professional practice in it

By Oskar Ciebien and David Newman

Table of Contents

[Introduction 1](#_Toc69909661)

[System Requirements 2](#_Toc69909662)

[Technology and the Reasons Why 2](#_Toc69909663)

[Implementation and Design 3](#_Toc69909664)

[*Asset design* 3](#_Toc69909665)

[Database Design and Implementation 6](#_Toc69909666)

[Architecture of the Game 7](#_Toc69909667)

[The Game 7](#_Toc69909668)

[Scenes 7](#_Toc69909669)

[Code 7](#_Toc69909670)

[Highscore table 7](#_Toc69909671)

[Database Connection 7](#_Toc69909672)

[Conclusion 8](#_Toc69909673)

[References 9](#_Toc69909674)

# Introduction

This is a joint team project of Oskar Ciebien and David Newman for Professional Practice in IT, the project will be the creation of a matching card or memory game where the user must flip over cards 2 at a time and try to match each flipped card the game will be called Super Memory Turbo Game, our objectives will be to make a playable version of this game where the user will gain score points as they match cards and at the end of the game or level the score the user achieved will be entered into the high score table with their name, have that high score link to database and then be stored inside that database, allow the score to be retrieved from the database and be displayed to the user to show them who has scored the highest.

# System Requirements

The game has been tested and run in Unity Editor. It is available to be played currently only in the Unity Editor. The user needs to have Unity installed to test the game.

# Technology and the Reasons Why

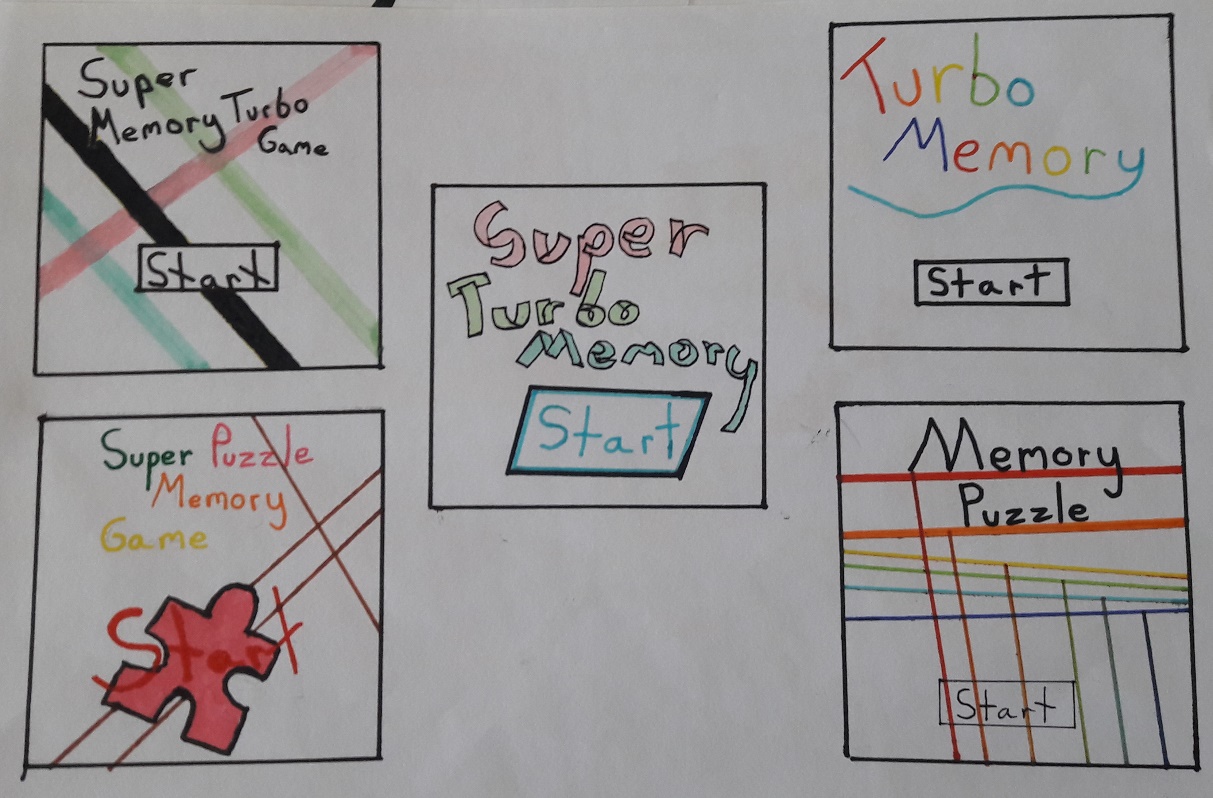
For the project we were originally planned to used Unity for the game and MySQL for the database, due to issues with the newest version of Unity and MySQL we instead looked into other databases and ended up going with SQLite, Unity is a cross platform video game engine it can be used to create 2d and 3d games and has wide use with many companies using the engine to make video games. For the graphics design in Unity we used GIMP 2.0 to create each of the assets used in the game.

# Implementation and Design

## Asset design

The first thing we did with Super Memory Turbo Game was to create the assets for the game as we could then start working on the coding inside unity and the database. First some concept artwork was done to give us an idea what we should go for with the design of the menu and the design of the cards. The concept art for both cards and menu was done on paper with ink and pens, then it was recreated in GIMP 2.0 using layers we were able to create 9 different cards in one file without having to recreate the whole design each time.

1.Start Menu Concept Art



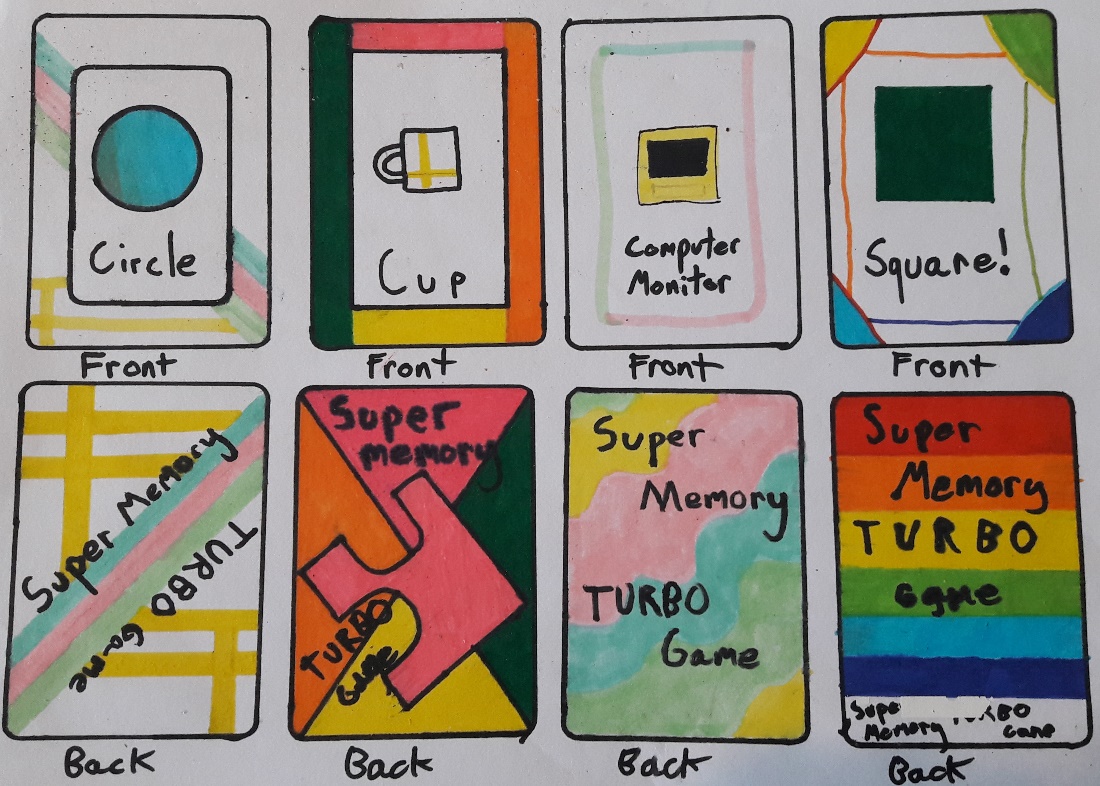
The start menu concept was based on simple colours and lines, its to grab the user interest and attention, we ended up going with this design as it matched our requirements and gave a nostalgic aesthetic 90s feel.

2.Finished Menu Screen Design



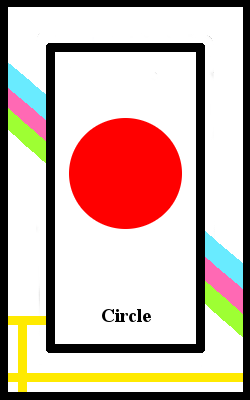
Next was the concept design for the cards that will be flipped during the game. With this we tried to make the object in the card stand out as much as we could so the user could not be confused as to what they are seen on screen, as such with went with a simple design again out of all the concept designs that we had.

3. Concept Design for Matching Cards



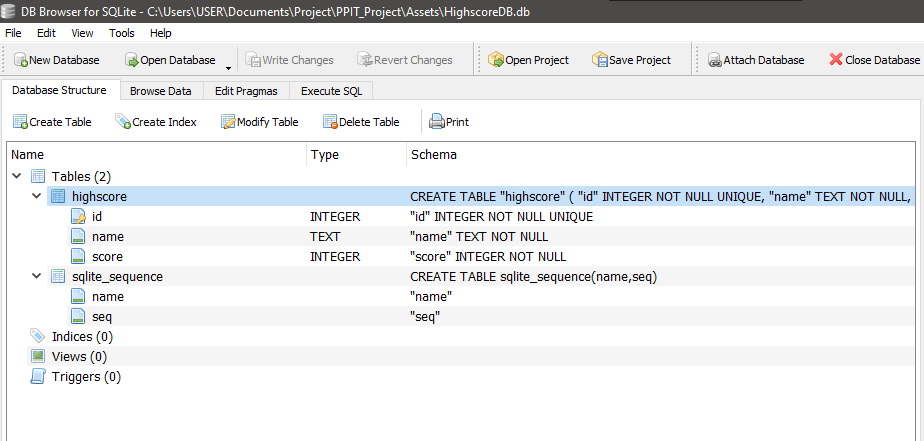
As we wanted the game to have an overall connected design and theme, we went with cards that matched the start menu in design. We use 9 cards in total in the game each with a different shape that the user must match.

4. Back of Card 5. Front of Card.

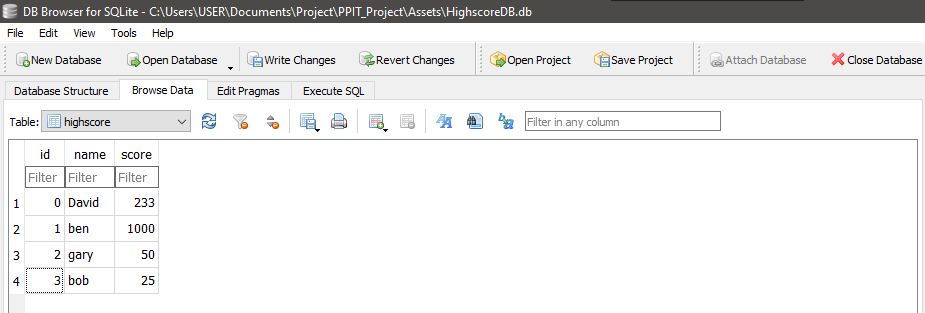


## Database Design and Implementation

Originally we were going to use MySQL but due to conflicting problems with the newest version of Unity we had to go with a different database, so for the database we used SQLite which was just what we needed for the project, DB Browser was used to create the Database.



The database is made up of a table called highscore, inside the table with have 3 columns id, name and score, id is a unique integer, name is text and can’t be null and score is an int and can’t be null. The database is designed so that in the game when the user gets a highscore there name and score will be stored in this database.



Example of user stored data in the database, the data will be retrieved from the database and showing to the user in game as a highscore table. The user will be able to enter a custom name but not enter a score.

# Architecture of the Game

## About the Game

The main idea of the game is to pair up all cards.

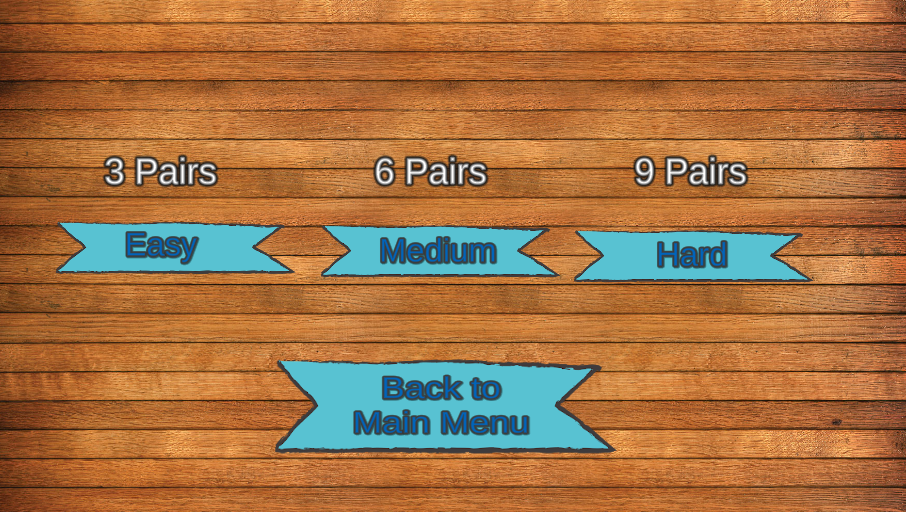
## The Menu

On the start of the game, the user is greeted with the Main Menu, where they can choose one of the three available options.

* **Play –** The user is then moved to a different scene where they can choose the level of difficulty, easy, medium, or hard.
* **Options –** The layout of the menu changes. The user can then choose to turn the sounds of the game off or on. By default, once the game starts, the sounds are on.
* Exit **–** Stops the game.

## Game Difficulty

The game offers three levels of difficulty. Each of them offers more difficulty by adding more cards that need to be paired up by the player. It is very easy to spot the difference in cards because the card’s faces are different types of shapes. Each pair consists of two cards.

****

The easy level, where the player must pair up 3 pairs of cards.

A medium level, where the player must pair up 6 pairs of cards.

The hard level, where the player must pair up 9 pairs of cards.

## Highscore table

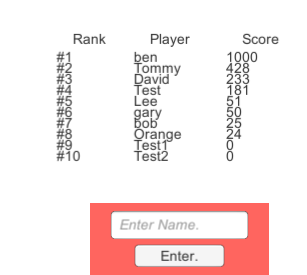
The highscore table the player views at the end of the game is created using Unity UI with scripts and coding, the user can view the highscores that have being achieved so far and add their own name to the highscore table using an input box.

## Database Connection

The database connects to the highscore table using scripts and prefab of the scores, when the user gets to the end screen, they can put in their name which is then saved with their score in the highscore database.

## In-Game

While in-game the player is presented with the cards (amount varies, depending on the chosen level). All of which are lying face down. The score starts to count as early as the player pressed on one of each levels. The player can then click on one of each of the cards to see the face of the chosen card and do the same with another. Each time the player gets a second to look at both cards. After that second, they are both flipped back, and the player can choose another two cards. Once the player pairs up all the cards the player will then be transferred to another scene.



The next scene will show a table of all scores that is connected to the database and once the player presses “ESC” button, a table will appear where the player can input their name. Once that is done, their name and score will be added to the database and the table for the player to see.

## Code

The code that we have implemented in this game has been written by us and some was also adapted and changed to work in this project. The references of all adapted code can be found at the end of this document and in the code files commented.

## Platforms

The game has only been tested in the Unity Editor but since it was developed on a Windows machine, it won’t have any problems running on a Windows machine since it has the same controls (mouse button only needed).



# Conclusion

In conclusion we…..

# References

* **Creation of Database and Highscore and linking them together.**

<https://www.youtube.com/playlist?list=PLX-uZVK_0K_7NmsYfe2BTOk_IamWC2kU3>

* **In-Game sounds functionality**

<https://www.youtube.com/watch?v=ABqmWhD5Ny8&ab_channel=Rabidgremlin>

* **Score (Time)**

<https://www.youtube.com/watch?v=M9yOOIVI2xM&ab_channel=Cezary_Sharp>

* **Game Manager code for the cards**

<https://github.com/pronaypeddiraju/Memory-Game/blob/master/Memory%20Game/Assets/Scripts/gameManager.cs>

* **CardBehaviour code for the cards**

<https://github.com/pronaypeddiraju/Memory-Game/blob/master/Memory%20Game/Assets/Scripts/cardScript.cs>

List of headings

Design Methodology Applied

Software Development Life Cycle

Limitations and Known Bugs

Testing Plans

Recommendations for Future Development