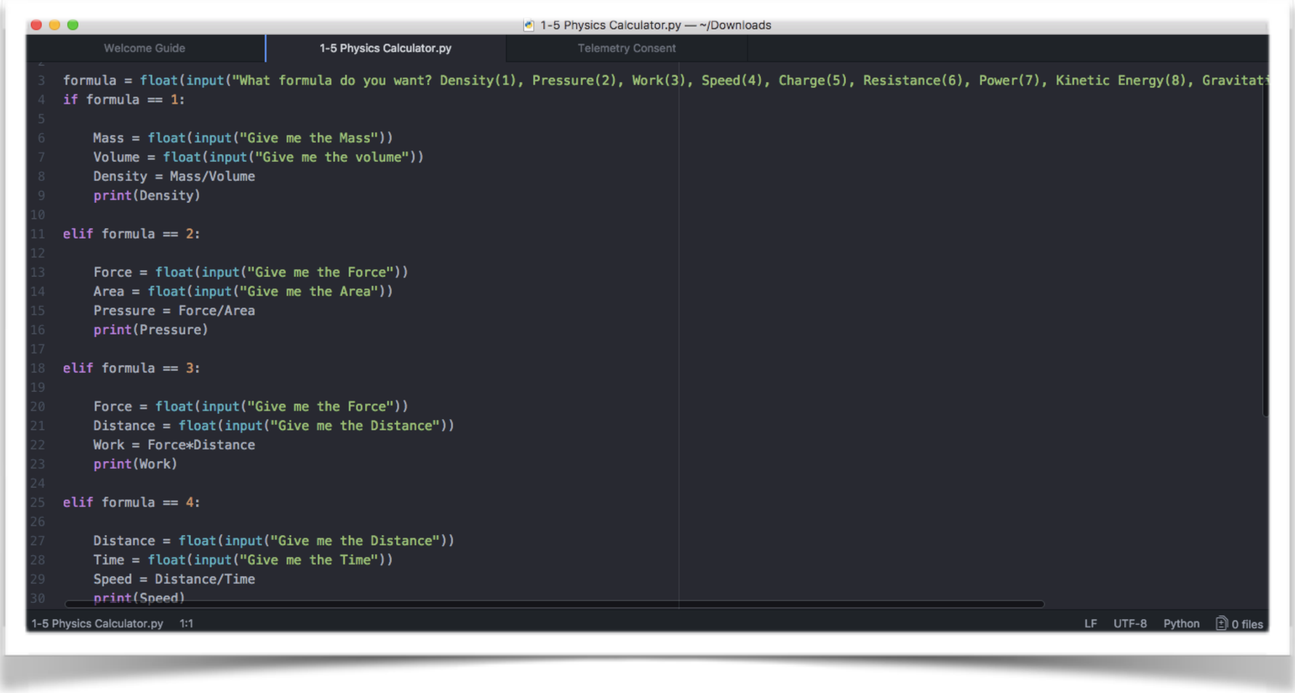
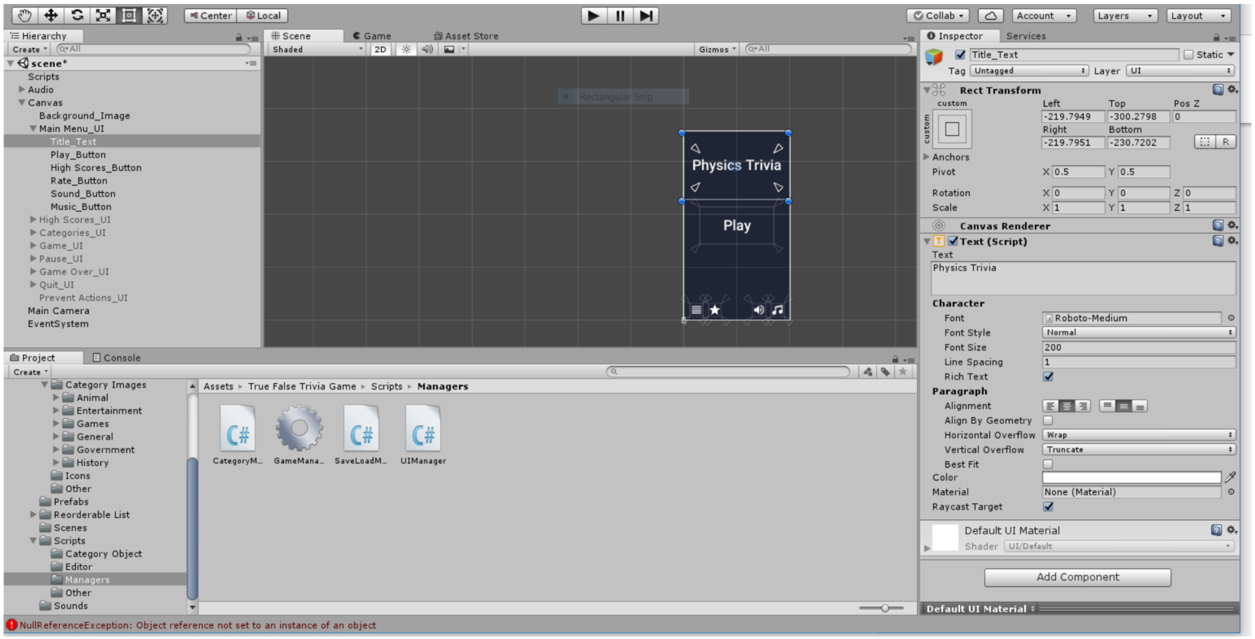
At the start of my EP Project, I started on coding in Python which is an interpreted high-level programming language for general-purpose programming. Over my years in Computer Science, I've continued to do multiple projects in Python where it completed simple tasks required. When coming to my Physics Formula App. I first tried making a code to solve simple formulas that are used regularly in IGCSE via the Syllabus.


1-5 Physics Calculator.py — ~/Downloads
Welcome Guide
1-5 Physics Calculator.py
Telemetry Consent
1-5 Physics Calculator.py 1:1
LF UTF-8 Python 0 files

```python
formula = float(input("What formula do you want? Density(1), Pressure(2), Work(3), Speed(4), Charge(5), Resistance(6), Power(7), Kinetic Energy(8), Gravitati
if formula == 1:

    Mass = float(input("Give me the Mass"))
    Volume = float(input("Give me the volume"))
    Density = Mass/Volume
    print(Density)

elif formula == 2:

    Force = float(input("Give me the Force"))
    Area = float(input("Give me the Area"))
    Pressure = Force/Area
    print(Pressure)

elif formula == 3:

    Force = float(input("Give me the Force"))
    Distance = float(input("Give me the Distance"))
    Work = Force*Distance
    print(Work)

elif formula == 4:

    Distance = float(input("Give me the Distance"))
    Time = float(input("Give me the Time"))
    Speed = Distance/Time
    print(Speed)
```

Upon doing a bit coding on Python, I quickly realized how hard it may be to continue with Python due to lack of features in the design side of the code. This meant that it'll be extremely hard for me to code a decent looking app for the mobile. This is when I decided to move onto Unity and C++ Coding.

Unity's offerings are well known, from its animation tech mechanism to its new suite of 2D tools. These 2D tools have been exceptionally helpful to me and one of the main reasons I had picked to work on Unity in the first place. This is based on the fact that it is extremely useful when it comes to app development based on mobile devices. Rather than just a simple engine, Unity now offers an increasing selection of services making it a near one-stop shop for all of my games development needs



Hierarchy

scene*
- Scripts
- Audio
- Canvas
  - Background_Image
  - Main Menu_UI
    - Title_Text
    - Play_Button
    - High Scores_Button
    - Rate_Button
    - Sound_Button
    - Music_Button
  - High Scores_UI
  - Categories_UI
  - Game_UI
  - Pause_UI
  - Game Over_UI
  - Quit_UI
  - Prevent Actions_UI
- Main Camera
- EventSystem

Project

- Category Images
  - Animal
  - Entertainment
  - Games
  - General
  - Government
  - History
  - Icons
  - Other
- Prefabs
- Reorderable List
- Scenes
- Scripts
  - Category Object
  - Editor
  - Managers
  - Other
- Sounds

Assets ▸ True False Trivia Game ▸ Scripts ▸ **Managers**

CategoryM… GameMana… SaveLoadM… UIManager

Inspector: Title_Text — Tag Untagged — Layer UI

Rect Transform (custom)

	Left -219.7949	Top -300.2798	Pos Z 0
	Right -219.7951	Bottom -230.7202	
Pivot	X 0.5	Y 0.5	
Rotation	X 0	Y 0	Z 0
Scale	X 1	Y 1	Z 1

Canvas Renderer

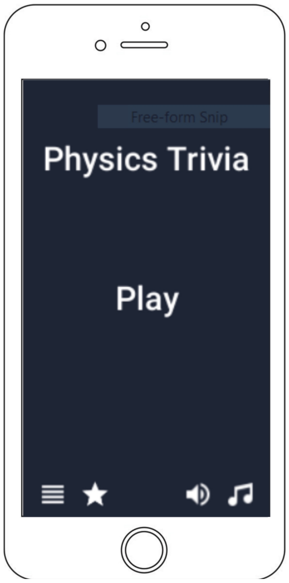
Text (Script)

Text: Physics Trivia

Character	
Font	Roboto-Medium
Font Style	Normal
Font Size	200
Line Spacing	1
Rich Text	☑
Paragraph	
Align By Geometry	☐
Horizontal Overflow	Wrap
Vertical Overflow	Truncate
Best Fit	☐
Material	None (Material)
Raycast Target	☑

Default UI Material — Shader UI/Default

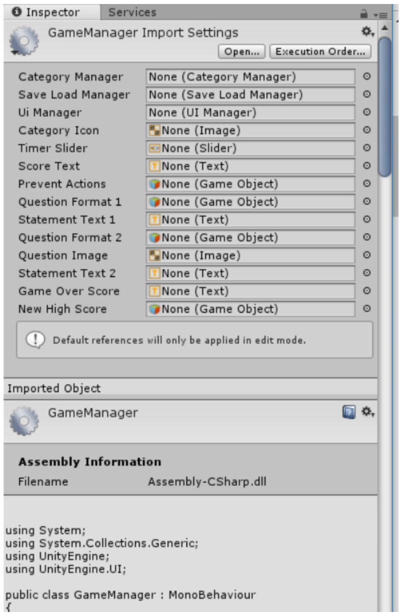
NullReferenceException: Object reference not set to an instance of an object



Game View - The physics game trivia is built for portrait resolution. So required for me to create a resolution that suits mobile use. This includes IOS and Android Development.

9 : 16 Border System to Fit the Mobile Device in Portrait Mode

Assets Used - During my app development in Unity, One thing that was exceptionally helpful was the Asset store. This helped me get assets on my app that I may not have been able to code in C++. The assets used really helped me develop a simple template for the trivia game. This template included a Game Manager code that helped organize all my variables together in a single C++ code. Variables are mentioned below ⤵



GameManager Import Settings

Category Manager	None (Category Manager)
Save Load Manager	None (Save Load Manager)
Ui Manager	None (UI Manager)
Category Icon	None (Image)
Timer Slider	None (Slider)
Score Text	None (Text)
Prevent Actions	None (Game Object)
Question Format 1	None (Game Object)
Statement Text 1	None (Text)
Question Format 2	None (Game Object)
Question Image	None (Image)
Statement Text 2	None (Text)
Game Over Score	None (Text)
New High Score	None (Game Object)

Default references will only be applied in edit mode.

Imported Object

GameManager

Assembly Information

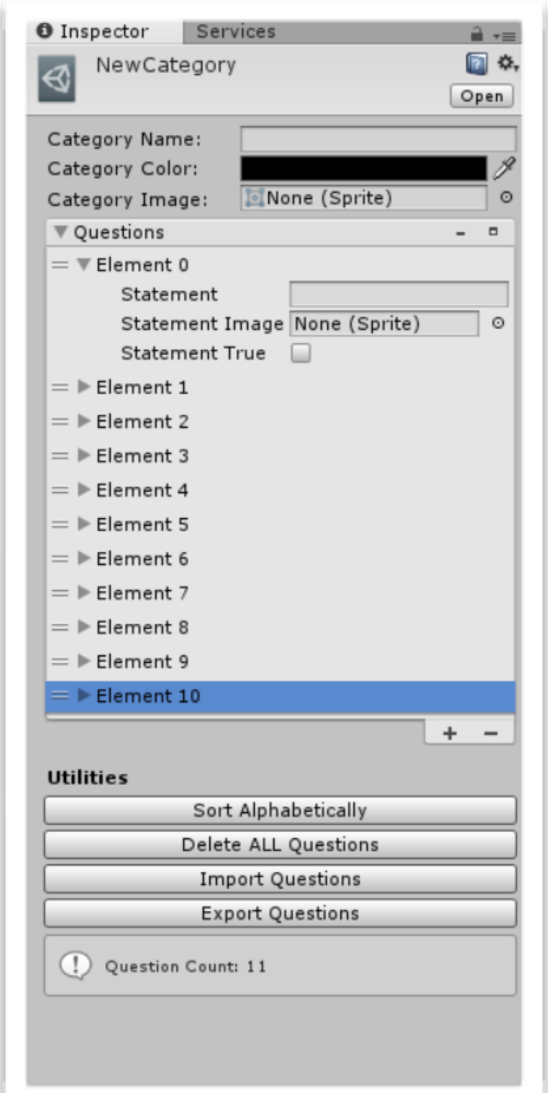
Filename Assembly-CSharp.dll

```
using System;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class GameManager : MonoBehaviour
{
```

- **Category Object:** Handles Category Object.
- **Make Category Object:** Creates a Category Object.
- **Category Manager Editor:** Custom inspector for Category Manager.
- **Category Object Editor:** Custom inspector for Category Object.
- **Category Manager:** Handles Categories.
- **Game Manager:** Handles core gameplay.
- **Save Load Manager:** Handles saving and loading the game.
- **UI Manager:** Handles most UI

Formula Based Questions - In this area, I had to make sure I used my resources of the Cambridge IGCSE Student Exercise Book & Cambridge IGCSE® Physics 0625 Syllabus which is solely based for the students learning IGCSE Physics to show everything that they could possibly be tested on in the examinations. This meant that I was easily able to use these resources to come up with questions that would help students learn them in an interactive way and be prepared for them to come up in the exams.

When it came to inserting these questions in my app, the following variables were created to make it easier when coding on C++. ⤵



Inspector | Services

NewCategory

Open

Category Name:

Category Color:

Category Image: None (Sprite)

Questions

Element 0

Statement

Statement Image None (Sprite)

Statement True □

Element 1

Element 2

Element 3

Element 4

Element 5

Element 6

Element 7

Element 8

Element 9

Element 10

+ −

Utilities

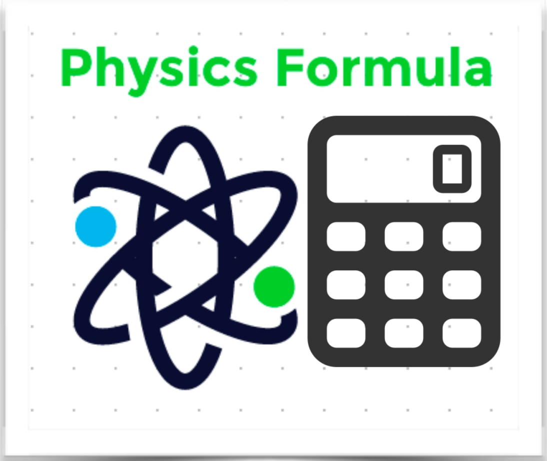
Sort Alphabetically

Delete ALL Questions

Import Questions

Export Questions

Question Count: 11



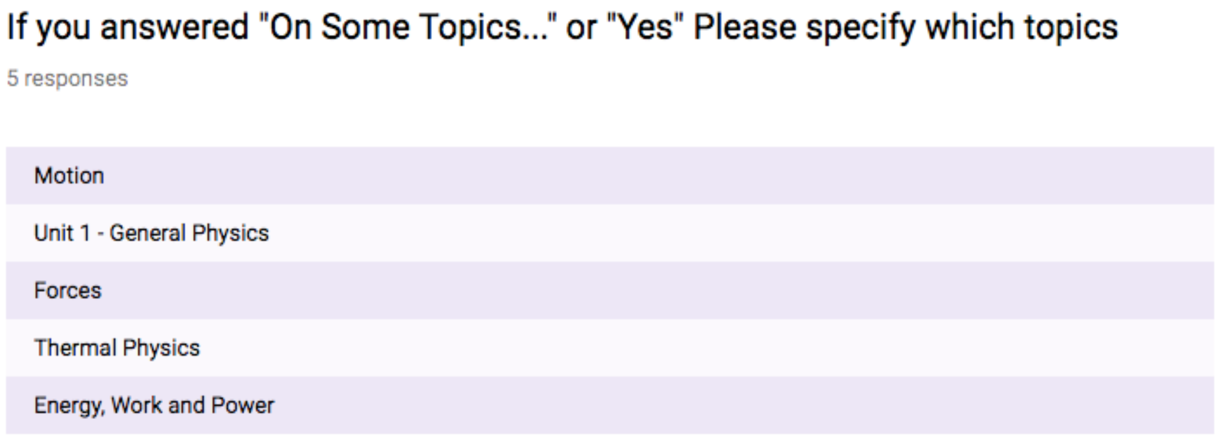
- **Category Name:** Name of category. (For Example IGCSE Physics Formula Quiz)
- **Category Color:** Color of category.
- **Category Image:** Image of category. (For Example My own Logo design for App) ⤵
- **Questions:** Questions for the category.

(Explained in Formula Based Questions ⤴)

- **Utilities:** ⤵

- **Sort Alphabetically**: Sorts questions alphabetically.
- **Delete ALL Questions**: Deletes all questions for this category.
- **Import Questions**: Imports question from a XML file, see example xml under categories folder. (However, does not support images.)
- **Export Questions**: Exports question to a XML file. (However, does not support images.)

<u>Example Questions -</u> When deciding on the questions to use as part of my trivia. As important as it was to make sure I cover the essential topics and formulas, I also felt that I should check with some current IGCSE Students and check if there was a specific trend in the struggles people had with specific topics and the formulas that came with it.



If you answered "On Some Topics..." or "Yes" Please specify which topics

5 responses

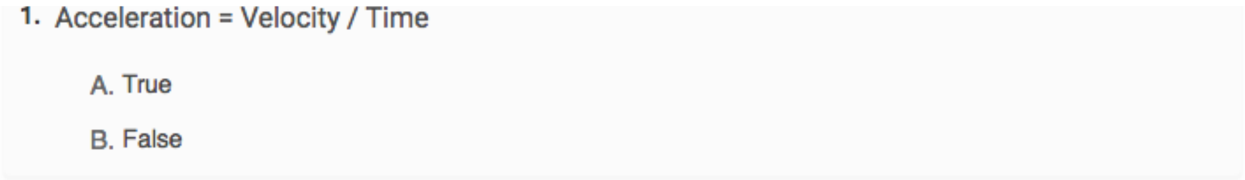
Motion

Unit 1 - General Physics

Forces

Thermal Physics

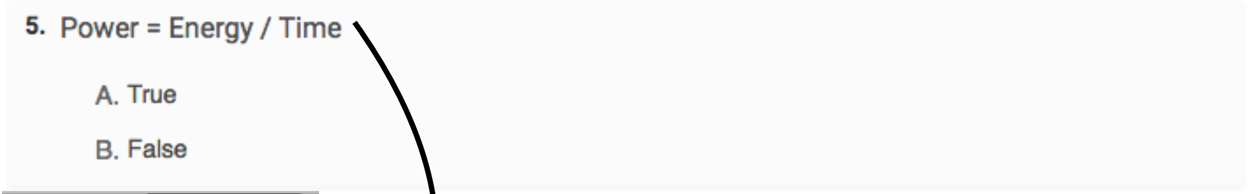
Energy, Work and Power



1. Acceleration = Velocity / Time

   A. True

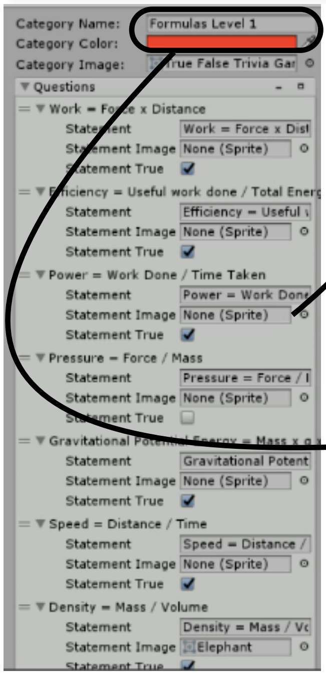
   B. False



5. Power = Energy / Time

   A. True

   B. False



Category Name: Formulas Level 1

Category Color:

Category Image: True False Trivia Gar

▼ Questions

▼ Work = Force x Distance

Statement: Work = Force x Dist

Statement Image: None (Sprite)

Statement True ☑

▼ Efficiency = Useful work done / Total Energ

Statement: Efficiency = Useful w

Statement Image: None (Sprite)

Statement True ☑

▼ Power = Work Done / Time Taken

Statement: Power = Work Done

Statement Image: None (Sprite)

Statement True ☑

▼ Pressure = Force / Mass

Statement: Pressure = Force / I

Statement Image: None (Sprite)

Statement True ☐

▼ Gravitational Potential Energy = Mass x g x

Statement: Gravitational Potent

Statement Image: None (Sprite)

Statement True ☑

▼ Speed = Distance / Time

Statement: Speed = Distance /

Statement Image: None (Sprite)

Statement True ☑

▼ Density = Mass / Volume

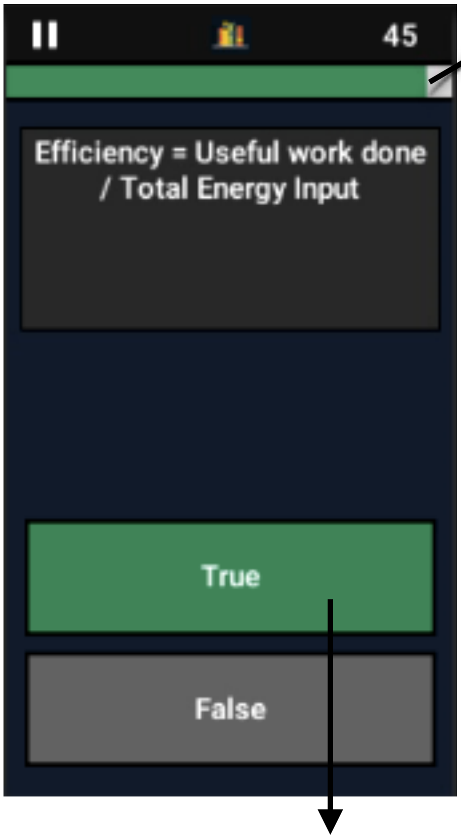
Statement: Density = Mass / Vc

Statement Image: Elephant

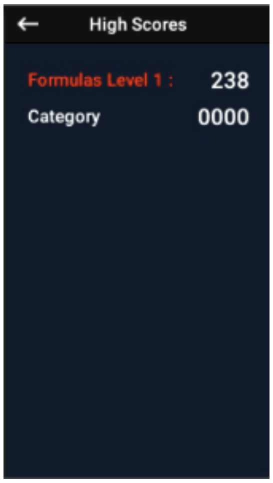
Statement True ☑

As important as it was to use the primary research of gathering questions / topics that were frequently struggled and checking if there was a general trend. It was still extremely crucial for me to cover every single formula and information provided in the syllabus.

Looking through the syllabus, I quickly realized the range in difficulties the formulas were in and adjusted the trivia categories into levels going from Level 1 to Level 3. (Level 1 is shown on the image in the right.



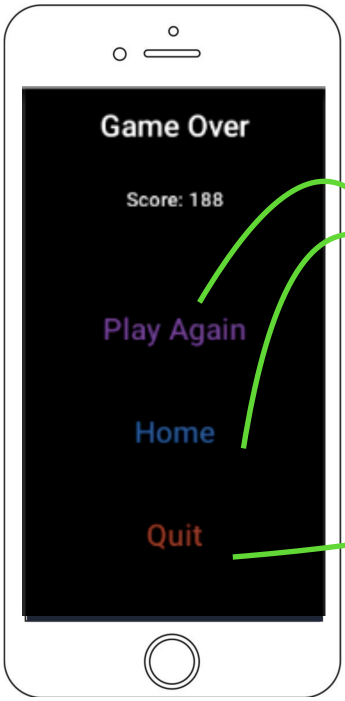
Efficiency = Useful work done / Total Energy Input



Formulas Level 1 :	238
Category	0000

This is the System I had added to make the game for interactive and creative for the users based on how students learn best. This system is a score based system that gives a maximum of 50 points per correct answered question. The quicker the answer given, the more points given. Within a few questions the scores will be added up. It is important of students to have a competitive mindset so that they have a reason to improve. To track their improvement, I have also created a system using C++ on a List of High Scores as Shown on the left

This is another feature I had added to help identify whether or not the user has got the question right by a simple green light on the click if correct, If not a red light on the click if wrong.


Game Over

Score: 188

Prior to the system made at the top, It was also important to create a menu system to help in the basic interaction of the app and how people can find their way around it. This will allow the users / students to play again, return to main menu and progress onto another game mode or access settings / high scores. Or even quit the app and return to their mobile device home screen.