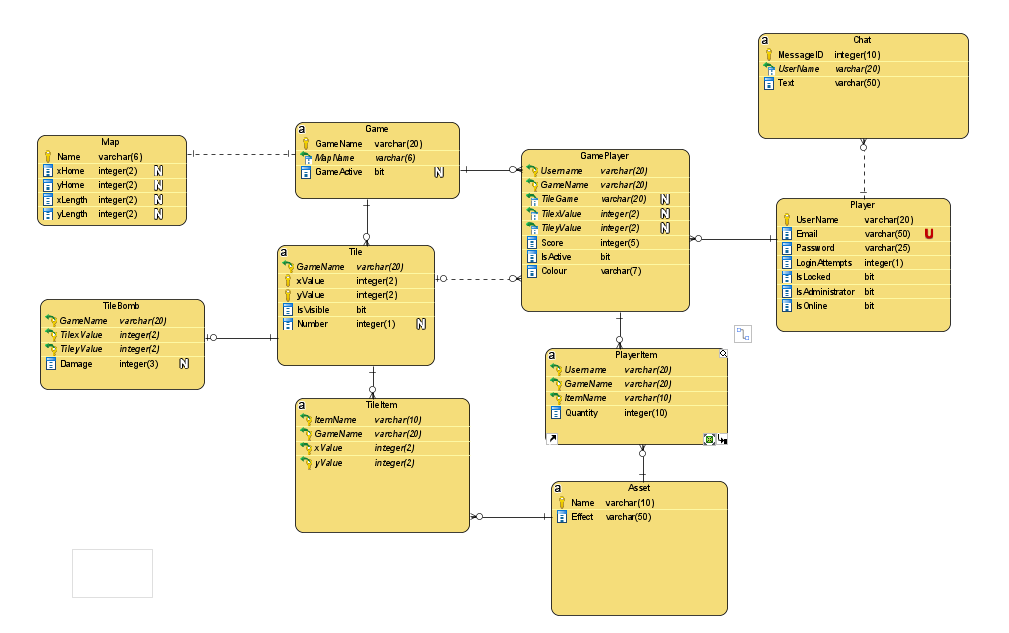


## Crud Table

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Entity/Attribute | Check Username | Register User | Validate Password | Lock Account | Log in | Log Out | Create Game | Join Game | Send message | Use Item | Move | Pick up item | Admin Access | Admin End Game | Admin Update user | Admin Delete user | Admin add new user |
| **Player** | R | C | R |  | U | U | R | R | R |  |  |  | R |  | RU | D | RC |
| Username | R | C | R |  |  |  | R | R | R |  |  |  |  |  | RU | D | RC |
| Password |  | C | R |  |  |  |  |  |  |  |  |  |  |  | RU | D | C |
| LockedStatus | R | C | R | U |  |  |  |  |  |  |  |  |  |  | RU | D | C |
| Administrator |  | C |  |  |  |  |  |  |  |  |  |  | R |  | RU | D | C |
| OnlineStatus | R | C |  |  | U | U |  |  |  |  |  |  |  |  | RU | D | C |
| Login Attempts | R | C |  |  | U |  |  |  |  |  |  |  |  |  |  | D | C |
| **GamePlayer** |  |  |  |  |  |  |  | CR |  | RU | R |  |  |  |  | D |  |
| Username |  |  |  |  |  |  |  | CR |  | R | R |  |  |  |  | D |  |
| GameID |  |  |  |  |  |  |  | CR |  | R | R |  |  |  |  | D |  |
| TileLocation |  |  |  |  |  |  |  | CR |  | R | R |  |  |  |  |  |  |
| Colour |  |  |  |  |  |  |  | CR |  | R | R |  |  |  |  |  |  |
| Score |  |  |  |  |  |  |  | CR |  | RU | R |  |  |  |  |  |  |
| **Tile** |  |  |  |  |  |  | C |  |  |  |  |  |  |  |  |  |  |
| Location |  |  |  |  |  |  | C |  |  |  |  |  |  |  |  |  |  |
| **TileItems** |  |  |  |  |  |  | C |  |  |  |  |  |  |  |  |  |  |
| TileLocation |  |  |  |  |  |  | C |  |  |  |  |  |  |  |  |  |  |
| ItemName |  |  |  |  |  |  | C |  |  |  |  |  |  |  |  |  |  |
| **Asset** |  |  |  |  |  |  |  |  |  | R |  | R |  |  |  |  |  |
| ItemName |  |  |  |  |  |  |  |  |  | R |  | R |  |  |  |  |  |
| Effect |  |  |  |  |  |  |  |  |  | R |  | R |  |  |  |  |  |
| **PlayerItem** |  |  |  |  |  |  |  |  |  | D |  | C |  |  |  | D |  |
| PlayerUserName |  |  |  |  |  |  |  |  |  | D |  | C |  |  |  | D |  |
| GameID |  |  |  |  |  |  |  |  |  | D |  | C |  |  |  | D |  |
| **Game** |  |  |  |  |  |  | C |  |  |  |  |  |  | R |  |  |  |
| GameID |  |  |  |  |  |  | C |  |  |  |  |  |  | R |  |  |  |
| GameName |  |  |  |  |  |  | C |  |  |  |  |  |  | R |  |  |  |
| GameActive |  |  |  |  |  |  | C |  |  |  |  |  |  | RU |  |  |  |
| **Message** |  |  |  |  |  |  |  | R | C |  |  |  |  |  |  |  |  |
| GameID |  |  |  |  |  |  |  | R | C |  |  |  |  |  |  |  |  |
| Username |  |  |  |  |  |  |  | R | C |  |  |  |  |  |  |  |  |
| Text |  |  |  |  |  |  |  | R | C |  |  |  |  |  |  |  |  |

MILESTONE 2

# Refined Database



I made some updates to my database. Chat is now global and is between players. Each Game tile now has a tile bomb and it can have 0 or many of them.   
Added a map to allow for variable game size. Added additional fields such as email. Made location x and y instead of using an ID as it is easier to perform procedures.

# Crud Updates

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Entity/Attribute | Find Users | Register User | Log in | Log Out | Create Game | Find Game | Join Game | See Game Players | Send message | See Messages | Use Item | Move | Pick up item | Admin Access | Delete Player | Edit Player | Delete Game | Deactivate Game |
| Player | R | C | U | U | R |  | R |  | R |  |  |  |  | R | D | RU |  |  |
| Username | R | C |  |  | R |  | R |  | R |  |  |  |  |  | D | RU |  |  |
| Email |  | C |  |  |  |  |  |  |  |  |  |  |  |  | D | RU |  |  |
| Password |  | C |  |  |  |  |  |  |  |  |  |  |  |  | D | RU |  |  |
| LockedStatus |  | C |  |  |  |  |  |  |  |  |  |  |  |  | D | RU |  |  |
| Administrator |  | C |  |  |  |  |  |  |  |  |  |  |  | R | D | RU |  |  |
| OnlineStatus |  | C | U | U |  |  |  |  |  |  |  |  |  |  | D | RU |  |  |
| Login Attempts |  | C | U |  |  |  |  |  |  |  |  |  |  |  | D | RU |  |  |
| GamePlayer |  |  |  |  |  |  | C | R |  |  | RU | RU | R |  | D | U | D |  |
| Username |  |  |  |  |  |  | C | R |  |  | R | R | R |  | D | U | D |  |
| GameName |  |  |  |  |  |  | C |  |  |  | R | R |  |  | D |  | D |  |
| xValue |  |  |  |  |  |  | C |  |  |  | R | R |  |  | D |  | D |  |
| yValue |  |  |  |  |  |  | C |  |  |  | R | R |  |  | D |  | D |  |
| Score |  |  |  |  |  |  | C | R |  |  | RU | RU |  |  | D |  | D |  |
| IsActive |  |  |  |  |  |  | C | R |  |  | R | R |  |  | D |  | D |  |
| Colour |  |  |  |  |  |  | C | R |  |  |  | R |  |  | D |  | D |  |
| Tile |  |  |  |  | C |  |  |  |  |  | RU | U |  |  |  |  | D |  |
| GameName |  |  |  |  | C |  |  |  |  |  |  | R |  |  |  |  | D |  |
| xValue |  |  |  |  |  |  |  |  |  |  | R | R |  |  |  |  | D |  |
| yValue |  |  |  |  |  |  |  |  |  |  | R | R |  |  |  |  | D |  |
| IsVisible |  |  |  |  |  |  |  |  |  |  | U | RU |  |  |  |  | D |  |
| Number |  |  |  |  |  |  |  |  |  |  |  | RU |  |  |  |  |  |  |
| TileItems |  |  |  |  | C |  |  |  |  |  |  |  | RD |  |  |  |  |  |
| ItemName |  |  |  |  | C |  |  |  |  |  |  |  | RD |  |  |  |  |  |
| GameName |  |  |  |  | C |  |  |  |  |  |  |  | RD |  |  |  |  |  |
| xValue |  |  |  |  | C |  |  |  |  |  |  |  | RD |  |  |  |  |  |
| Entity/Attribute | **Find Users** | **Register User** | **Log in** | **Log Out** | **Create Game** | **Find Game** | **Join Game** | **Find Game Players** | **Send message** | **See Messages** | **Use Item** | **Move** | **Pick up item** | **Admin Access** | **Delete Account** | **Edit Account** | **Delete Game** |  |
| yValue |  |  |  |  | C |  |  |  |  |  |  |  |  |  |  |  | D |  |
| Asset |  |  |  |  |  |  |  |  |  |  | R |  | R |  |  |  |  |  |
| ItemName |  |  |  |  |  |  |  |  |  |  | R |  | R |  |  |  |  |  |
| Effect |  |  |  |  |  |  |  |  |  |  | R |  | R |  |  |  |  |  |
| TileBomb |  |  |  |  | C |  |  |  |  |  |  | RD |  |  |  |  | D |  |
| GameName |  |  |  |  | C |  |  |  |  |  |  | RD |  |  |  |  | D |  |
| xValue |  |  |  |  | C |  |  |  |  |  |  | RD |  |  |  |  | D |  |
| yValue |  |  |  |  | C |  |  |  |  |  |  | RD |  |  |  |  | D |  |
| Damage |  |  |  |  | C |  |  |  |  |  |  | RD |  |  |  |  | D |  |
| PlayerItem |  |  |  |  |  |  |  |  |  |  | UD |  | CU |  | D | U | D |  |
| UserName |  |  |  |  |  |  |  |  |  |  | D |  | C |  | D | U | D |  |
| GameName |  |  |  |  |  |  |  |  |  |  | D |  | C |  | D |  | D |  |
| ItemName |  |  |  |  |  |  |  |  |  |  | D |  | C |  | D |  | D |  |
| Quantity |  |  |  |  |  |  |  |  |  |  | UD |  | CU |  | D |  | D |  |
| Game |  |  |  |  | C | R |  |  |  |  |  |  |  |  |  |  | D | U |
| GameName |  |  |  |  | C | R |  |  |  |  |  |  |  |  |  |  | D |  |
| MapName |  |  |  |  | C | R |  |  |  |  |  |  |  |  |  |  | D |  |
| GameActive |  |  |  |  | C | R |  |  |  |  |  |  |  |  |  |  | D | U |
| Map |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Name |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| xHome |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| yHome |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| xLength |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| yLength |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Chat |  |  |  |  |  |  | R |  | C | R |  |  |  |  | D |  |  |  |
| MessageID |  |  |  |  |  |  | R |  | C | R |  |  |  |  | D |  |  |  |
| Username |  |  |  |  |  |  | R |  | C | R |  |  |  |  | D |  |  |  |
| Text |  |  |  |  |  |  | R |  | C | R |  |  |  |  | D |  |  |  |

CRUD TABLE Update.

Added New Fields. Got rid of unneeded events such as having a separate event for admin creating player (can just use register user).

Added new events.

# Procedures and Transactions

All procedures were written with the GUI in mind and how the data needed to be accessed. SQL Statements will be needed in order to get lists(players online list, game players list, admin list etc.)

## Basic User Functionality

### FindUser

Checks to see if a user exists. The purpose of this procedure is to test if targeted user exists in the data base.

### RegisterPlayer

This Procedure registers a player. It checks if the username or email already exist as they need to be unique, and if they do not it will create a Player record with the received data. Will progress user to game screen if successful.

### LoginPlayer

Checks if the data entered on the login screen corresponds to a user. Then checks if the user account is locked. If password is wrong, it will increase login attempts and/or lock the account when exceeding the attempts and prompt the user to contact an admin with a provided email. If the login name and password matches, and the account is not locked it will take the user to the lobby screen.

### LogoutPlayer

Sets the current players online status to false. This will happen when game is exited.

### GetOnlineplayers

Users will be able to see a list of the players that are online.

## Admin Commands

### AdminAccess

The Admin storyboard screen will only be accessible by admins, so all the MYSQL commands that can be accessible from there don’t need to check if user is an admin. This procedure will check if current player is an admin and will act as a gate allowing entry to the admin console screen.

### EditUser

Will Take entered information on Admin screen and update the Player Table if data is acceptable.

### DeleteUser

This procedure will delete the Player. As the tables are setup with a CASCADE the corresponding tables of user data will be deleted.

### DeleteGame

Will delete a game and corresponding records from a list of games.

### DeactivateGame

Keeps game around but no user will be able to join the game.

### GetAllGames

This procedure will get a list of all games to display on the admin screen so the admin can select one and delete it.

### GetAllPlayers

This procedure will list all players for the admin screen so the admin can select one and edit/delete them.

## Game Creation

### CreateGame

This procedure will create a game based on the chosen map type, and the game name entered. It Creates all the tiles, bombs, and items that will be accessible to the game.

## Game Interaction

### GetActiveGames

This will return a list that the user will see on the lobby screen and they will be able to join these games.

### JoinGame

This procedure will add a Player to the Game Player table. It will then set up their home tile, randomize the player colour, and will change GUI screens to the game screen. If the Player has joined the game before it will add them back on the tile they were on when they left(If tile is in use by another player the user will only be activated when they move to an adjacent tile).

### MovePlayer(Handles picking up item)

This procedure will determine the valid moves. Players will click a tile on the GUI and this procedure will run. If the same tile is clicked any item on that tile will be added to the player’s inventory.

If another tile is selected and it is adjacent to the previous tile (x +-1, y+-1) Then the user will move to that tile making it visible. If a bomb is on that tile the user will hit it and take damage.

Move Player will set a player in game status to active after they if the user joins the game on an in-use tile so they will be able to use items.

### GetActiveGamePlayers

This procedure will be able to get a list of all the users that are in the current game and update them on the game screen.

### LeaveGame

This procedure will run when the player exists the game and returns to the lobby screen. It will set their status in game to not active, meaning players will not be able to use items on them, but they will be able to occupy the square the player was on.

### GetItems

This procedure will allow user to see their items and quantity in their inventory on the game screen.

### ConsumeItem

The Use Item procedure will consume the selected item from the player’s inventory. It will carry out that items effect and reduce the quantity by one. For the time being I have implemented **binoculars** which make adjacent tiles visible, **Gold** Which adds Score, And **Club** which will remove score from active players in the game and add score to the user.

## Chat

### SendMessage

This procedure will add the user’s text to the database along with their username and a new messageID

### GetChat

This procedure returns the last 10 chat messages in descending order which the user will be able to read in the chat box.

# ACID

*“ACID is an acronym that stands for Atomicity, Consistency, Isolation, Durability.” (Database.Guide 2016)*

These properties ensure that the database transactions are going to be processed reliably. It is concerned with how the database is going to react when part of a transactions fails and how it copes with that.

## Atomicity

Atomicity Guarantees the whole of a transaction. Good atomicity is where if only part of the transaction gets completed then all the changes should roll back.

This is important for example in my game when a series of statements occur when using an item. If the tiles for a game are made, but the db either crashes or there is an error while adding bombs and items, then that whole game is not going to function and will not be able to be fixed. In my procedures transactions ensure that changes are only committed to the database when all the procedure is executed successfully.

## Consistency

Consistency in a database ensures that all the data in the tables will adhere to the same rules including constraints, cascades, triggers etc.

In my procedures the parameters passed in adhere to the constraints for the tables edited, and no invalid data will be passed into the database.

## Isolation

Isolation Is where a transaction can is not interfered with another transaction. This includes things such as seeing data from an incomplete transaction with another transaction.

Until the transactions are committed in my database the data will not be availbile for other transactions.

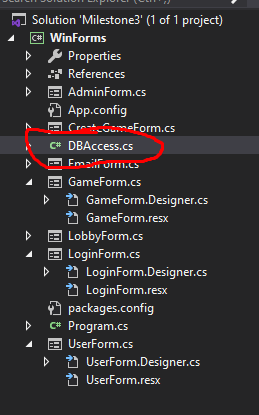
### Durability

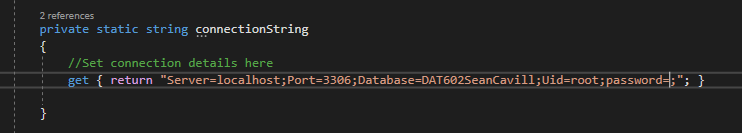
Durability ensured that any transaction that is completed will remain in the db system. This means that completed transactions are stored in non-volatile memory.

# Milestone 3

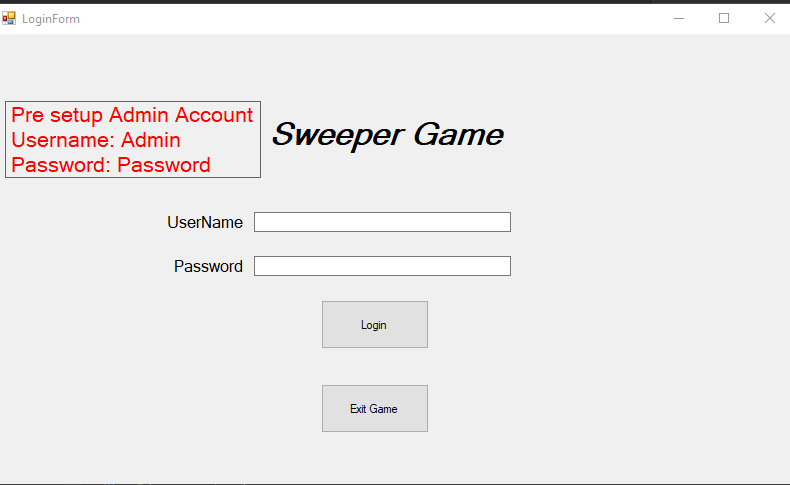
This is my submission for milestone 3. It is a windows form application that connects to a MySQL database using methods that are stored in the “DBAccess” class in c#.

To run this game you will need to run the included sql script, and in the DBAccess class in the winform application fill the password field in the connection details to your mysql password.



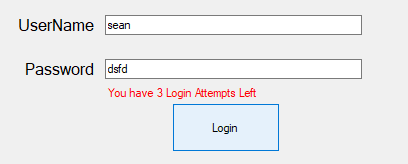


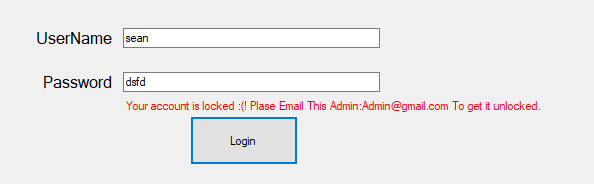
## Login Screen



User enters their username and password into the login screen to log in. The login procedure that is called will log in the user if successful. If the login is successful, the game moves onto the **LobbyForm**

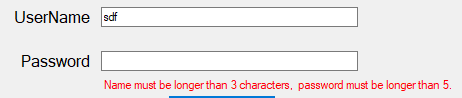
. If the password is wrong, a label notifying login attempts will show up, and if maximum login attempts are exceeded then the account will be locked a random admin email will display.



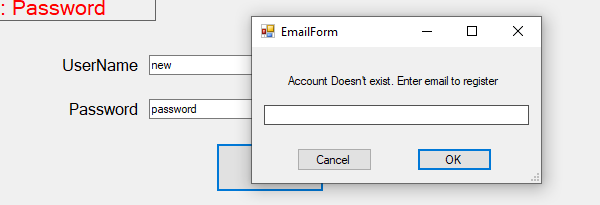


Note. When the account is locked, even a valid input will not allow login and the account must be unlocked through the admin form (or an sql update directly)

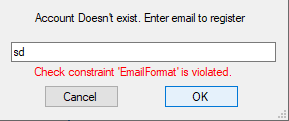
If the username does not exist in the db, the game will see if what were inputted would be valid for a registration. If not valid an error label will show



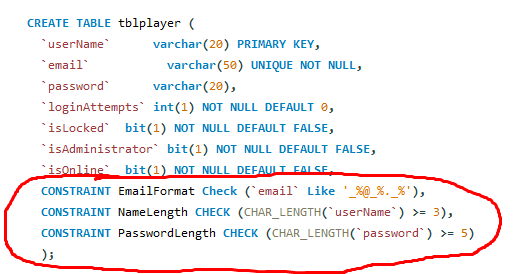
If the input is valid the email form will open.



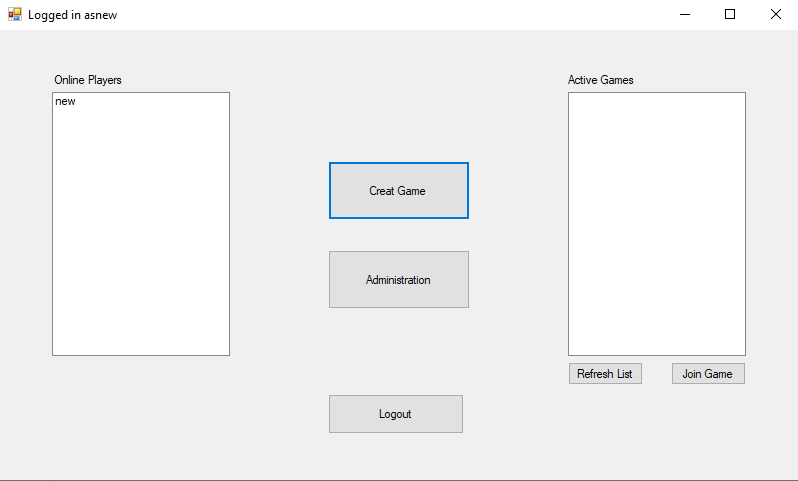
A valid email must be entered <text>@<text>.com, for this to accept and to register and login in the user. If the email is valid the **Lobby form** will open. Alternatively, the user can click cancel to return to the **login form.**



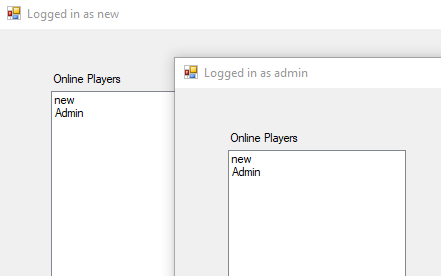
I needed to add constraints on the player table to achieve this. There is a constraint that checks email format, and constraints to make sure username is greater than 3 characters, and password greater than 5.



## LobbyForm

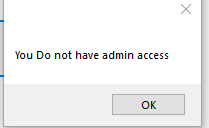


The lobby form lists all the currently online players.

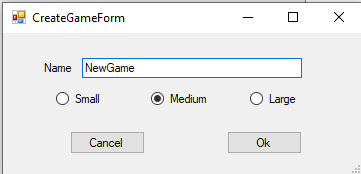


The user can click logout on the lobby form to logout their user (setting online status to false).

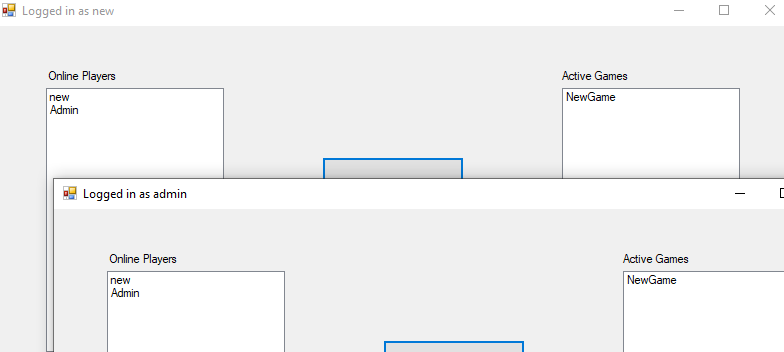
If the user is administrator, they can access the **AdminForm** through the administration button. If they do not have the required isAdmin == true then they will get this error.



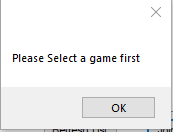
The players can also create games which brings up a create game form screen.



If no radio button is selected the default map will be small. If a duplicate game name is inputted an error label will show. If a radio button is selected, the map size will be dependent on that.

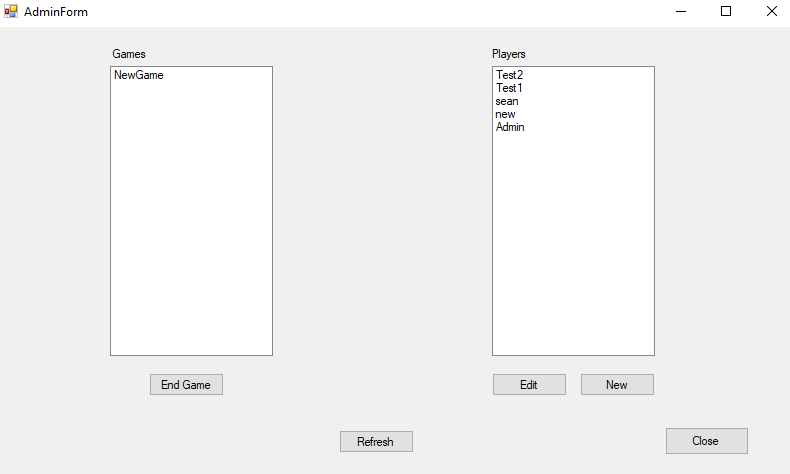


Active games is available on both screens. The players can join a game by clicking the join game button. If no game is selected, then this message will appear.



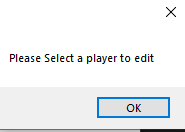
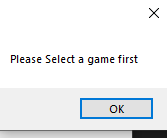
If a game is selected from the list box, then the **Game Form** will open.

## Admin Form

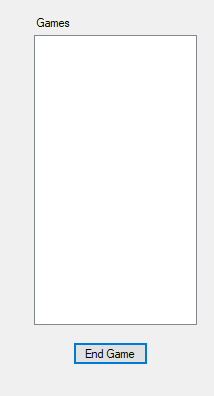


The admin form provides the functionality of ending games and editing and adding new players. If the user clicks the close button, they will return to the lobby form.

Just like in the lobby form, if no item from the respective listbox is chosen before clicking the end game or edit buttons, message boxes will appear notifying the user they need to make a selection.

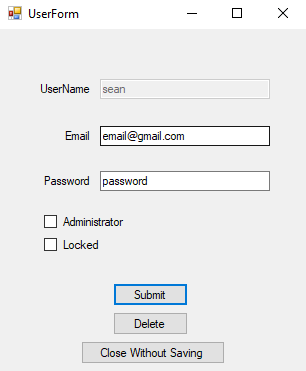


Selecting a game and clicking to end game will end the game and remove it from the list.

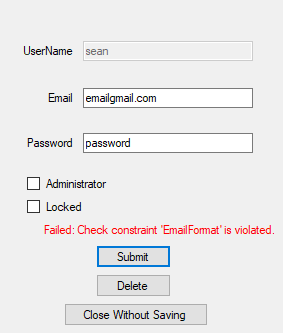


The admin has functionality to edit and to add new users. All players in the database are listed in the list box. Once the new button or edit button (with a selection) is clicked, the **User Form** will Open

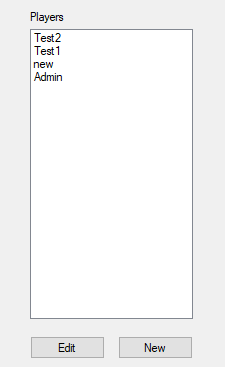
## User Form



This is the default screen when editing. The difference between this and a new user is the delete button is visible, and the username is not editable(originally had it so you could edit the username and the change would cascade, but decided this was cleaner and made more sense). The submit button will attempt to create the user(if user exists it updates it in the procedure instead of creating) if there are errors with the input an error will show.



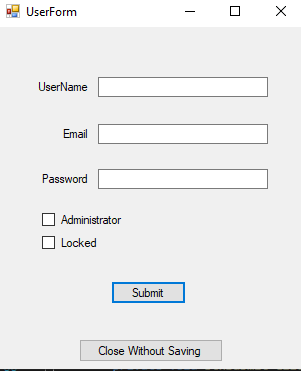
If the update is successful, the user will return to the admin form. If delete is clicked the selected account will be deleted and the user will return to the admin form. If close without saving is selected, the user returns to the admin form and nothing is changed.



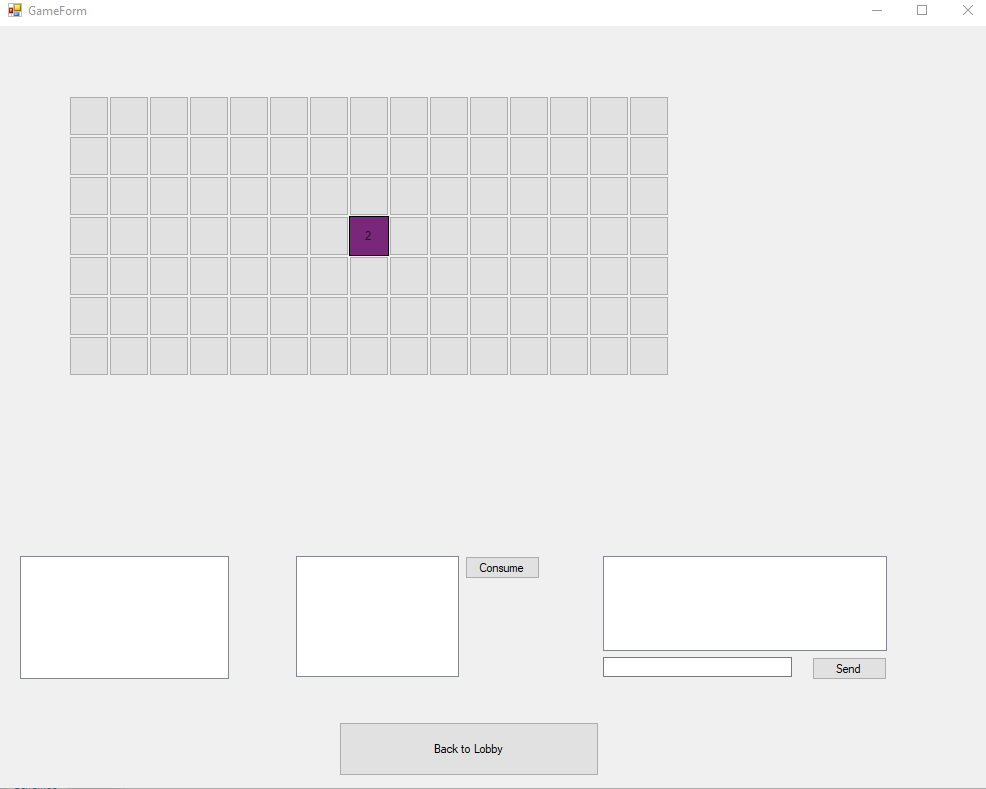
Clicked the delete button and “sean” has been deleted and removed.

Also note, if the player being editing is an admin, I have removed the option to remove them as admin.

If the new button is clicked instead of the edit, the same form opens, only the fields and blank, name can be inputted, and the delete button is not visible.

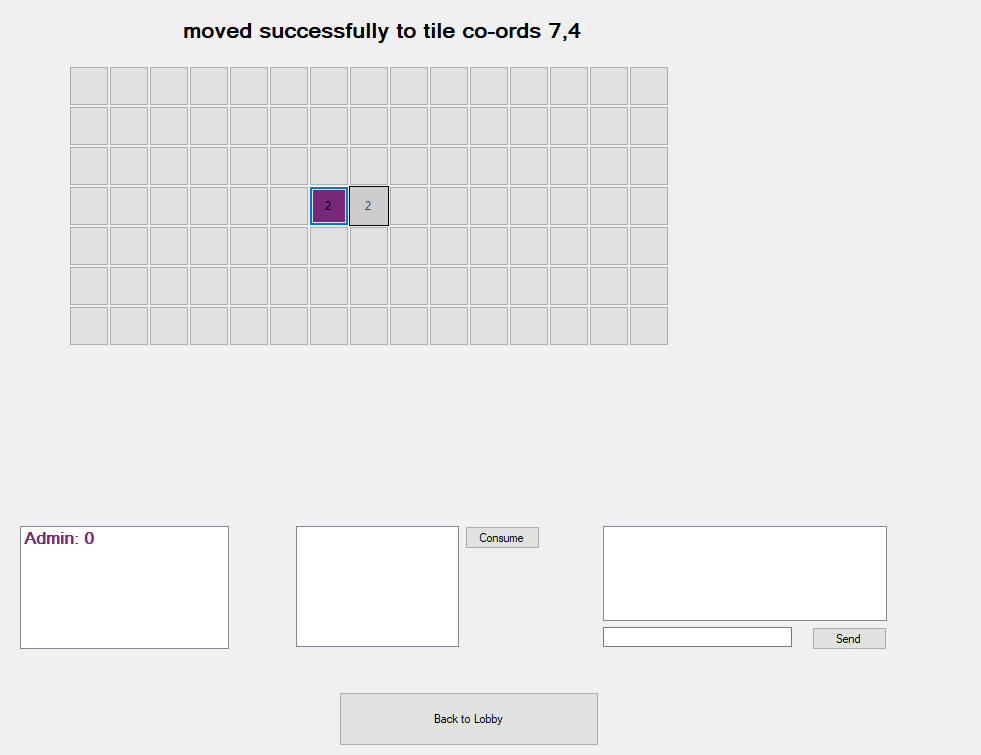


## Game Form

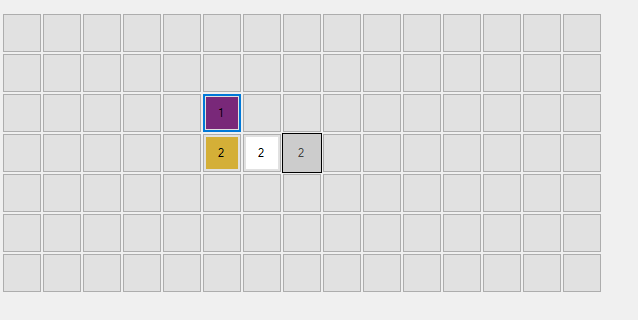


Upon joining the game, you are greeted with this screen. The size is changed depending on what map is chosen. This is the small map.

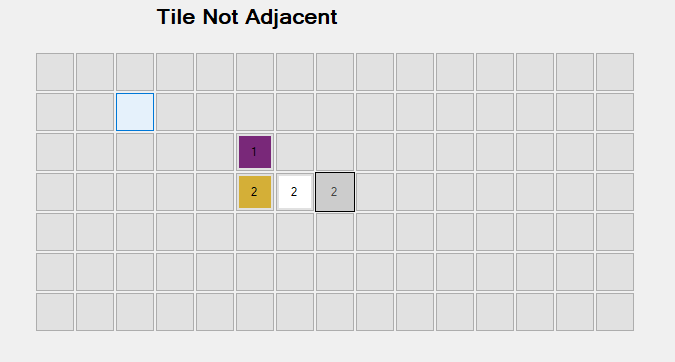
Currently the player is inactive until they make a move. Once they move to a valid square, they will show up on the player list. The players colour is random and will change between each game.



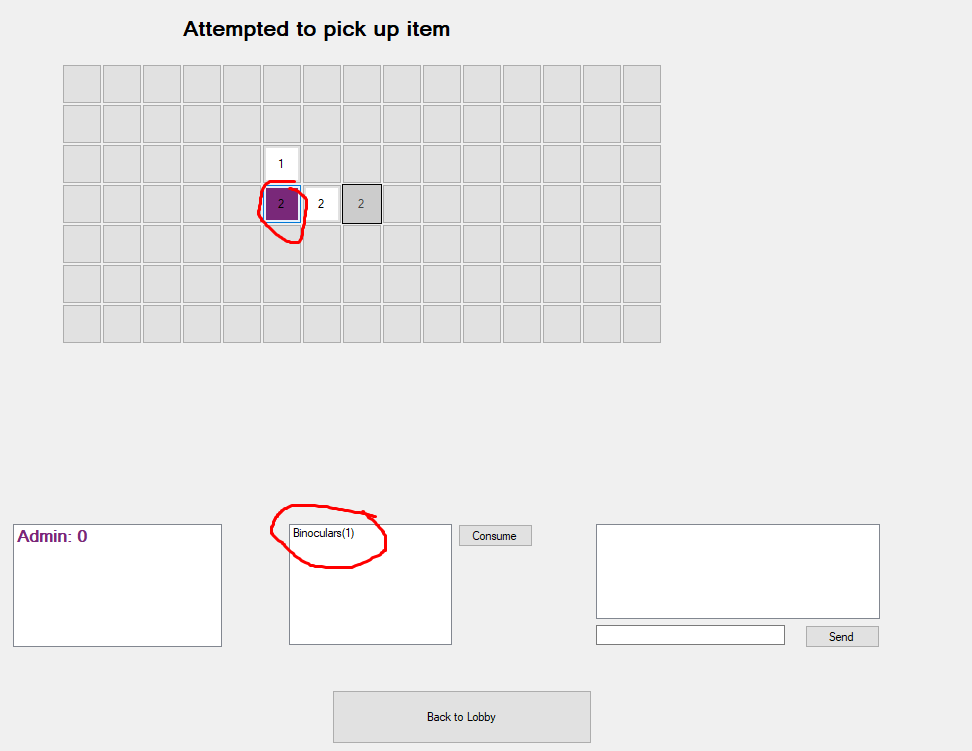
Moving message at the top of the screen. Notice the home square has a different design, and the user will not be able to move to it again once they have move off it.



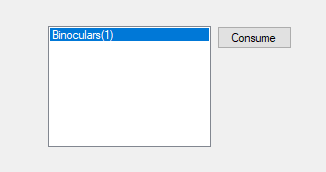
Squares with items on them will be gold. Squares that are not hidden are white. The numbers in the squares represent how many bombs are adjacent to that square.



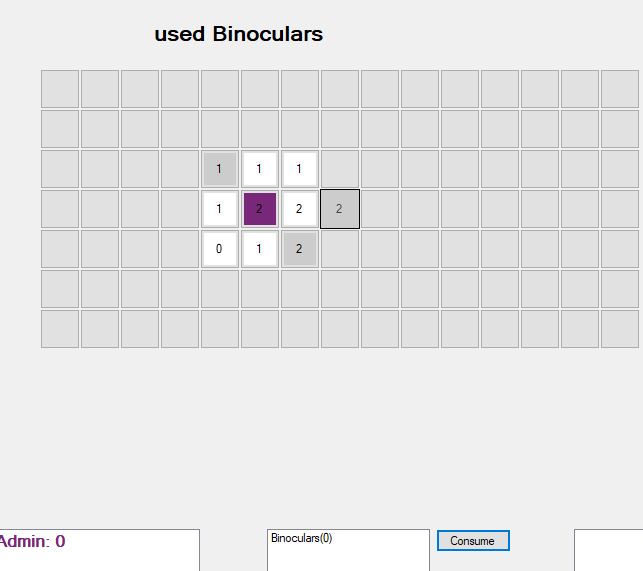
Non-adjacent tile clicked and player was not moved.



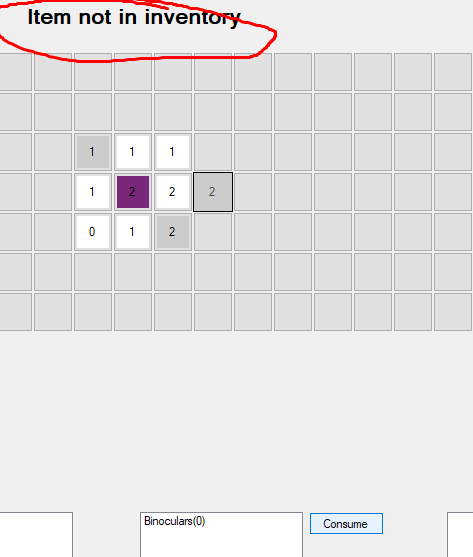
Clicking on a tile the user already resides on will attempt to pick up the items on that tile. If there is an item to be picked up it will add to the players inventory with the quantity. I have picked up the binoculars, they will make adjacent tiles visible.



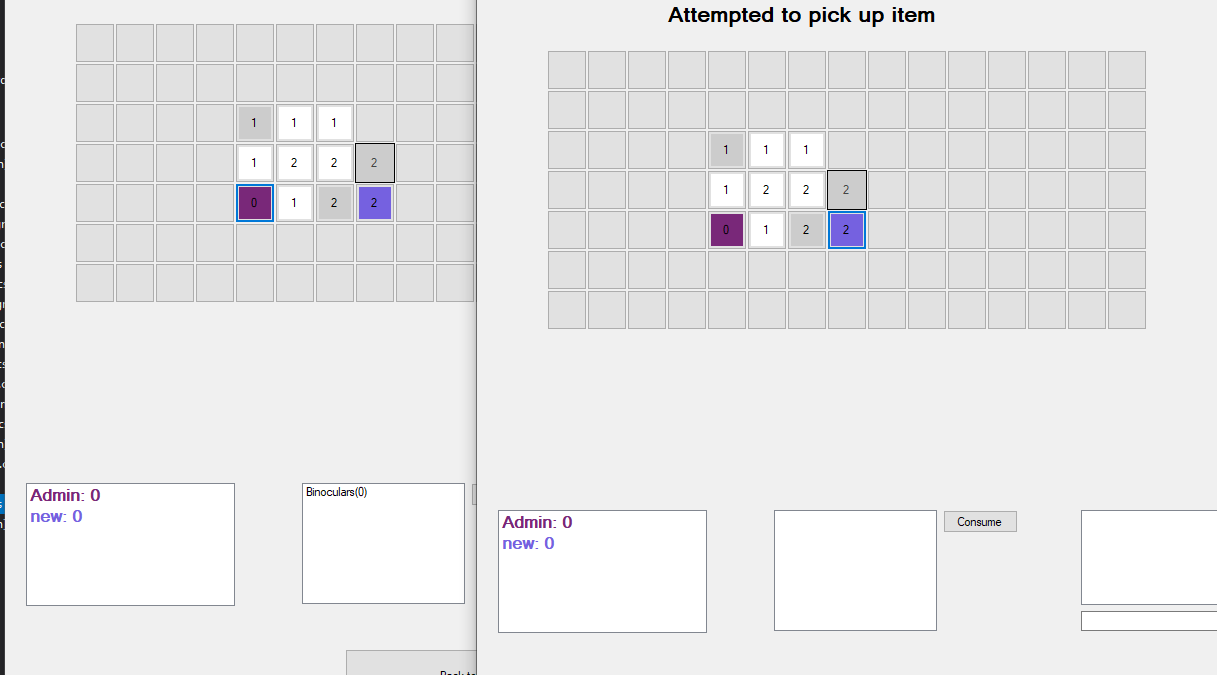
Select from list view and click consume



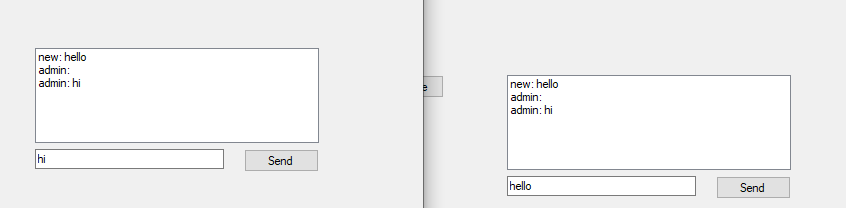
Grey squares are the ones with bombs on them. Note, binoculars amount has gone to 0 and the item can’t be used again, but it is now unlocked for the user to see the name.



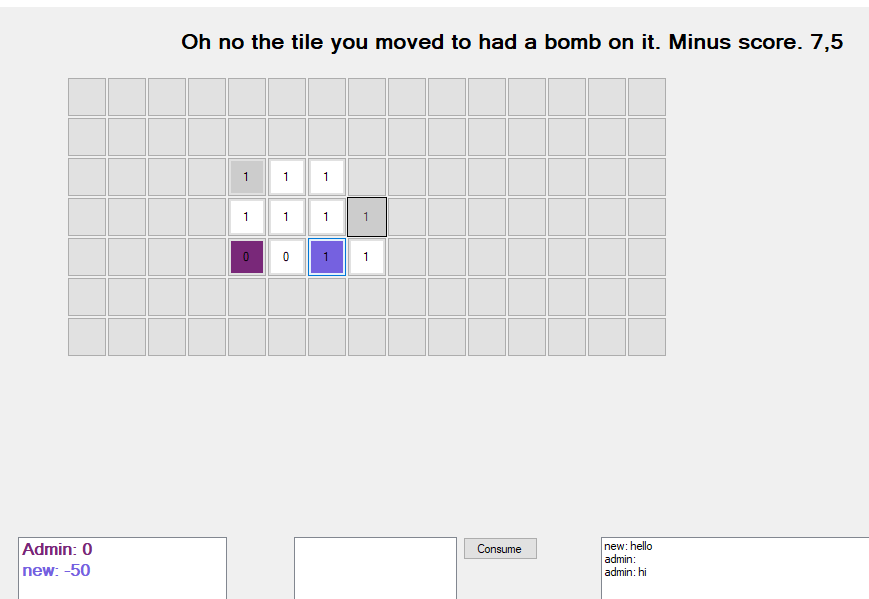
I will now join another player into the game.



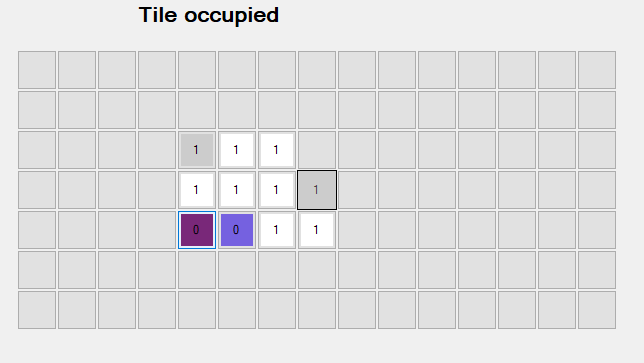
User new is now in the game.



Users can enter a message and press send to chat.



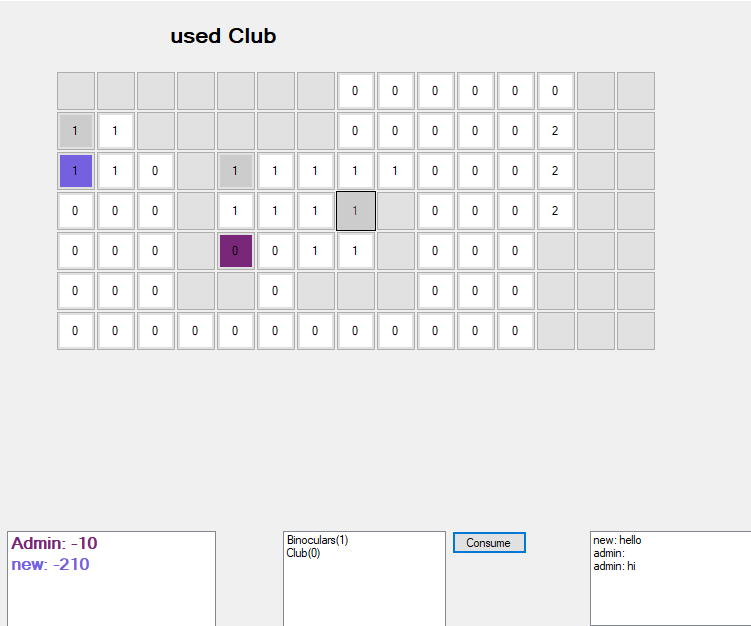
If a user moves on to the bomb they are met with a message and their score is deducted by the damage the bomb does (note the leader board on bottom left is sorted ascending by score).

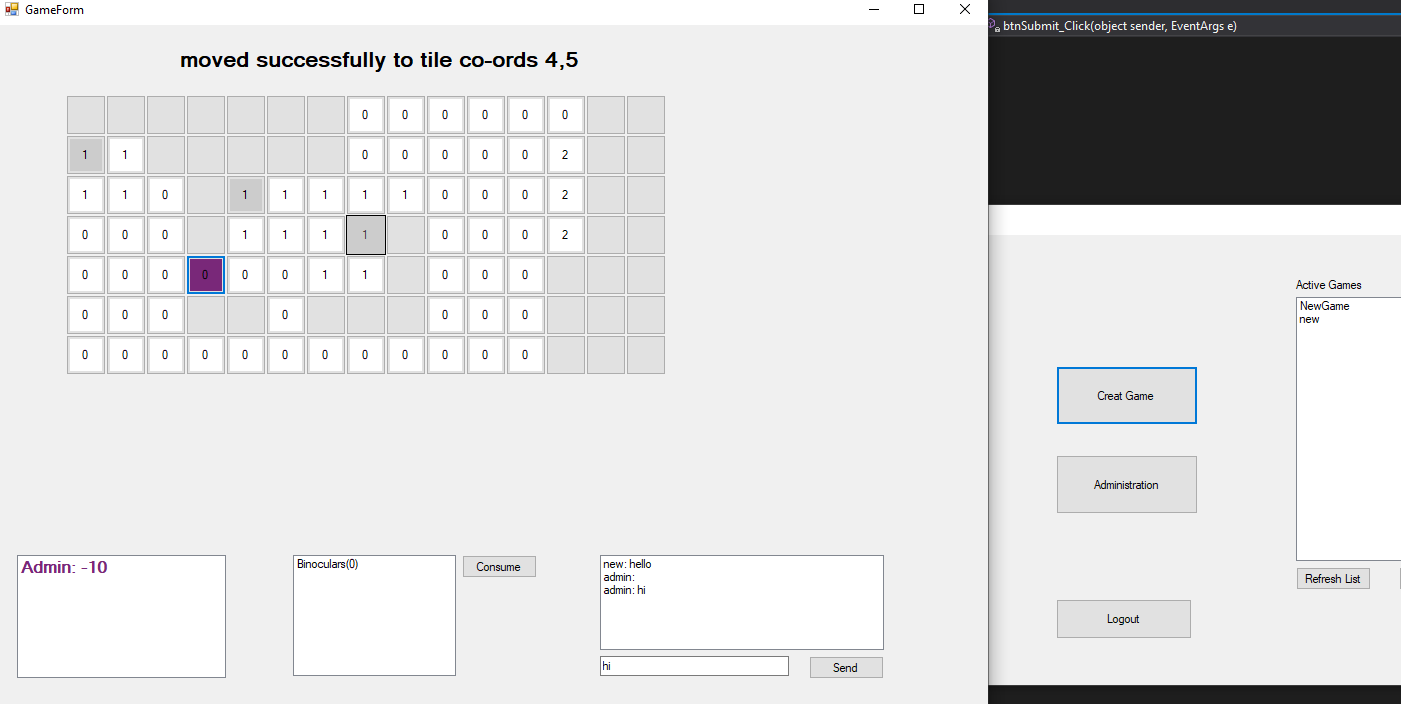


Can’t move to a square another user is on.

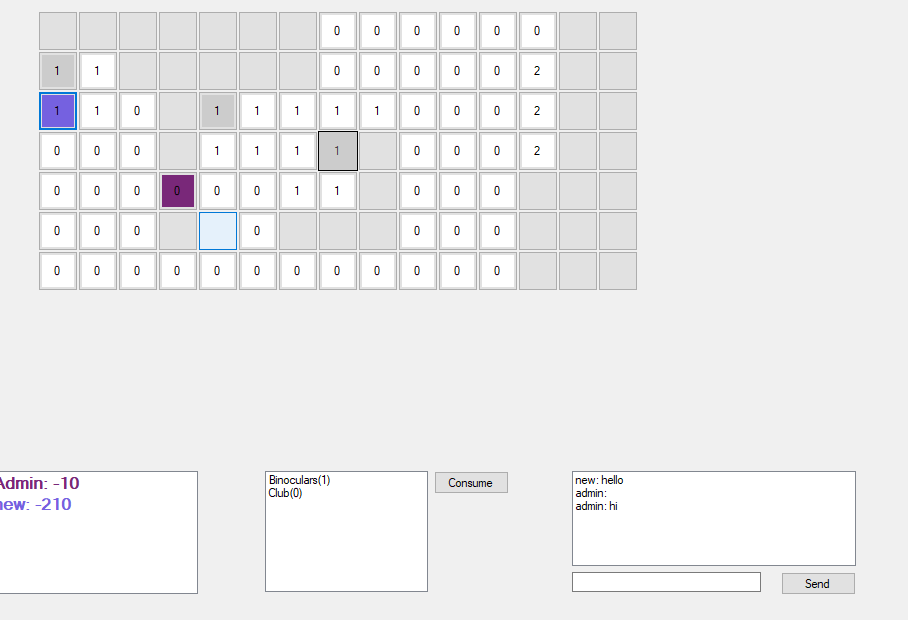


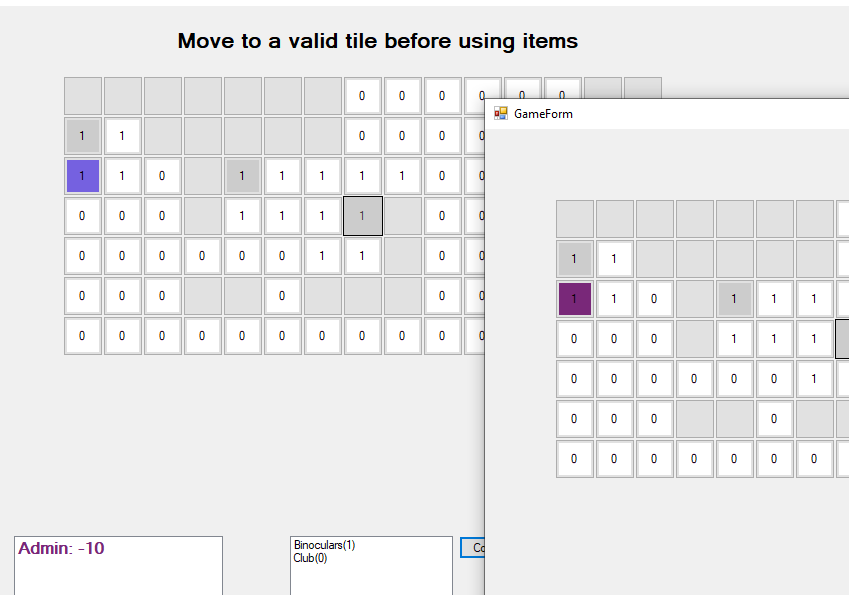
There are three items in the game. Binoculars uncovers tiles. Club minuses score from active players and adds to the user’s score. And gold just adds to the user’s score.



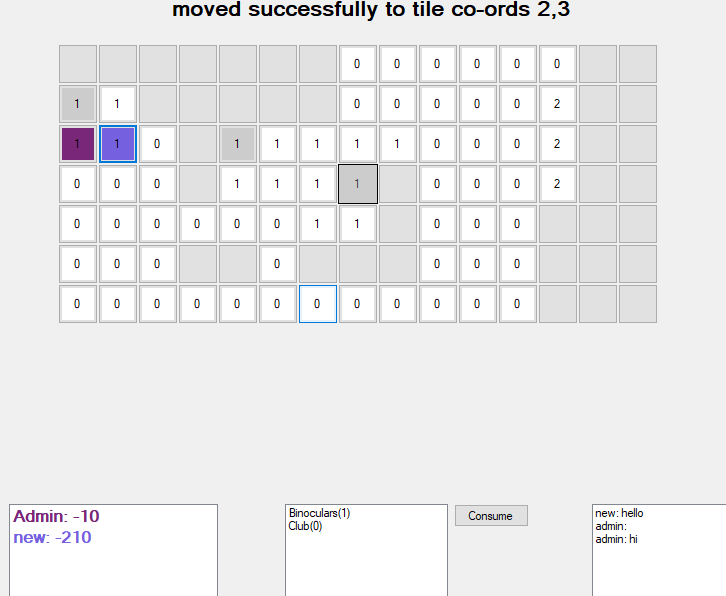


User left the game.



User re-joins the game and has their same score, items, and position. 

If the user leaves and rejoins the game and another user is on the tile they were on, they will not be able to use items or be active in the game until they move to a valid adjacent tile from where they were.



To make sure the users square was highlighted even if they weren’t active, I had to make a new sql procedure that returned their co-ordinates in game regardless of it they were active or not for a specific user, because I was having a problem where you would lose where your loser was and it was very difficult to find them again.

## Conclusion

I have learnt a lot in carrying out this assignment. From database CRUD/procedures/transactions, to connecting mysql to a c# application, and using methods to deploy a multiplayer game system.

## REFERENCES

Ian. (2016, June 20). *What does ACID mean in Database Systems? | Database.Guide*. Database.Guide. <https://database.guide/what-is-acid-in-databases/>