PROFESSIONAL PRACTICE IN IT

By Oskar Ciebien and David Newman

Supervisor – Martin Kenirons

Table of Contents

Introduction 2

System Requirements 2

Technology and the Reasons Why 2

Implementation and Design 3

Asset design 3

Database Design and Implementation 6

Architecture of the Game 8

About the Game 8

Game Difficulty 9

Highscore table 9

Database Connection 9

In-Game 10

Code 11

Platforms 11

Limitations and Known Bugs 11

Screencast of the Game 11

GitHub repository link 11

Software Development Life Cycle 12

Testing 13

Conclusion 14

References 15

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 two 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

Minimum requirements
Windows. OS: Any modern
OS. Processor: 1.1 GHz Processor.
Memory: 2 GB RAM

Technology and the Reasons Why

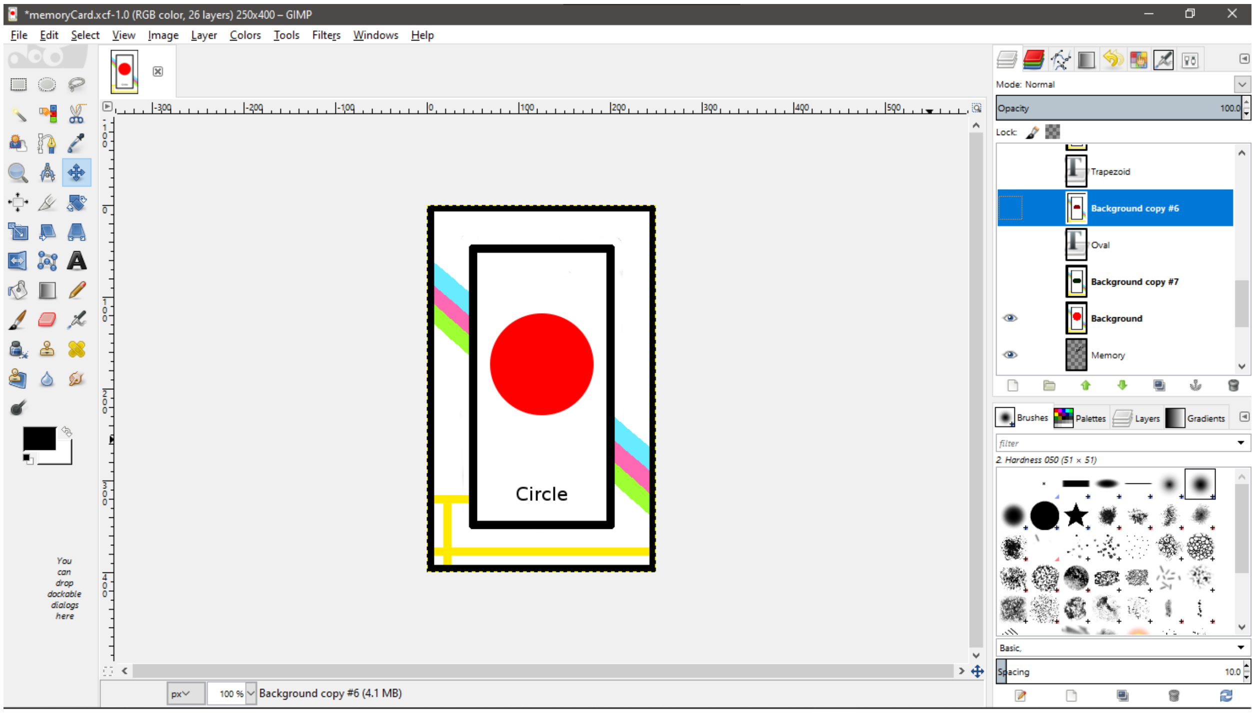
For the project we were originally planned to used Unity for the game and MySQL for the database, due to issues with the newest version of Unity and MySQL we instead investigated other databases and ended up going with SQLite, Unity is a cross platform video game engine it can be used to create 2D and 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

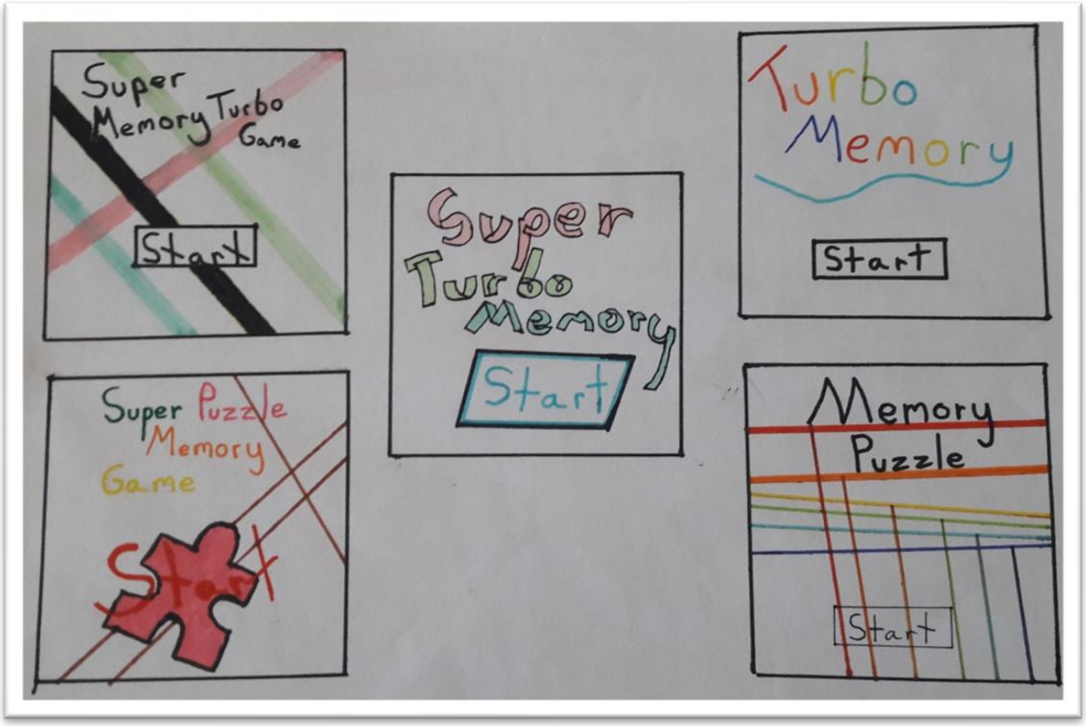
Asset design

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 first a base layer for the background was created, then once that background was done, the image was reversed and the centre was removed, then an area for the symbol that will be matched in game was made, once the base for the front of the card was finished, multiple layers were made and each shape was created with a layer for each text, this way we can have every card finished just be hidden or showing layers.

1.Unity Interface

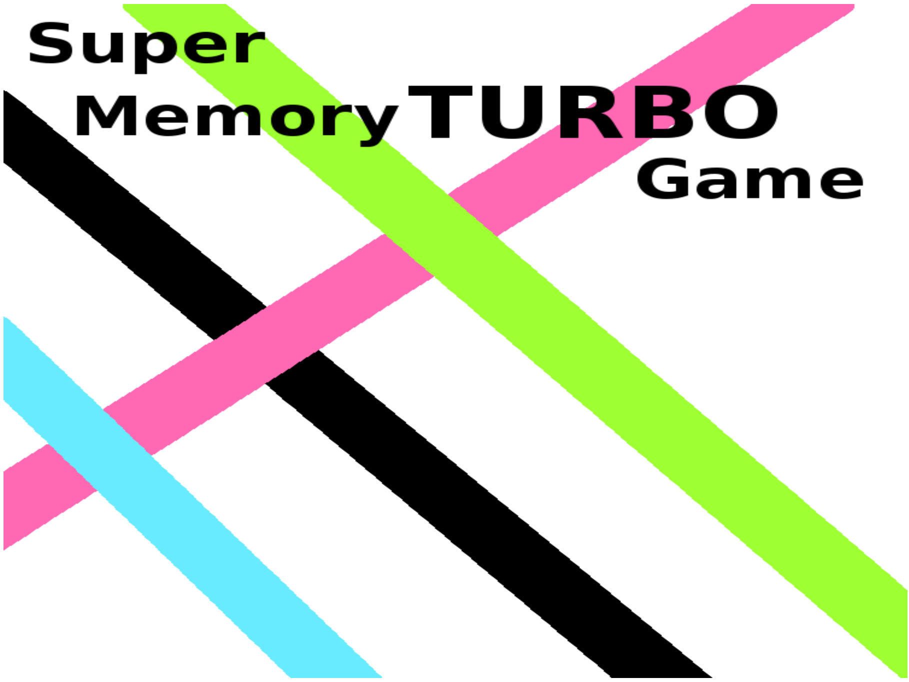

*memoryCard.xcf-1.0 (RGB color, 26 layers) 250x400 – GIMP
File Edit Select View Image Layer Colors Tools Filters Windows Help
Circle
You can drop dockable dialogs here
Mode: Normal
Opacity 100.0
Lock:
Trapezoid
Background copy #6
Oval
Background copy #7
Background
Memory
Brushes Palettes Layers Gradients
filter
2. Hardness 050 (51 × 51)
Basic,
Spacing 10.0
px
100 %
Background copy #6 (4.1 MB)

1. Start Menu Concept Art



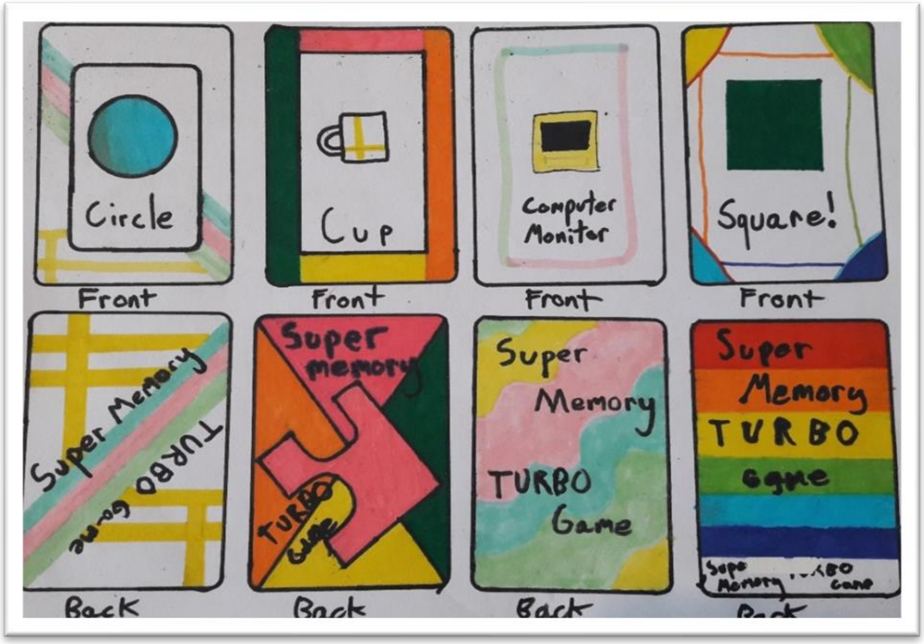
The start menu concept was based on simple colours and lines, it's to grab the user's interest and attention, we ended up going with this design as it matched our requirements and gave a nostalgic aesthetic 90s feel.

2.Finished Menu Screen Design



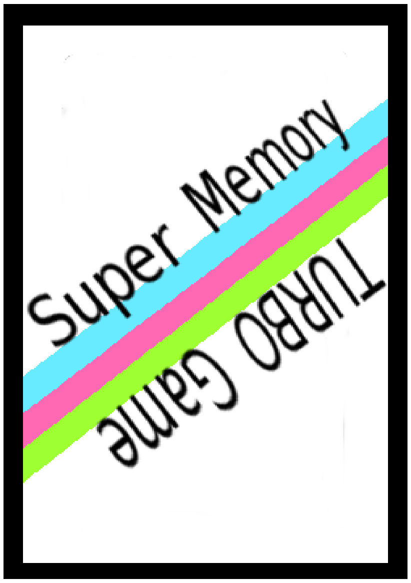
3. Concept Design for Matching Cards

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.

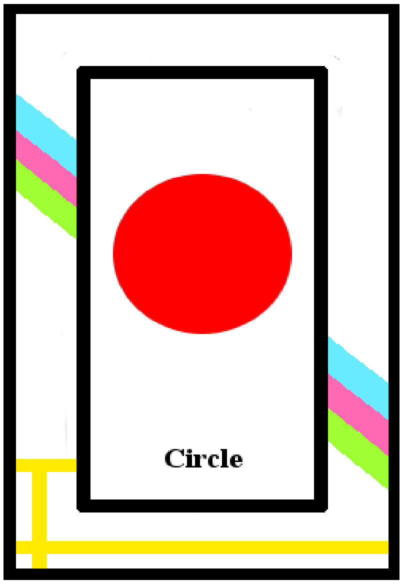


As we wanted the game to have an overall connected design and theme, we went with cards that matched the start menu in design. We use 9 cards in total in the game each with a different shape that the user must match.

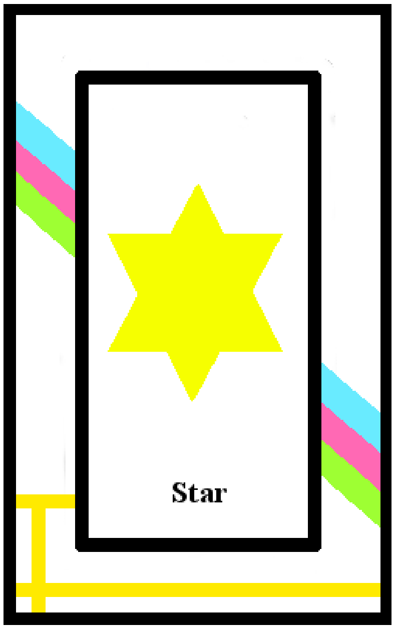
4. Back of Card

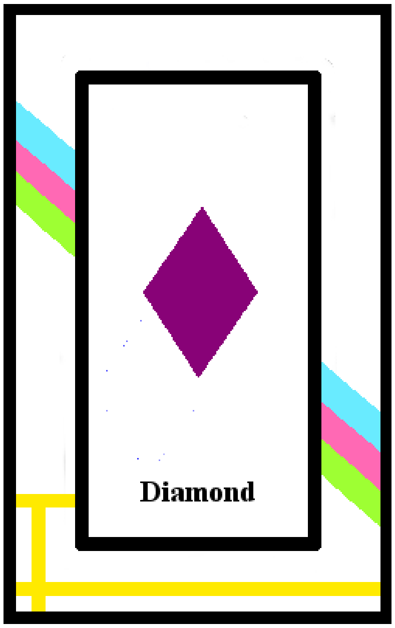


5. Front of Card.



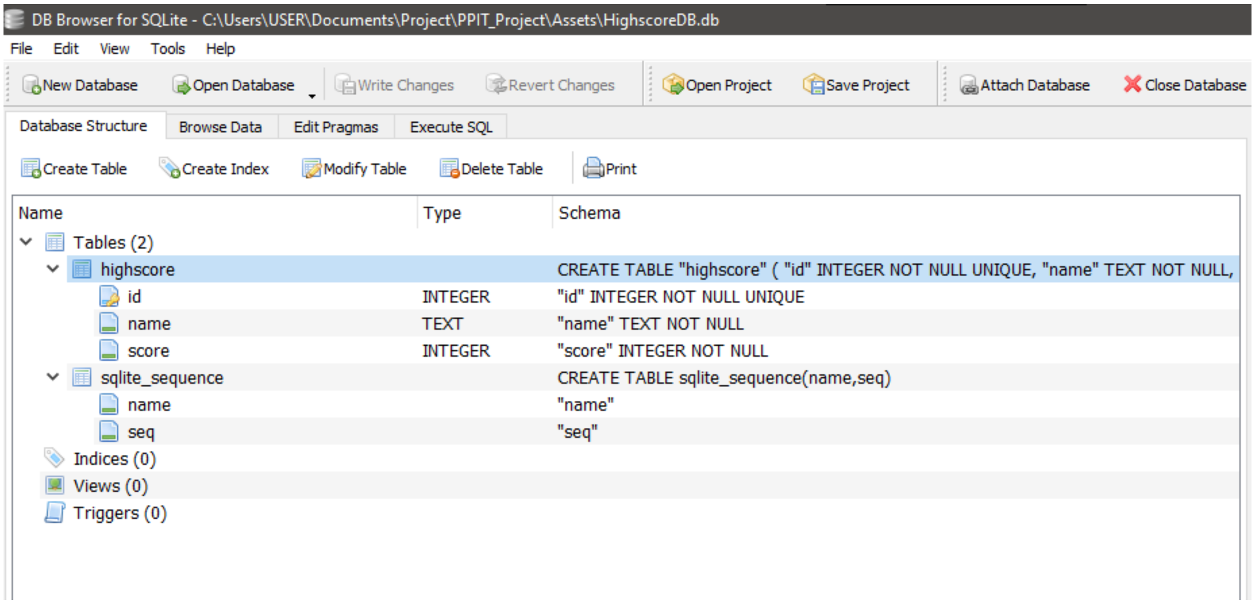
Examples of other cards





Database Design and Implementation

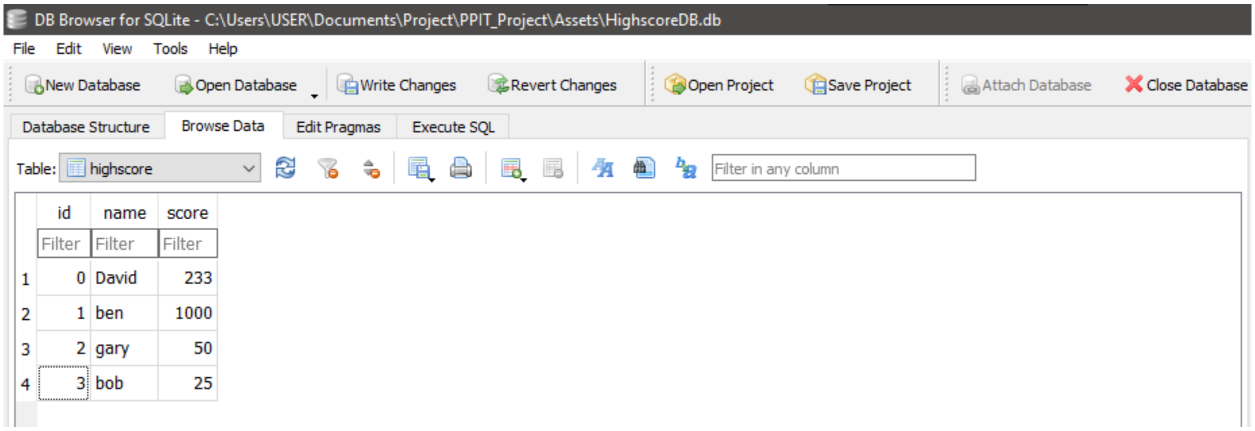
Originally we were going to use MySQL but due to conflicting problems with the newest version of Unity we had to go with a different database, so for the database we used SQLite which was just what we needed for the project, DB Browser was used to create the Database.



Name	Type	Schema
Tables (2)		
highscore		CREATE TABLE "highscore" ("id" INTEGER NOT NULL UNIQUE, "name" TEXT NOT NULL,
id	INTEGER	"id" INTEGER NOT NULL UNIQUE
name	TEXT	"name" TEXT NOT NULL
score	INTEGER	"score" INTEGER NOT NULL
sqlite_sequence		CREATE TABLE sqlite_sequence(name,seq)
name		"name"
seq		"seq"
Indices (0)		
Views (0)		
Triggers (0)		

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 their 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. DB Browser was used and is a very easy to use SQLite database with a database, the database that was made using it was inserted into the assets folder of Unity allowing easy access to the database as shown below.



	id	name	score
	Filter	Filter	Filter
1	0	David	233
2	1	ben	1000
3	2	gary	50
4	3	bob	25

Architecture of the Game

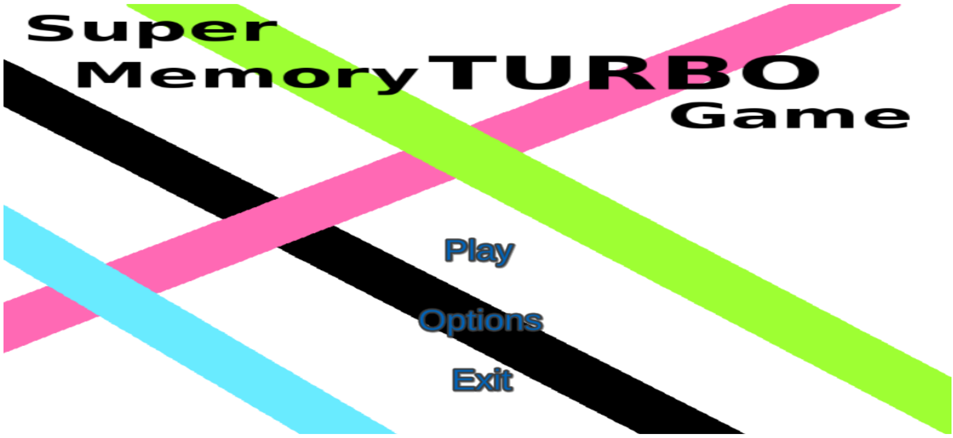
About the Game

The main idea of the game is to pair up all cards the user clicks on a card that card is then flipped over to show the face of the card and the symbol on the face of that card is what the user must match to another card, they then select another card which flips over and if both cards match they stay flipped and the users gets points for the highscore, they then continue to match cards.

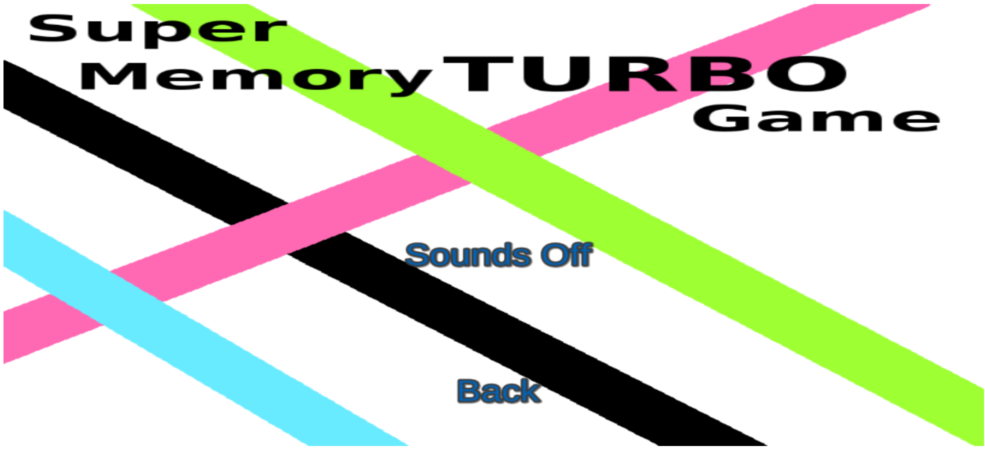
The Menu

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.



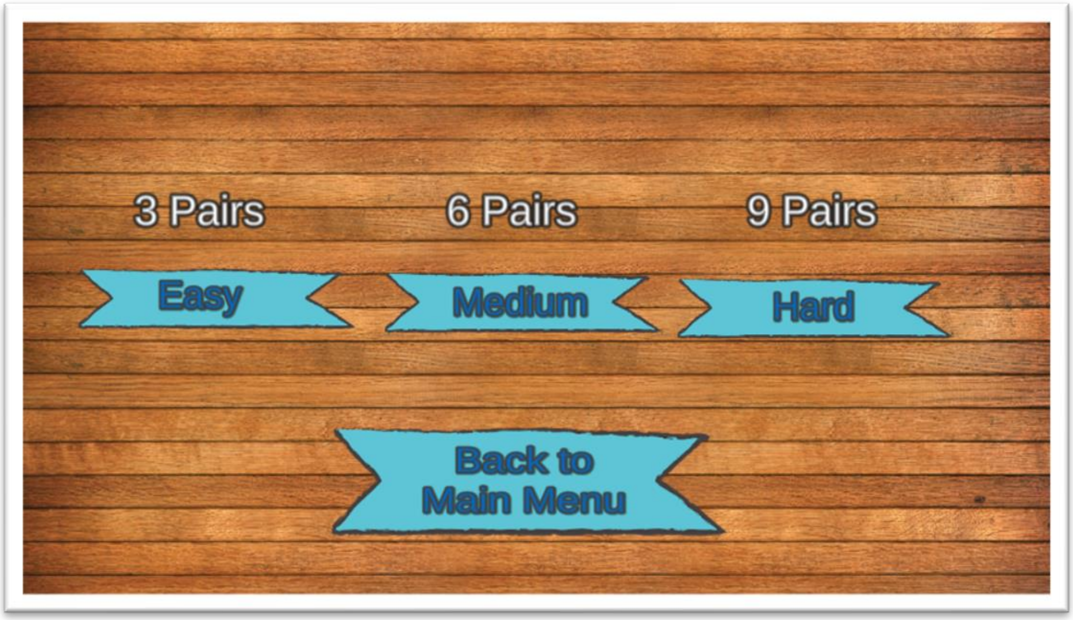
- **Options –** The layout of the menu changes. The user can then choose to turn the sounds of the game off or on. By default, once the game starts, the sounds are on.



- **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

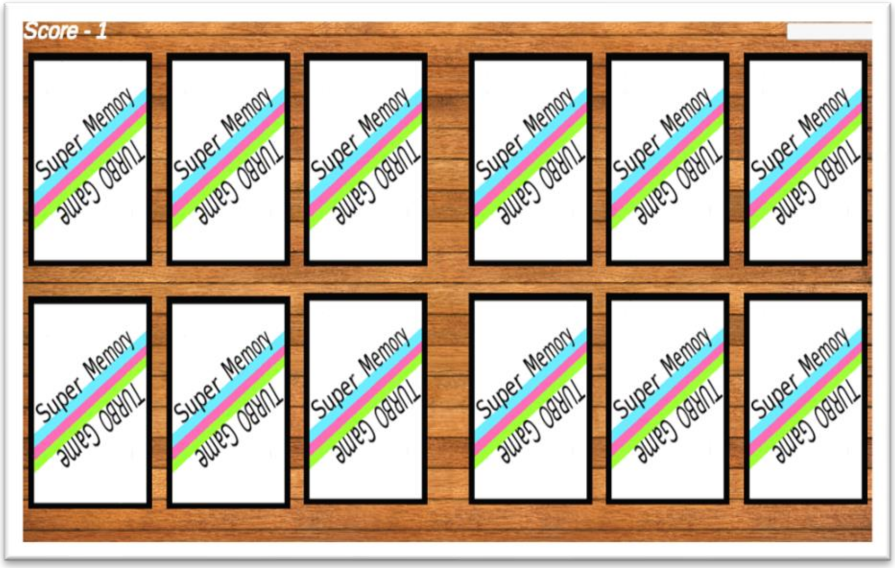
The highscore table the player views at the end of the game is created using Unity UI with scripts and coding, the user can view the highscores that have being achieved so far and add their own name to the highscore table using an input box.

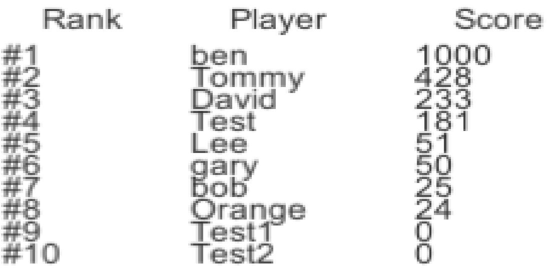
Database Connection

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.

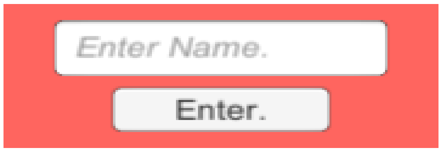
In-Game

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 level. 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.





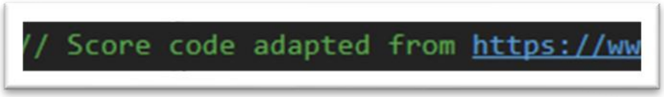
Rank	Player	Score
#1	ben	1000
#2	Tommy	428
#3	David	233
#4	Test	181
#5	Lee	51
#6	gary	50
#7	bob	25
#8	Orange	24
#9	Test1	0
#10	Test2	0



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.



```
// Score code adapted from https://ww
```

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).

Limitations and Known Bugs

The game right now has only been tested to run on windows machines, only has select levels for each difficulty mode.

Screencast of the Game

https://web.microsoftstream.com/video/6a2d6cfa-484e-475f-9e47-5fe69b15a5ab

GitHub repository link

https://github.com/Oskar-Ciebien/PPIT_Project

Software Development Life Cycle

We first started with analysing what the project would be, the ideas we had for the project at the start was a driving theory test game, movie database, learning website and an educational game. After going through what we could and couldn't do in the time giving for the project we decided to go with an educational game at first but going through what we would need to do we decided to do a puzzle matching game. Then we went on to the design of the game and what it would look like for the user, we draw up designs for the cards and the menus of the game and thought about how each one would look in game and how we could recreate them for the game, what the user would like and enjoy, before finally decided on the ones we did. Next we started on the codding of the game Oskar started with the base of how the game would work in Unity and I went and got the database working and connecting it to Unity, once the connection was done and the game was finished we tested the game by playing it a lot to see where it works and didn't work, at this stage the game would be deployed to a platform like Steam or the Google Play Store, after that with feedback from the user we would then go on to either make changes or updates to the game and fix bugs.

Testing

We didn't find many bugs during this testing, during the initial testing of the database connection we found out that the version of Unity we were working with didn't work very well with the version of MySQL we were using as such we had to switch and use a different database.

We also found a bug with the database that we could not get to figure out. The score points were not transferred from one script or one scene to the highscores scene. Therefore, each time the player enters their name, their score will be 0. We have implemented a line of code that enters a random score for the player.

While creating the main menu. We had a problem with the settings menu. The buttons from the settings menu were appearing in runtime when they should only appear when the user would press the settings menu button. In the end we figured out that it was only a minor mistake in the unity editor.

When we tested all the levels one by one, we made sure that the user could not select more cards at a time.

Conclusion

In conclusion we were able to create and finish the project within time of the deadline, we both gained a great knowledge in the software we used to help create the game, thanks to this project we have a wide scope of software to use when it comes to creating a database and linking them and creating video games, it can be hard working together as a team we both feel like we were able to help each other any time we had a problem or not stuck with the code or design of the game. We think the game is a great tool for any age to experience and enjoy even though it is somewhat basic games that are basic can sometimes become hugely successful just by the enjoyment of the users that play them, giving more time for the project we would have added more levels and in the future if we were to continue the project things like an endless more where the user keeps matching cards to build up a high score or a puzzle mode where the user has to match cards in different patterns to beat the level, we both look forward to working on group projects in the future and we believe this has been a great learning experience for both of us and as students has allowed us to grow.

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