

DAT602

Assignment

Mitchell Trow



TABLE OF CONTENTS

<i>Game Description</i>	<i>2</i>
<i>Storyboards.....</i>	<i>3</i>
<i>Interface design analysis</i>	<i>14</i>
<i>Conceptual ERD.....</i>	<i>15</i>
<i>Crud Table.....</i>	<i>16</i>
<i>Logical Design Altercations.....</i>	<i>17</i>
<i>CRUD TABLE UPDATES</i>	<i>18</i>
<i>SQL ProcedureS & FUnctions.....</i>	<i>19</i>
<i>Player registration and login</i>	<i>19</i>
<i>Player Selection and Game creation.....</i>	<i>19</i>
<i>Game communication.....</i>	<i>21</i>
<i>Game SPECTATION (MULTIPLAYER)</i>	<i>21</i>
<i>Administration functionality.....</i>	<i>21</i>
<i>ACID and Multiplayer support.....</i>	<i>22</i>
<i>TEST DATA.....</i>	<i>23</i>
<i>C# Implementation</i>	<i>23</i>
<i>References.....</i>	<i>24</i>

MILESTONE 1

GAME DESCRIPTION

The game is based off the concept of the moon landing. Players control a spacecraft as it descends toward the surface of the moon. Using thrusters, the player must direct the spacecraft to safely land on a flat area of terrain whilst battling the initial horizontal motion and gravity.

Each map has its own characteristics such as gravity and terrain difficulty and you are rewarded with a score bonus for it.

Each lander also has its own characteristics such as weight, thruster power and different centre of gravity. Depending on the difficulty of each, the score is also affected.



("Lunar Lander," 2017)

My version of the game implements the functionality of accounts, player spectating and a chat system.

STORYBOARDS

LOGIN/REGISTER

A storyboard for a 'Login or Register' form. The form is enclosed in a rectangular frame. At the top, the text 'Login or Register' is underlined. Below this, there is a text input field labeled 'Username'. A small box labeled '(a)' is positioned to the left of the input field. Below the input field, there is a 'Continue' button. A small box labeled '(b)' is positioned to the right of the button.

- a) Where a username is entered.
- b) When hitting continue the database checks if that user exists. If the user exists, the user is directed to the login password form. If they're not, they're directed to the register password form. If the username is not valid at all. They are directed to the login error form.

REGISTER PASSWORD

The diagram shows a web form layout. At the top is the title "Login or Register" followed by a horizontal line. Below this is a "Register" section, also followed by a horizontal line. Under the "Register" section is the text "Welcome. Enter password to continue". Below this text is a rectangular input field labeled "Password" with a small box labeled (a) to its right. Below the input field is a rectangular button labeled "Continue" with a small box labeled (b) to its right. The entire form is enclosed in a large rectangular border.

- a) Where the user enters the password they would like to use.
- b) After entering the password and clicking this, they will be added to the database granted that their password is a valid input, otherwise they are directed to the login error form. If their information is valid, they are sent to the character selection screen.

LOGIN PASSWORD

Login or Register

Login

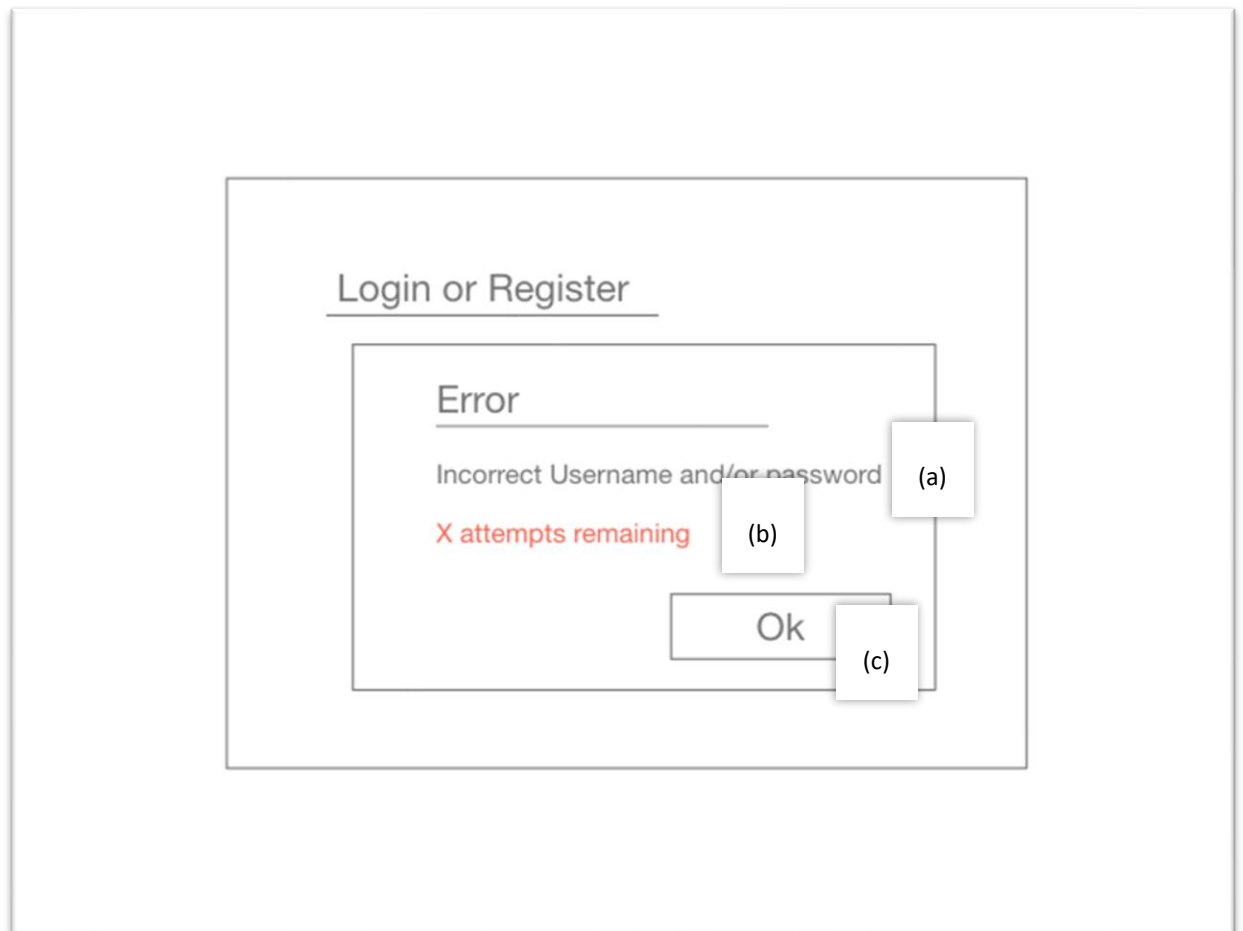
Welcome back <username>

Password (a)

Login (b)

- a) Where the user enters their password for their account.
- b) After entering a password, it is checked that it is valid, after the user is sent the character selection screen. If it is not, the user will be directed to the login error form. If the user fails too many times, their account is locked and sent to the form informing them so. If their account is locked, they are sent to the account locked screen.

LOGIN ERROR SCREEN



- a) Where the error information goes. This can be either:
 - Invalid username
 - Invalid password creation
 - Invalid password for user
 - Invalid username
- b) If it is an unsuccessful login because the password is incorrect, the attempts remaining is displayed. If they fail 5 times in a row, their account is locked.
- c) To go back and try again.

ACCOUNT LOCKED SCREEN

Login or Register

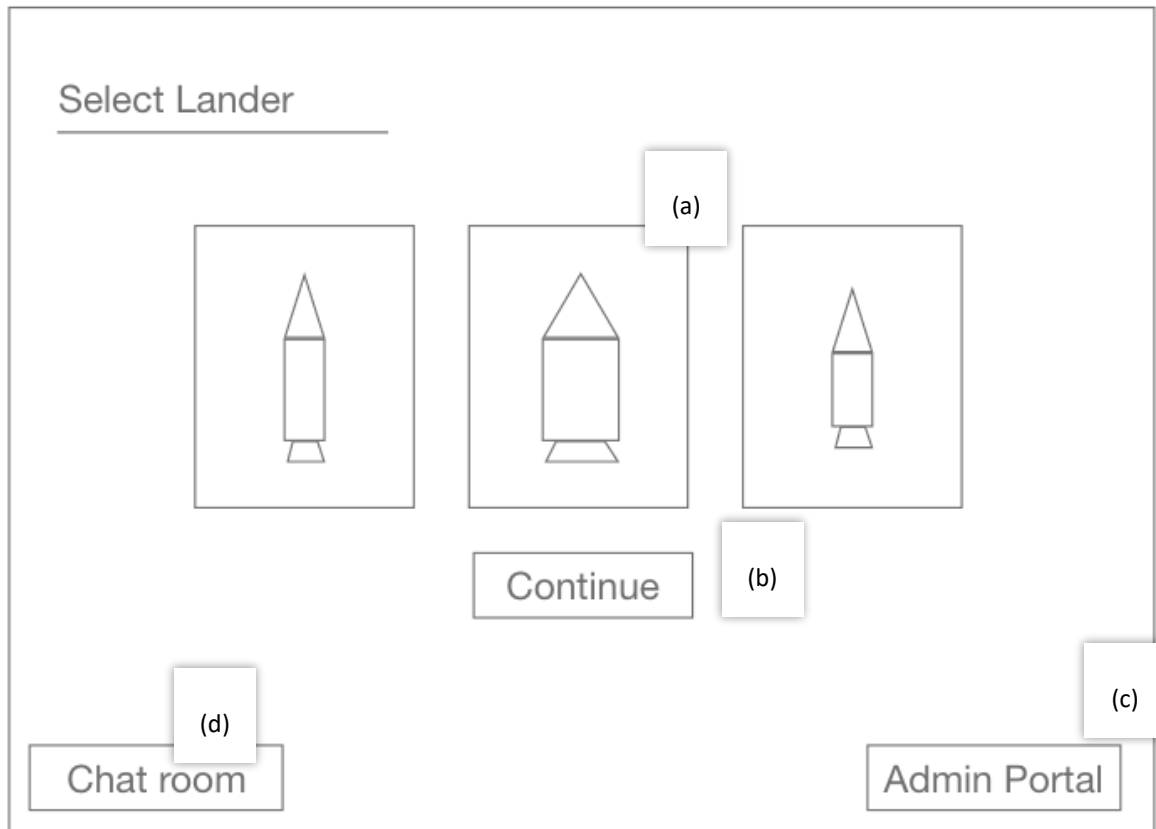
Error

Account is locked (reason) (a)

Ok (b)

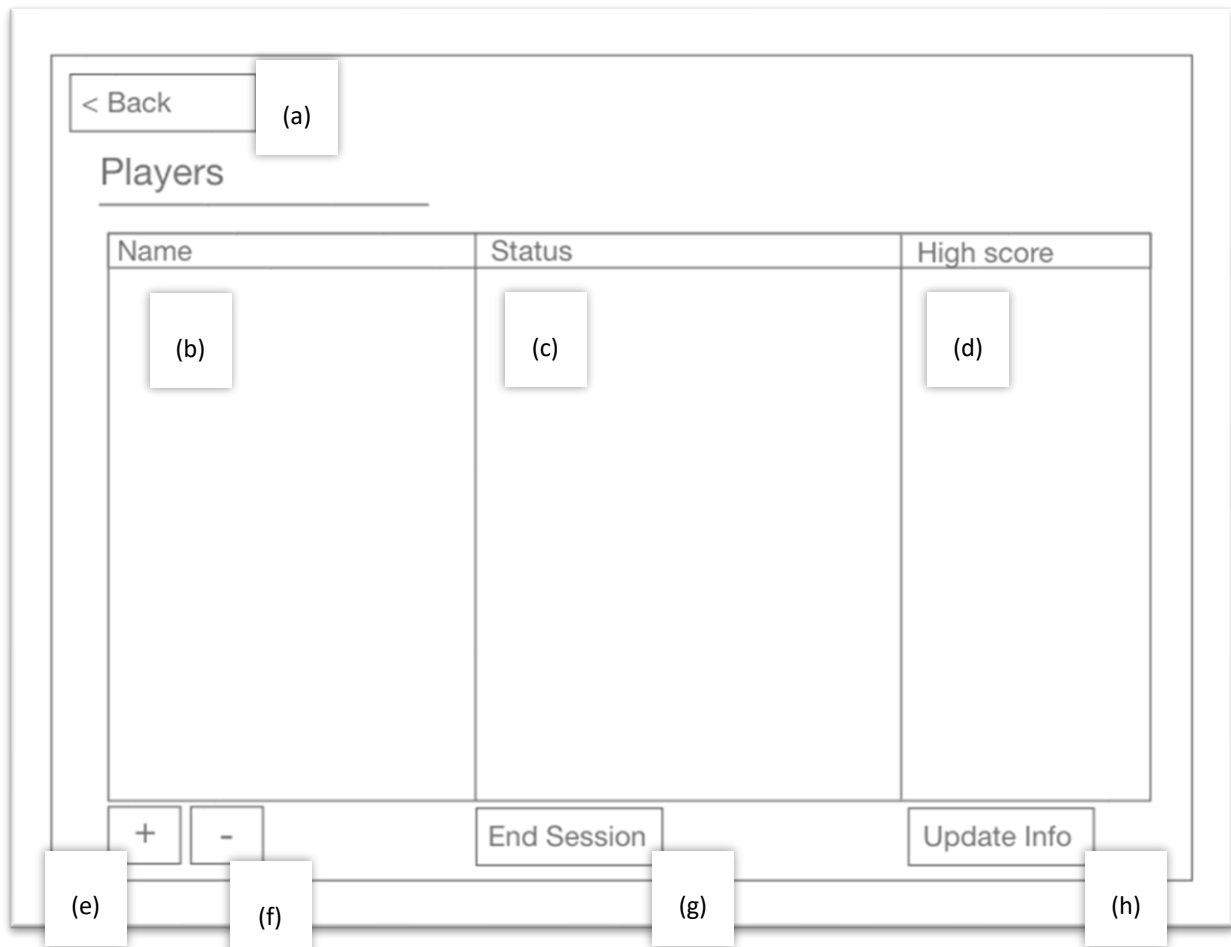
- a) The reason for their account being locked, either because they failed the login too many times or an admin locked them out.
- b) To go back to the login screen.

CHARACTER SELECTION SCREEN



- a) Where the user selects their lander they want to play as. Not limited to three as in the screenshot.
- b) After selecting their lander, this button will become enabled and when clicked will send them to the gameplay screen
- c) Only visible to admins, when clicked, they will be sent to the admin console.
- d) Sends the player to the global chat room

ADMIN CONSOLE



The image shows a wireframe of an 'Admin Console' for managing players. At the top left is a '< Back' button labeled (a). Below it is the title 'Players' with a horizontal line underneath. The main area is a table with three columns: 'Name', 'Status', and 'High score'. The first row of the table contains labels (b), (c), and (d) respectively. Below the table are four buttons: a '+' button labeled (e), a '-' button labeled (f), an 'End Session' button labeled (g), and an 'Update Info' button labeled (h).

Name	Status	High score
(b)	(c)	(d)

Below the table are the following controls:

- (e) +
- (f) -
- (g) End Session
- (h) Update Info

- a) Sends the user back to the character selection screen
- b) Lists the player names
- c) Lists the player statuses. Either:
 - Offline
 - Online
 - Online, In Chat room
 - Online, In-game (Lander name)
- d) Lists the player high-scores
- e) Adds a new user and sends the user to the admin new user dialog
- f) Removes the selected user
- g) Logs the current selected user out
- h) Sends the user and the currently selected user to the update info dialog.

ADMIN NEW USER DIALOG

The diagram shows a 'New User' dialog box with the following elements:

- Username:** A text input field containing the placeholder text '<username>' and labeled (a).
- Password:** A text input field containing placeholder text '*****' and labeled (b).
- Login Attempts:** A text input field containing the value '0' and labeled (c).
- Locked:** A checkbox labeled (d).
- Admin:** A checkbox labeled (e).
- Buttons:** 'Cancel' (labeled f) and 'Create' (labeled g) buttons are at the bottom of the dialog.

The background window features a '< Back' button, a 'Players' section with a 'Name' label, and buttons for '+', '-', 'End Session', and 'Update Info' at the bottom.

- a) New user username
- b) New user password
- c) New user login attempts (0 is default)
- d) New user locked status
- e) New user admin status
- f) Cancel and go back to admin form
- g) Create the user

ADMIN EDIT USER DIALOG

The diagram shows a modal dialog box titled "Edit <username> Info". The dialog contains the following elements:

- Username:** A text input field containing "<username>". A callout box (a) points to this field.
- Password:** A text input field containing "*****". A callout box (b) points to this field.
- Login Attempts:** A text input field containing "3". A callout box (c) points to this field.
- Locked:** A checkbox. A callout box (d) points to this checkbox.
- Admin:** A checkbox. A callout box (e) points to this checkbox.
- Buttons:** "Cancel" and "Update". A callout box (f) points to the "Cancel" button, and a callout box (g) points to the "Update" button.

The dialog is overlaid on a background window titled "Players". The background window has a "< Back" button at the top left and a list of players with a "Name" header. At the bottom of the background window are buttons for "+", "-", "End Session", and "Update Info".

- a) Edit user username
- b) edit user password
- c) edit user login attempts
- d) edit user locked status
- e) edit user admin status
- f) Cancel and go back to admin form
- g) Commit the changes

GLOBAL CHAT ROOM

< Back (a)

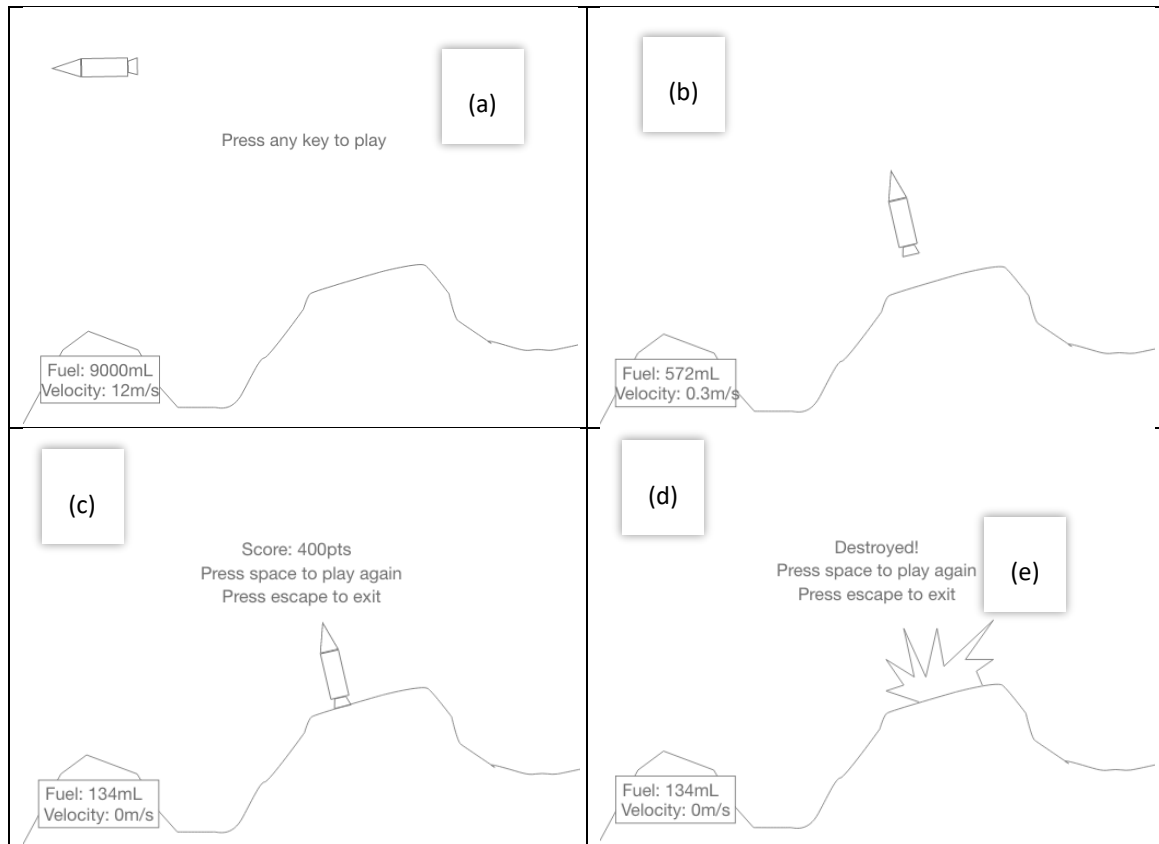
Global Chat

(Time)	Name	Message
(b)	(c)	(d)

(e) Send (f)

- a) Sends the player back to the character selection screen
- b) The time the player sent the message
- c) The name of the player
- d) The message the player posted
- e) The input field for sending the message
- f) The send button to send the message to the chat room

GAMEPLAY



- a) User presses any key to start the game.
- b) User uses a combination of space and the directional keys to control the lander
- c) If the user does not exceed the speed limit, angle limit and does not run out of fuel and lands on the terrain, they win and are given a score.
- d) If the user exceeds the speed limit, angle limit or runs out of fuel before landing, they craft will be destroyed on impact.
- e) Pressing escape will take the user back to the character selection screen. Pressing space will allow the user to retry.

INTERFACE DESIGN ANALYSIS

PLAYER REGISTRATION

The design was created to fit the design brief but also be as user friendly as possible. The brief called for a player to be able to login and or register on the same form. The forms are responsive to show feedback for the user's actions such as login failure or success.

CHARACTER SELECTION

This design was also created to fit the brief. The design is a simple character selection screen, showing the name, stats and other attributes for each lander the player can choose from.

GAME ADMINISTRATION

This form isn't as user friendly since it isn't designed to be seen by the average player. This screen is only for admins to access extra functionality relating to managing the game and the players. This screen was designed around the brief which called for the admin to be able to manage users and games.

GAMEPLAY

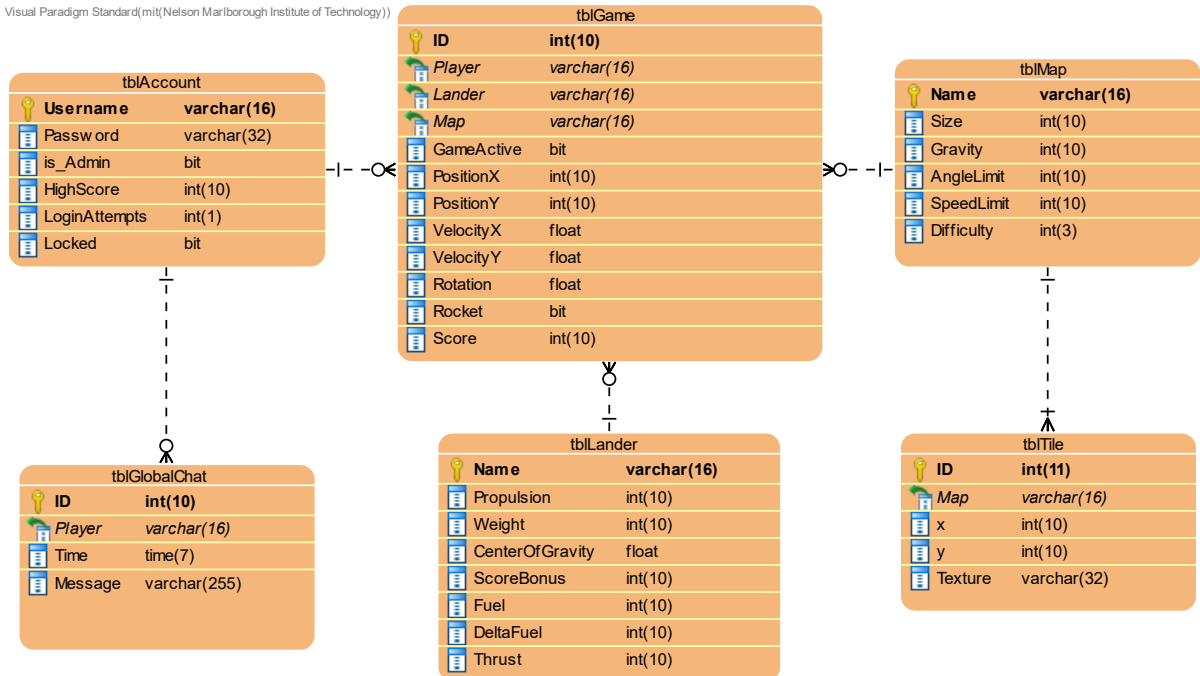
These diagrams are designed to fit the idea for the gameplay of the game. It shows the stages the player will experience when playing the game and the outcomes that follow it.

PLAYER GLOBAL CHAT

This section is for players to communicate with other online players. This section will update with the most recent messages. This section was created to fit the brief as it called for player communication.

CONCEPTUAL ERD

Visual Paradigm Standard(mit(Nelson Marlborough Institute of Technology))



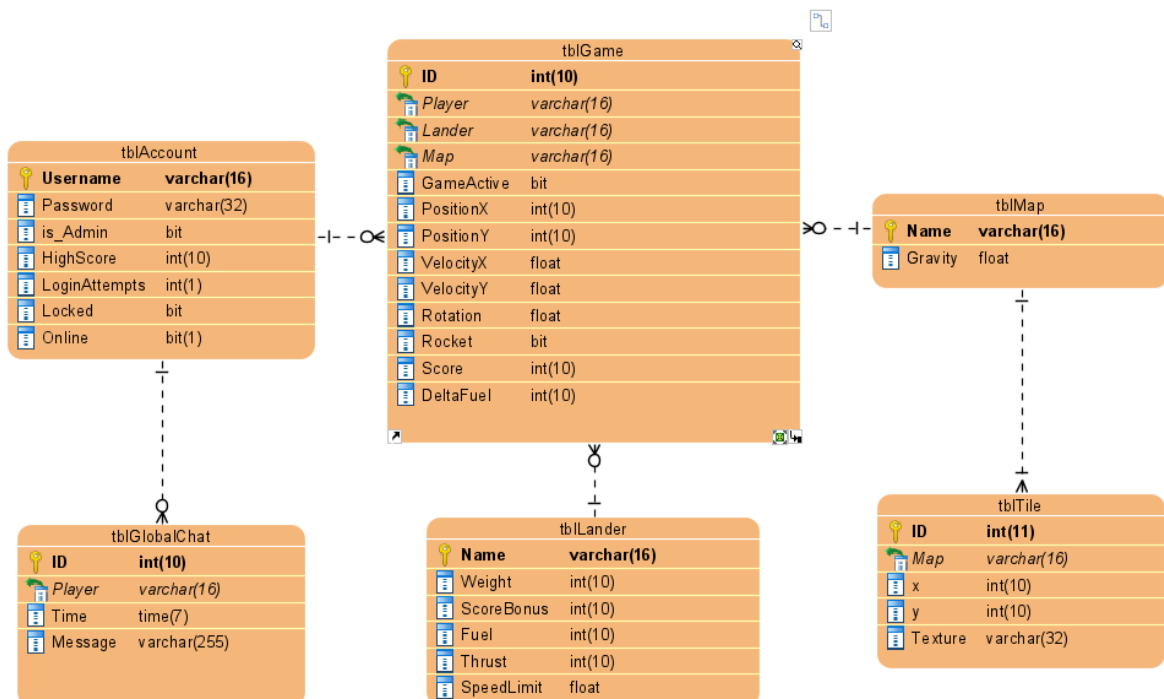
CRUD TABLE

	Player registers new account	Player attempts to log in	Player fails to log in	Successful Login	Start new Game	Update game frame	Game ends	Admin opens admin panel	Admin Deletes player	Admin updates player info	Player sends message	Player Receives message
tblAccount	C	R	RU	RU	R		RU	R	D	RU	R	R
Username	C	R			R			R	D	RU	R	R
Password	C	R							D	RU		
isAdmin	C			R				R	D	RU		
Highscore	C			R			RU		D	RU		
LoginAttempts	C		RU	U					D	RU		
Locked	C	R	U						D	RU		
tblGame					C	U	R	R	D			
ID					C			R	D			
Player					C			R	D			
Lander					C			R	D			
Map					C			R	D			
GameActive					C	U			D			
PositionX					C	U			D			
PositionY					C	U			D			
VelocityX					C	U			D			
VelocityY					C	U			D			
Rotation					C	U			D			
Rocket					C	U			D			
Score					C	U	R		D			
tblLander					R	U	R	R				
Name					R			R				
Propulsion					R							
Weight					R							
CentreOfGravity					R							
ScoreBonus					R		R					
Fuel					R							
DeltaFuel					R	U						
Thrust					R							
tblMap					R		R	R				
Name					R			R				
Seed					R							
Gravity					R							
AngleLimit					R							
SpeedLimit					R							

Difficulty					R		R					
tblTile					R			R				
<i>Map</i>					R			R				
x					R							
y					R							
Texture					R							
tblGlobalChat								R	D		C	R
<i>ID</i>								R	D		C	R
Player								R	D		C	R
Time								R	D		C	R
Message								R	D		C	R

MILESTONE 2

LOGICAL DESIGN ALTERCATIONS



Rationale

Some of the design consideration undertaken in milestone one did not reflect how the game actually played under the hood. The calculations required would have been too complicated for this project. Furthermore, some of the design requirements were not satisfied in the design yet outlined in the plan.

CRUD TABLE UPDATES

	Player registers new account	Player attempts to log in	Player fails to log in	Successful Login	Start new Game	Update game frame	Game ends	Admin opens admin panel	Admin Deletes player	Admin updates player info	Player sends message	Player Receives message
tblAccount	C	R	RU	RU	R		RU	R	D	RU	R	R
Username	C	R			R			R	D	RU	R	R
Password	C	R							D	RU		
isAdmin	C			R				R	D	RU		
Highscore	C			R			RU		D	RU		
LoginAttempts	C		RU	U					D	RU		
Locked	C	R	U						D	RU		
Online	C			U				R	D	RU	R	
tblGame					C	U	R	R	D			
ID					C			R	D			
Player					C			R	D			
Lander					C			R	D			
Map					C			R	D			
GameActive					C	U			D			
PositionX					C	U			D			
PositionY					C	U			D			
VelocityX					C	U			D			
VelocityY					C	U			D			
Rotation					C	U			D			
Rocket					C	U			D			
Score					C	U	R		D			
DeltaFuel					C	U						
tblLander					R	U	R	R				
Name					R			R				
Weight					R							
ScoreBonus					R		R					
Fuel					R							
Thrust					R							
SpeedLimit					R							
tblMap					R		R	R				
Name					R			R				
Gravity					R							
tblTile					R			R				
Map					R			R				
x					R							
y					R							
Texture					R							
tblGlobalChat								R	D		C	R
ID								R	D		C	R

Player								R	D		C	R
Time								R	D		C	R
Message								R	D		C	R

Rationale

The CRUD table has been updated to reflect the new schema design as reflected in the logical database design diagram.

SQL PROCEDURES & FUNCTIONS

See attached SQL File.

PLAYER REGISTRATION AND LOGIN

newUser

Input: Username, Password

Output: "OK" or "EXISTS"

Creates a new user in the account table if the username does not already exist.

userLogin

Input: username, password

Output: "OK", "BADPASS", "LOCKED" or "NOUSER"

Logs the user in and sets the user status to "Online". If the account name and password is wrong, nothing happens. If only the password is wrong, the attempts counted is incremented. 5 wrong attempts and the account is locked, needing an admin to unlock it. On successful login, attempts counted is reset to zero.

userLogout

Input: Username, Password

Output: "OK", "OFFLINE" or "NOUSER"

Changes the account status to offline granted the input information is valid.

PLAYER SELECTION AND GAME CREATION

listUsers

Output: Username, Highscore, Online, Ingame, "OK"

Returns a list of the users, whether they're online and if they're ingame.

getMap

Output: Mapname, gravity level

Lists all the maps in the game.

getLander

Output: Name, weight, thrust

Lists all the landers in the game.

newGame

Input: Lander Name, Player Name, Map Name

Outputs: "OK", Game ID

Creates a new game session, ready to be played.

collision

Input: map name, position X, position Y

Returns: BOOLEAN

Checks whether the lander is to collide with the surrounding terrain based on the position.

updateShip

Input: GameID

Output: "OK", "SUCCESS", "COLLISION"

Update the position of the ship based on external factors such as gravity and whether the engine is on or not.

Ends the game if the lander wins the game or collides with the terrain.

controlRocket

Input: Engine, AngleDelta, GameID

Output: "OK"

Updates the lander based on user input.

endgame

Input: GameID

Output: "OK"

Ends the game early if the user wants to quit. No need to crash the lander.

GAME COMMUNICATION

sendChat

Input: Username, Message

Output "OK"

Creates a new message on the global game chat with the username and adds the current timestamp to it.

getChat

Output: Username, Message, "OK"

Returns the last 20 messages sent by other users from the global game chat.

GAME SPECTATION (MULTIPLAYER)

getGameInfo

Input: Username

Output: GameID, MapName, LanderName "OK", "NOTINGAME", "OFFLINE"

Returns the game ID, map name and lander name of the specified player. Granted they are currently in a game.

getShipPosition

Input: GameID

Returns the position and rotation of the ship in the specific game ID.

ADMINISTRATION FUNCTIONALITY

verifyAdmin

Input: Username, Password

Returns: BOOLEAN

Checks whether the specified user is an administrator and returns either true or false.

listUsersAdmin

Input: Username, password

Output: everything from account table and whether they are in-game or not. "OK", "NOACCESS"

Returns all the information about the users, including information not visible to regular users.

getUserByNameAdmin

Input: Username, password, target username

Output: Everything about a specific user, "OK", "NOACCESS"

newUserAdmin

Input: Username, password, targetUsername, targetPassword, target is admin, target highscore, target login attempts, target locked.

Output: "OK", "EXISTS", "NOACCESS"

Creates a new user account but allowing the admin to specify the hidden data manually.

editUserAdmin

Input: Username, password, targetNewUsername, targetOldUsername, targetPassword, target is admin, target highscore, target login attempts, target locked.

Output: "OK", "NOACCESS"

Edits the information about a specific user and the hidden information.

deleteUserAdmin

Input: username, password, target name

Output: "OK", "NOACCESS"

Removes a user from the database.

deleteChatAdmin

Input: username, password, message ID

Output: "OK", "NOACCESS"

Removes a chat message from the global game chat.

ACID AND MULTIPLAYER SUPPORT

What is ACID?

ACID stands for Atomicity, Consistency, Isolation and Durability. These are principals that are applied to a database system to ensure data integrity and security of that system. Specifically, when transactions happen in said system.

ATOMICITY

This is where the any transactions carried out in the database must carry out successfully or not at all. This prevents data from being partially updated before an error occurring etc. In this game example, if the player position is only partially updated due to an error halfway through a procedure, it would cause the game to misbehave and be inconsistent.

CONSISTENCY

This ensures data is in a consistent state before the transaction begins and after the transaction ends. E.g., Any relations in the database must be upheld by the database rules in the transaction.

ISOLATION

This prevents the data from being by another transaction or process whilst the original transaction is working on it. So, the data used by the transaction and the transaction itself must be the only two entities in a relationship whilst the transaction is running.

DURABILITY

This is where data returned by any completed transactions is committed and saved by the database. The point of the transaction is to manipulate the data independent of the database whilst it is running (usually in volatile memory). When the transaction completes, the successfully processed data is saved to the hard disk (non-volatile).

MULTIPLAYER SUPPORT

The game supports multiplayer via game spectating and game chat functionality. Since this is the case, the database can be accessed through simple data access requests and data push requests. This way, there should not be a case where the same data is being modified by two different users simultaneously, since only one user can be playing the game, yet another can be watching or “reading” the data.

Since all the functionality for the game happen using procedures, the data is handled in a single operation thus making occurrences of data corruption much less likely to occur.

MYSQL and ACID

MySQL is built on top of the InnoDB database engine. InnoDB supports extra functionality that protects against data corruption situations. These are called COMMIT and ROLLBACK commands and can be added to existing procedures. These commands change the way the data is handled when it has been modified, instead keeping a log of changes until it is confirmed. This way the changes can be undone without any harm.

TEST DATA

See “insertSchema” procedure in SQL file.

C# IMPLEMENTATION

See “LanderConsole” project folder.

REFERENCES

Lunar Lander.png. (2017). In *Wikipedia*.

https://en.wikipedia.org/w/index.php?title=File:Lunar_Lander.png&oldid=813854906