DAT602

Assignment 2

Regan Zohrab

2021

Contents

[1.0 Game Description 3](#_Toc68200477)

[1.1 Base Idea 3](#_Toc68200478)

[1.2 Logging in 3](#_Toc68200479)

[1.3 Game Lobby 3](#_Toc68200480)

[1.4 Gameplay 3](#_Toc68200481)

[1.4.1 Objective and Scoring 3](#_Toc68200482)

[1.5 Administrators 3](#_Toc68200483)

[2.0 Storyboards, Screen Design and Rational 4](#_Toc68200484)

[2.1 Login/Registration Screen 4](#_Toc68200485)

[2.2 Character Selection 6](#_Toc68200486)

[2.3 Game Screen 7](#_Toc68200487)

[2.4 Admin Screen 8](#_Toc68200488)

[2.5 Update player info screen 9](#_Toc68200489)

[3.0 Entity Relationship Diagram 10](#_Toc68200490)

[3.1 Entity Relationship Diagram Rational 10](#_Toc68200491)

[3.1.1 Player, Player\_Session, Session 10](#_Toc68200492)

[3.1.2 Player, Player\_Character, Character 10](#_Toc68200493)

[3.1.3 Character, Character\_Tile, Tile 10](#_Toc68200494)

[3.1.4 Tile, Game 11](#_Toc68200495)

[4.0 CRUD Table 12](#_Toc68200496)

[4.1 CRUD Analysis 12](#_Toc68200497)

[Check Username in Database. 13](#_Toc68200498)

[Create New User 13](#_Toc68200499)

[Check password against Username. 13](#_Toc68200500)

[Lock Account 13](#_Toc68200501)

[Successful login/Lobby Loads 13](#_Toc68200502)

[Start new Game 13](#_Toc68200503)

[Join existing game. 13](#_Toc68200504)

[Player Moves 13](#_Toc68200505)

[Game Ends 13](#_Toc68200506)

[Player logs off. 13](#_Toc68200507)

[Open Admin Console 13](#_Toc68200508)

[Admin Kills current game. 14](#_Toc68200509)

[Admin deletes player. 14](#_Toc68200510)

[Admin creates new player. 14](#_Toc68200511)

[Admin edits existing player info 14](#_Toc68200512)

[5.0 SQL 14](#_Toc68200513)

[5.1 DDL 14](#_Toc68200514)

[5.1.1 Player Table 14](#_Toc68200515)

[5.1.2 Player\_Session Table 14](#_Toc68200516)

[5.1.3 Session Table 14](#_Toc68200517)

[5.1.4 Game table 15](#_Toc68200518)

[5.1.5 Tile Table 15](#_Toc68200519)

[5.1.6 Character\_Tile table 15](#_Toc68200520)

[5.1.7 Character Table 15](#_Toc68200521)

[5.1.8 Player\_ Character table 15](#_Toc68200522)

[5.2 Test Data and Queries 16](#_Toc68200523)

Milestone 1

# 1.0 Game Description

## 1.1 Base Idea

Jake and Blake Snake is a turn-based take on the game snake. Players choose either Jake or Blake and take turns moving the snakes across the map to eat the Juicy Nugget. Players must make a move within five seconds or it’s the next players turn. When a player eats the Juicy Nugget, their character grows by one square. Once a player has eaten 5 Juicy Nuggets, the player wins the game.

## 1.2 Logging in

When players enter the game, they are presented with a login screen. Existing players will be able to enter the game by entering in their username and password. When the player clicks the submit button the username and password will be checked against the database. If the username and password are entered incorrectly, they will be locked out. If a new player wishes to play the game, they click the new user button. They are then taken to the registration screen where they can choose a username and enter in a password, they enter in an email address then click the button create. The new player is then entered into the database and then taken to game lobby.

## 1.3 Game Lobby

Players choose either Jake or Blake as a character. Players are then taken to the game map where they will be placed with a waiting player who has chosen the other character or placed on the map to wait for another player.

## 1.4 Gameplay

Players will be given a 5 second count down and then the game begins. One player will be randomly chosen to start, they will have five seconds to choose the direction in which their charter moves, it is then the other players turn.

### 1.4.1 Objective and Scoring

The objective for the players is to guide their snake towards the Juicy Nugget, once a player has eaten the nugget the players snake will grow by one square in length. Opposing players cannot cross the other players snake tail, so this can be used to trap other players. If a player touches the other players tail, the game will end with the other player winning. Also, the game will end once a player has eaten five Juicy Nuggets. Players wins will be recorded in the database.

## 1.5 Administrators

Administrators are players with more privileges than other players. This will give them admin controls that allow them to kill existing games, edit player details such as passwords and usernames. They can also delete accounts and perform account unlocks for players who have been locked out.

# 2.0 Storyboards, Screen Design and Rational

## 2.1 Login/Registration Screen

|  |
| --- |
|  |
| 1. Enter Username  2. Enter Password  3. Register New User  4. Submit Username and Password to log in |
| I wanted the design on the screens and the game to be as simple as possible, and to be able to navigate without too many written instructions. The login screen consists of two boxes where the user can enter a username and password and then submit or click the new user button of they are new. |
|  |
| 1. Enter email address  2. Enter username  3. Enter password  4. Click button to create. |
| Going with the simple design, new players will enter the details needed in the text boxes then click create. |

## 2.2 Character Selection

|  |
| --- |
|  |
| 1. Jake Image  2 . Blake Image  3. Select Jake Button  4. Select Blake Button |
| This screen keeps stays with the same aesthetic. This screen shows the images of the characters that can be chosen (Jake or Blake). Images haven’t been decided on yet but can be added later. |

## 2.3 Game Screen

|  |
| --- |
|  |
| 1. Time Tag  2. Current Play Time  3. Image of Jake  4. Image of Blake  5. Jake moveable character  6. Blake moveable character  7. Juicy Nugget  8. Game grid |
| The game (map) screen consists of 3 main areas. The time, this shows the current time the game has taken. The player section that has the images of the two players. These images will light up when it is the players turn, this will change every 5 seconds. Finally, is the grid map. This is a 10x10 grid where the game will be played. I thought that having a map size of 10x10 would be big enough for players to use some strategy, but not too big so the games would take a long time to play. |

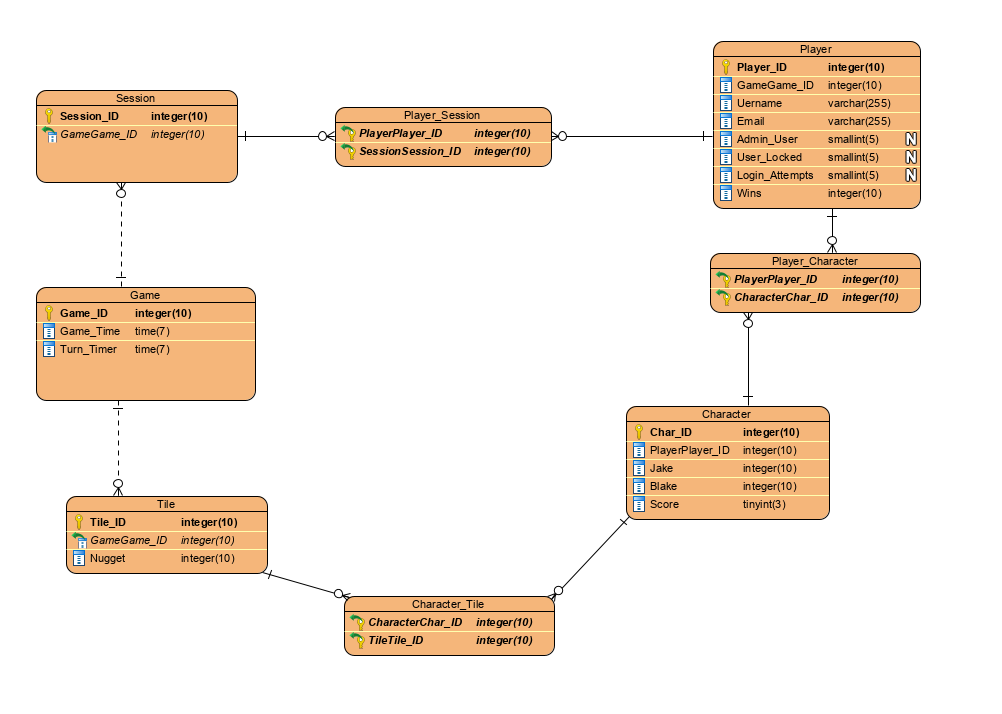
## 2.4 Admin Screen

|  |
| --- |
|  |
| 1. List of active players  2. Add new player button  3. Kill game Button  4. Remove player button  5. Update player info button. |
| The Admin screen will be only available to players who have an administrator status. The screen will show the list of connected players on the left of the screen. By selecting a player from the list, an administrator will be able to perform the functions indicated by the buttons. |

## 2.5 Update player info screen

|  |
| --- |
|  |
| 1. Player being updated  2. Total Wins  3. Player Username  4. Current Password  5. Email address. |
| The update player screen is where info on the players is updated, by entering in a new username, password or email and clicking the save button, the info will be saved. |

# 3.0 Entity Relationship Diagram



## 3.1 Entity Relationship Diagram Rational

### 3.1.1 Player, Player\_Session, Session

The player table will hold information on the character such as account information and game stats. A session ID is then created. Because there can be many players that can join many game sessions a join table was created so the data can be normalized.

### 3.1.2 Player, Player\_Character, Character

The Player, Character relationship is the same as Player and Session relationship where many players can choose from many (2) character so a join table was created. The Player table contains two columns for the two selectable characters, the rows will contain is the character is active, position on the grid, size, score and turn

### 3.1.3 Character, Character\_Tile, Tile

Another join table was created as many players can be on many tiles. The Tile table contain information on the tiles, this contains 1 column which is Tile\_ID. The rows will contain what tiles are activated when a character moves onto them, also which tile contains the Nugget, and what tiles will be set as the home (starting) tiles.

‘

### 3.1.4 Tile, Game

The game table consists of a game ID that contains total time, turn timer. The tiles are then associated with the game.

# 4.0 CRUD Table

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Entity/Attribute | **Check Username in Database** | **Register New User** | **Check Password against Username** | **Lock Account** | **Successful Login/Lobby Loads** | **Start New Game** | **Join Existing Game** | **Player Moves** | **Game Ends** | **Player Logs Off** | **Open Admin Console** | **Admin Kill Running Game** | **Admin Delete Player** | **Admin Creates New Player** | **Admin Edits Existing Player Info** |
| **Player** | R | C |  |  | R | R | R |  | RU | RU | R |  | D | RC | RU |
| Player ID |  | C |  |  |  | R | R |  |  | R | R |  | D | C | R |
| Username | R | C | R |  | R |  |  |  |  |  | R |  | D | RC | RU |
| User Password |  | C | R |  |  |  |  |  |  |  |  |  | D | C | RU |
| Email |  | C |  |  |  |  |  |  |  |  |  |  | D | C | RU |
| Locked User | R | C |  | U |  |  |  |  |  |  |  |  | D | C | RU |
| Admin User |  | C |  |  | R |  |  |  |  |  |  |  | D | C | RU |
| Login Attempts | R | C | RU |  | U |  |  |  |  |  |  |  | D | C | RU |
| Wins |  | C |  |  |  |  |  |  | RU | RU |  | RU | D | C | RU |
| **Player/Session** |  |  |  |  |  | R | R |  |  |  | R |  |  |  |  |
| Player/Player ID |  |  |  |  |  | R | R |  |  |  | R |  | D | C |  |
| Session/Session ID |  |  |  |  |  | C | R |  | U | RU | R | U | D |  |  |
| **Session** |  |  |  |  | R | C | R |  |  |  | R | U | D |  |  |
| Session ID |  |  |  |  | R | C | CR |  | RU |  | R | U | D |  |  |
| **Game** |  |  |  |  | R | C |  |  |  |  | R |  |  |  |  |
| Game ID |  |  |  |  | R | C | R |  |  |  | R |  |  |  |  |
| Session/Session ID |  |  |  |  | R | C | R |  |  |  | R |  |  |  |  |
| Game Time |  |  |  |  | R | C | R | U | U |  |  |  |  |  |  |
| Turn Time |  |  |  |  | R | C | R | U | U |  |  |  |  |  |  |
| **Tile** |  |  |  |  |  | C |  |  |  |  |  |  |  |  |  |
| Tile ID |  |  |  |  |  | C | R | U |  |  |  |  |  |  |  |
| Game/Game ID |  |  |  |  |  | C | R |  |  |  |  |  |  |  |  |
| Nugget |  |  |  |  |  | C | R | U |  |  |  |  |  |  |  |
| **Character/Tile** |  |  |  |  |  | C |  |  |  |  |  |  |  |  |  |
| Character/Char ID |  |  |  |  |  | C | CR |  | R |  |  |  |  |  |  |
| Tile/Tile ID |  |  |  |  |  | C | R | U |  |  |  |  |  |  |  |
| **Character** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Char ID |  |  |  |  |  | R | R |  |  |  |  |  |  |  |  |
| Player/Player ID |  |  |  |  |  | R | R |  |  |  |  |  | D |  |  |
| Jake |  |  |  |  |  | R | R | U | U |  |  |  |  |  |  |
| Blake |  |  |  |  |  | R | R | U | U |  |  |  |  |  |  |
| Score |  |  |  |  |  | R | R | U | U | U |  | R | D |  |  |
| **Player/Character** |  |  |  |  | R | R | R | U | U |  |  |  |  |  |  |
| Player/Player ID |  |  |  |  | R | R | R | U | U | U |  |  | D |  |  |
| Character/Char ID |  |  |  |  | R | R | R | U | U | U |  |  |  |  |  |

## 4.1 CRUD Analysis

The CRUD analysis was a logical ERD was created, which contained the entities and attributes that are going to be in the database. Then an analysis of the storyboards and project requirements was done to create the CRUD table and analysis.

### Check Username in Database.

This is a simple function that checks for right fields in the player table. The player will be asked to register if no record of the username can be found, if the username is registered it will check to see how many times the player has tried to log in.

### Create New User

This event will handle the creation of a new player, an UPDATE statement would be needed to create the information needed in the player table.

### Check password against Username.

The Database is designed to display three outcomes when the player tried to log in. In the first scenario the player logs in successfully. The second is when the player fails to make a successful login attempt, the third is where the players account is locked due to too many login attempts.

### Lock Account

When the players account is locked, the players locked account status will be set to true, it will then need an admin to unlock the account.

### Successful login/Lobby Loads

With this event there are a lot of moving parts. The players login attempts need to be reset upon login; the account needs to be checked for admin rights. On the game screen the timers will need to be displayed, along with wins and characters set to their starting positions.

### Start new Game

New game IDs will be created. Records for tiles will be updated as well as the nugget position and starting locations for players. Timers will be activated, and a player character is randomly chosen to start.

### Join existing game.

In this event a player is inserted into a game where a player is already waiting. The database needs to check for games that are waiting for characters to join and that have a different character than the player waiting in the map. Then insert that player into the map.

### Player Moves

A player will be randomly chosen to start. The game timer and turn timer are initiated. The player will have five seconds to make a move then it is the other players turn. The players character will be able to move one square in any direction on the map. The score will be updated when a character reaches the nugget and the map is then reset.

### Game Ends

The game will end once one player reaches five points or the game is terminated by an admin. The win or loss will be registered in the player records.

### Player logs off.

When a player logs off their win record will be updated, and online status will be terminated.

### Open Admin Console

Administrator status will be checked. Admins will be able to open the console and see open games and update player information.

### Admin Kills current game.

The administrator has the option of killing a live game. This will act as a game ending naturally and the database will update accordingly.

### Admin deletes player.

This will trigger a delete cascade on the player table records and kill any sessions that the player is currently active.

### Admin creates new player.

The admin can create a player by clicking on the create player button in the console. The database will check the username is available and then create the new player.

### Admin edits existing player info

The admin will retrieve the players info. If username is being changed, the Database will check that its available. If there is no error, the players info will be updated.

# 5.0 SQL

## 5.1 DDL

### 5.1.1 Player Table

**Player ID**

This was created to set the primary key, and to be used as a foreign key in other tables.

**Username**

Set as varchar to incorporate many characters for a username

**Password.**

Set to not null as required for account creation.

**Email**

Same as password.

**Admin user and User locked.**

These values can be set to Boolean as there only need to be true or false.

**Login Attempts**

Login attempts are set to tinyint as the number will not go over five.

### 5.1.2 Player\_Session Table

This table is crated to link the player and session tables in the database.

### 5.1.3 Session Table

**Session ID**

This table is created to establish a session and create a session ID that can be used in other tables.

### 5.1.4 Game table

**Game ID**

Created so the table has a primary key, that will be used as foreign keys in other tables

**SessionSession ID**

This is the foreign key from the Session table

**Game time**

The total time of the game, set as timer

**Turn timer**

Same as game time, but this timer will set the time for turns.

### 5.1.5 Tile Table

**Tile ID**

ID for each individual tile, also used as foreign key for other tables

**GameGame ID**

Foreign key from Game table

**Nugget**

Each tile can possibly have the nugget, will be set to Boolean as it only needs to be true of false.

### 5.1.6 Character\_Tile table

This is the same as player\_session table, was created to link the two tables Tile and Character

### 5.1.7 Character Table

**Char ID**

Set as primary key

**Jake, Blake, and turn**

Jake and Blake indicate which character have been chosen so only need to be true or false. Same as turn, set to Boolean.

### 5.1.8 Player\_ Character table

Join table created to link the player table and Character table.

## 5.2 Test Data and Queries

**Player table**

For the Player table I created 4 rows of player data, one with admin privileges and the rest standard players and 1 locked user. I used queries that would change a player’s status to locked, deleting a player, making a player an administrator.

**Player\_Session Table**

I got a bit stuck on this one as I still need to spend some more time gathering knowledge on how join tables work,

**Session Table**

For this I created 4 sessions, two set to active as these would have players the join the session, I created another for the purpose of deleting when a session has ended.

**Game Table**

For this table I created two game ID’s that link back to sessions and player. I used update queries to update times, and for the purposes of the assignment deleted times, but this would not be done ion the functional database, they would just reset.

**Tile Table**

For the tile table I created 100 rows of data, representing the game grid. I set five as active for each character (10 in total) and 1 tile for the nugget. An update query was run to change the nugget to another square. Again, in this table nothing would be deleted, but for the purposes of this assignment a tile was deleted and then returned.

**Character\_Tile Table**

Again, not sure on join tables

**Character table**

With this table the characters were both set to active and scores updated. A delete query was run to delete a character, this was done to fulfil the assignment requirements, no deleting would be needed in this table as character activation and score depends on player and session and the fields would update accordingly.

**Player\_Character Table**

Still need to study up on join tables.