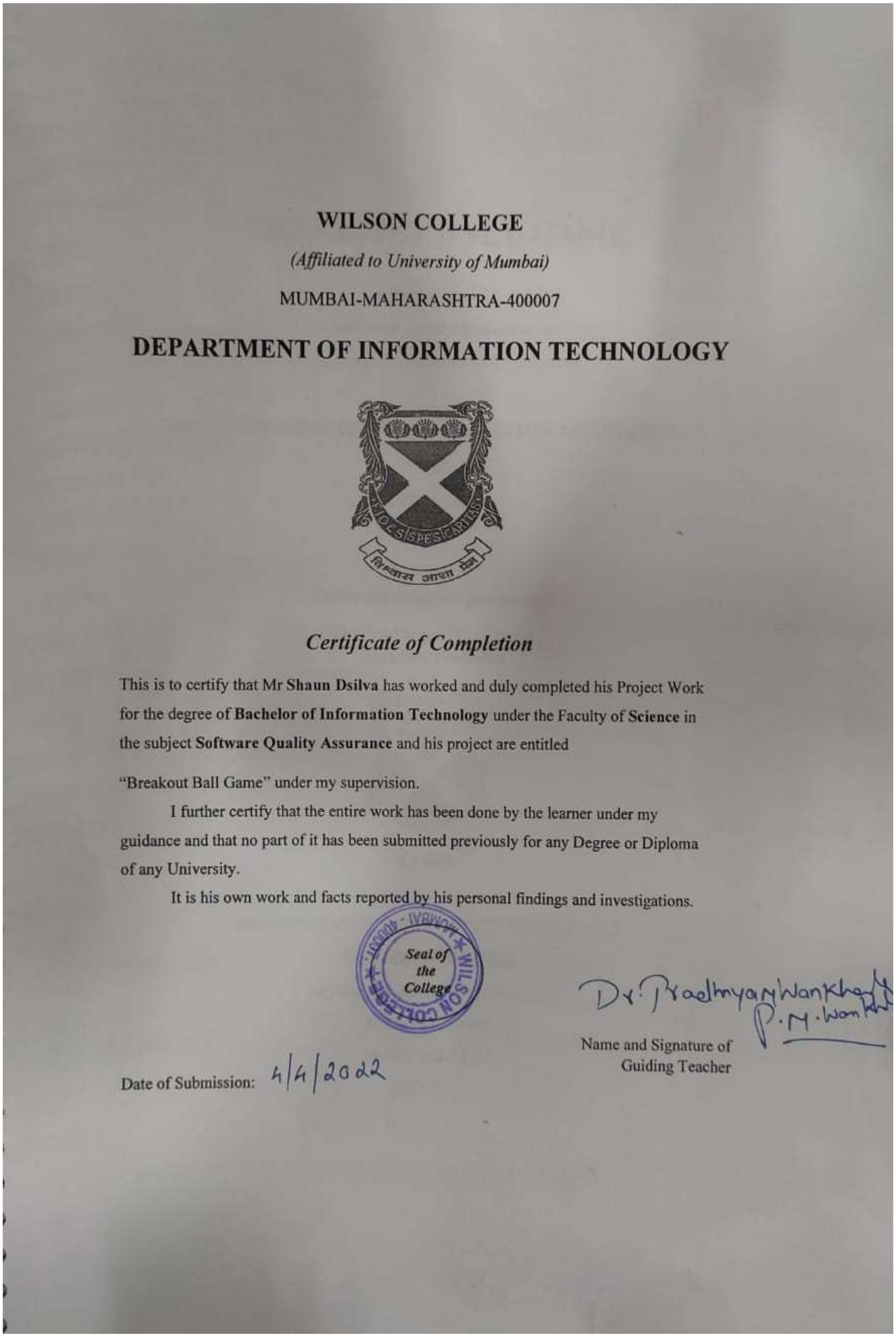


WILSON COLLEGE

(Affiliated to University of Mumbai)

MUMBAI-MAHARASHTRA-400007

DEPARTMENT OF INFORMATION TECHNOLOGY

Certificate of Completion

This is to certify that Mr **Shaun Dsilva** has worked and duly completed his Project Work for the degree of **Bachelor of Information Technology** under the Faculty of **Science** in the subject **Software Quality Assurance** and his project are entitled

"Breakout Ball Game" under my supervision.

I further certify that the entire work has been done by the learner under my guidance and that no part of it has been submitted previously for any Degree or Diploma of any University.

It is his own work and facts reported by his personal findings and investigations.

WILSON COLLEGE ★ MUMBAI - 400007 ★ Seal of the College

Dr. Pradhnya N Wankhade
P. N. Wankhade

Name and Signature of
Guiding Teacher

Date of Submission: 4/4/2022

BREAKOUT BALL GAME

A Project Report

Submitted in partial fulfilment of the

Requirement for the award of the Degree of

BACHELOR OF SCIENCE (INFORMATION TECHNOLOGY)

By

Shaun Dsilva
1920ITTY – 13

Under the esteemed guidance of

Dr Pradhnya Wankhade
Assistant Professor

DEPARTMENT OF INFORMATION TECHNOLOGY

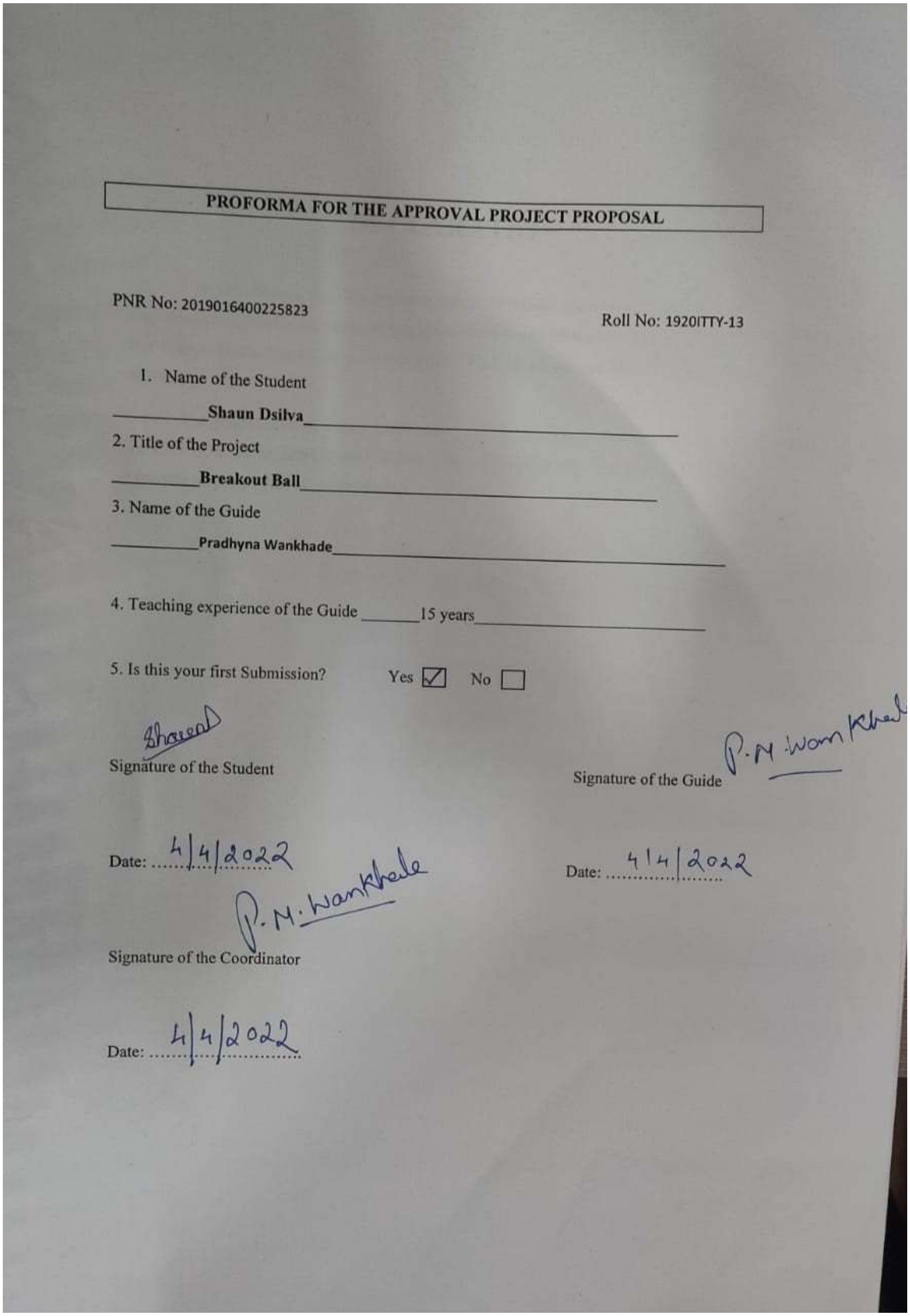
WILSON COLLEGE

(Affiliated to University of Mumbai)

MUMBAI, 400007

MAHARASHTRA

2021-2022



PROFORMA FOR THE APPROVAL PROJECT PROPOSAL

PNR No: 2019016400225823

Roll No: 1920ITTY-13

1. Name of the Student

__________Shaun Dsilva__________

2. Title of the Project

__________Breakout Ball__________

3. Name of the Guide

__________Pradhyna Wankhade__________

4. Teaching experience of the Guide ______15 years__________

5. Is this your first Submission? Yes ☑ No □

Shaun

Signature of the Student

Signature of the Guide P.M. Wankhad

Date: 4/4/2022

Date: 4/4/2022

P.M. Wankhede

Signature of the Coordinator

Date: 4/4/2022

DECLARATION

I hereby declare that the project entitled, "Breakout Ball Game" done at Wilson College, has not been in any case duplicated to submit to any other university for the award of any degree. To the best of my knowledge other than me, no one has submitted to any other university.

The project done in partial fulfillment of the requirement for the award of degree pf **BACHELOR OF SCIENCE (INFORMATION TECHNOLOGY)** to be submitted as final semester project as part of our curriculum.

Shaun Dsilva

ShaunD

ABSTRACT

This project Breakout Ball is a Game Development Project. It is developed on the platform named “Unity”.

Breakout Ball is a classic game. In Breakout Ball, a layer of bricks lines the top third of the screen and the goal is to destroy them all. The ball moves straight around the screen, bouncing off the top and two sides of the screen. When the brick gets hit, the ball bounces back and the brick is destroyed. The player loses a turn when the ball touches the bottom of the screen. The goal of this game is to break the bricks without missing the ball with your platform.

ACKNOWLEDGEMENT

I take this opportunity to express my profound gratitude and deep regards to my teacher for their exemplary guidance, monitoring and constant encouragement throughout the course of this project.

The blessings, help and guidance given by them, from time to time, shall carry me long way in the journey of life in which I am about to embark.

I also take this opportunity to express a deep sense of gratitude to Dr. Pradhnya Wankhade for her coordinal support, valuable information and guidance, which helped me in completing this task through various stages.

A large debt of gratitude is owned to my project guide Prof. Srilatha Ratnam who has not only endure, but also encouraged, assisted and inspired me for taking up the project on "Breakout Ball Game".

I want to acknowledge and thank her for giving me the opportunity to do this under her guidance and also sharing her immense knowledge. Her continuous guidance, time, valuable suggestions, inputs and helpful criticisms have helped me to accomplish such a challenging task.

Lastly, I thank almighty. My parents, family and friends for their constant encouragement, with which I could carry on this project through thick and thin.

TABLE OF CONTENT

Chapter 1: Introduction Page 1

1.1 Background

1.2 Objective

1.3 Purpose, Scope and Applicability

1.3.1 Purpose

1.3.2 Scope

1.3.3 Applicability

Chapter 2: Survey of technologies Page 4

Chapter 3: Requirements and Analysis Page 8

3.1 Problem Definition

3.2 Requirement and Specification

3.3 Planning and Scheduling

3.4 Hardware and Software Requirements

3.4.1 Hardware Requirements

3.4.1 Software Requirements

3.5 Conceptual Models

Game Flow Chart

3.6 Use Case Diagram

Chapter 4: System Design Page 14

4.1 Basic Modules

4.2 Procedural Design

4.3 User Interface

4.4 Test Cases Design

Chapter 5: Implementation and Testing Page 21

5.1 Implementation Approach

5.2 Coding Details

5.3 Testing Approach

5.3.1 Unit Testing

5.3.2 Integrated Testing

Chapter 6: Results and Discussion Page 35

6.1 User Documentation

Chapter 7: Conclusion and Future Scope Page 41

Chapter 1

Introduction

1.1 Background:

Game Development is one of the fun ways to learn technology. In this report, we will be discussing all the functional requirements, use case diagram, data flow diagram and basic implementation steps for the development of breakout ball game.

In this game, we have layers of colored bricks and ball with which to break the layers. The player moves the paddle from left to right to keep the ball from falling. A life is used when the player fails to hit the ball.

The paddle doesn't bounce the ball like a mirror, although it does so when the ball hits right in the middle. The closer the bounce take place to the left end of the paddle; a more significant left turn is added to an expected mirror bouncing.

The bricks get broken after coming in contact with the ball that bounces around the screen. At the bottom is a paddle that moves based on user input. The user has to make sure the ball bounces off the paddle without going of the bottom of screen. A camera will track the position of the paddle as the user moves the paddle.

1.2 Objectives:

The objective of the game is to eliminate all of the BRICKS that are distributed around the top of the game screen.

The bricks get broken after coming in contact with a ball that bounces around the screen.

At the bottom is a paddle that in game moves based on user input. The player moves a PADDLE from side-to-side to hit a BALL. The user has to make sure the ball bounces off the paddle without going off the bottom of the screen.

But, if the ball hits the bottom enclosure, the player loses and the game ends! To win the game, the player has to eliminate all the Bricks.

1.3 Purpose, Scope and Applicability:

1.3.1 Purpose:

- The purpose of this report is to highlight the key features and requirements of this project.
- To provide a vision of features and facilities that are provided to the game player.
- To provide more efficient, interactive graphical interface and multilevel game.

1.3.2 Scope:

- Since the gaming industry is continually expanding and numerous games are played on a regular basis by devoted audience, the scope of this game in future will continue to grow significantly.
- Our project deals with 2 types of users.
  - Game player
  - Developers
- There will be different modules for different users according to their needs.

1.3.3 Applicability:

- This game can be run on all commonly used platforms.
- The features of this game are:
  - Easy to access
  - User friendly interface
  - Easy to handle.

Chapter 2

Survey of Technologies

Microsoft visual studio:

Microsoft Visual is an integrated development environment (IDE). It is used to develop computer programs, websites, web apps. Visual studio uses platform such as Windows API, Windows Form, Window Store.

Visual Studio supports different languages and allows nearly any programming language, built-in languages include: -

- C
- C++
- Visual Basic
- .NET
- C#

Visual Studio supports running multiple instances on the environment (each with its own set of VS Packages). The professional edition includes a superset of the VS Packages in the standard edition.

Microsoft Visual Studios uses MSSCCI Version 1.3, which adds support for rename and delete propagation, as well as asynchronous opening.

MSSCCI (Microsoft Source Code Control Interface) provides a set of functions which are used to implement various source control functionality, with a standard Visual Studio user interface.

Features of Visual Studios are: -

- Enhanced support for multi-targeting.
- Support for parallel programming and debugging.
- Support for call hierarchy methods.
- Support for XSLT profiling and debugging.

The Visual Studio IDE provides support for designing and implementing applications targeted at .NET framework.

The Visual Studio IDE supports: -

- Call hierarchy of methods.
- Zoom in/out source code.
- Navigate to option.
- Generate to usage.

➢ C++

- C++ is a general-purpose programming language created by Bjarne Stroustrup as an extension of the C programming language, or "C with Classes".
- The language has expanded significantly over time, and modern C++ has object-oriented, generic, and functional features in addition to facilities for low-level memory manipulation.
- It is almost always implemented as a compiled language, and many vendors provide C++ compilers, so it is available on many platforms.

- Unity (game engine):

Unity is a cross-platform game engine developed by Unity Technologies, first announced and released in June 2005 at Apple Inc’s Worldwide Developers Conference as a Mac OS X-exclusive game engine.

The engine has since been gradually extended to support a variety of desktop, mobile, console and virtual reality platforms. It is particularly popular for IOS and Android mobile game development.

It is considered easy to use for beginner developers and is popular for indie game development.

The engine can be used to create three-dimensional (3D) and two-dimensional (2D) games, as well as interactive simulations and other experiences. The engine has been adopted by industries outside video gaming, such as film, architecture, engineering, construction, and the United States Armed Forces.

Unity was initially released for Mac OS X, later adding support for Microsoft Windows and Web browsers.

The Unity engine allows developers to make both 2D and 3D games. It currently supports only the C sharp programming language. It supports Direct3D, OpenGL, OpenGL ES, Metal, Vulkan and proprietary API. Since 2016, Unity offers services on the cloud.

In 2016 Unity changed from one time purchase into a subscription model. There currently exist one free and three paid licensing options: Personal (Free), Plus, Pro and Enterprise.

Unity has a robust collection of official tutorials to help familiarize new developers with the engine. Additionally, it has thorough documentation that they update to be accurate with each new Unity release.

The tutorials and documentation make use of pre-built assets that are available in the default Unity program. Developers can use these instead of being forced to write all of their code from scratch.

Within 2D games, Unity allows importation of sprites and an advanced 2D world renderer.

For 3D games, Unity allows specification of texture compression, mipmaps, and resolution settings for each platform that the game engine supports, and provides support for mapping, reflection mapping, parallax mapping, screen space ambient occlusion(SSAO), dynamic shadows using shadow maps, render-to-texture and full-screen post-processing effects.

Two separate render pipelines are available, High Definition Render Pipeline (HDRP) and Universal Render Pipeline (URP), in addition to the legacy built-in pipeline.

All three pipelines are incompatible with each other. Unity offers a tool to upgrade shaders using the legacy renderer to URP or HDRP.

Chapter 3

Requirements and Analysis

3.1 Problem Definition:

The problem of current versions of this game are less efficient, less interactive graphical interface and have limited levels.

It affects the Game player, Game Development and Gaming industry.

The impact of problem is that, Game players are moving towards other attractive arcade games which are efficient and have interactive graphics, so for that games they have to pay more.

The successful solution proposed is to build game with high definition graphics which will be very efficient and have multilevel in it as well as number of broken bricks will be counted according to their color.

3.2 Requirements Specification:

- To create a arcade game which can be played by everyone.
- To display username in the middle of the screen.
- Scores must be stored in Database.
- To display scores and lives.
- The game should be light-weight and that supports all mobile Platforms so it could be installed on every device.

3.3 Planning and Scheduling:

The software Development Life-Cycle model being used for the development of this Agile model.

In software development, Agile model practices include requirements discovery and solutions improvements through the collaborative effort of self-organizing and cross-functional teams with their customers/end users, adaptive planning, evolutionary development, early delivery, continual improvement, and flexible responses to changes in requirements, capacity and understanding of the problems to be solved.

Agile SDLC model is a combination of iterative and incremental process models with the processes of adaptability and customers satisfaction by rapid delivery of working software products.

Agile model breaks the products into small incremental build. These builds are provided by iterations. Each iteration usually last for one to three weeks. Every iteration involves cross functional teams working continuously on various areas like

- Planning
- Reporting analysis
- Design
- Coding
- Unit testing
- Acceptance testing

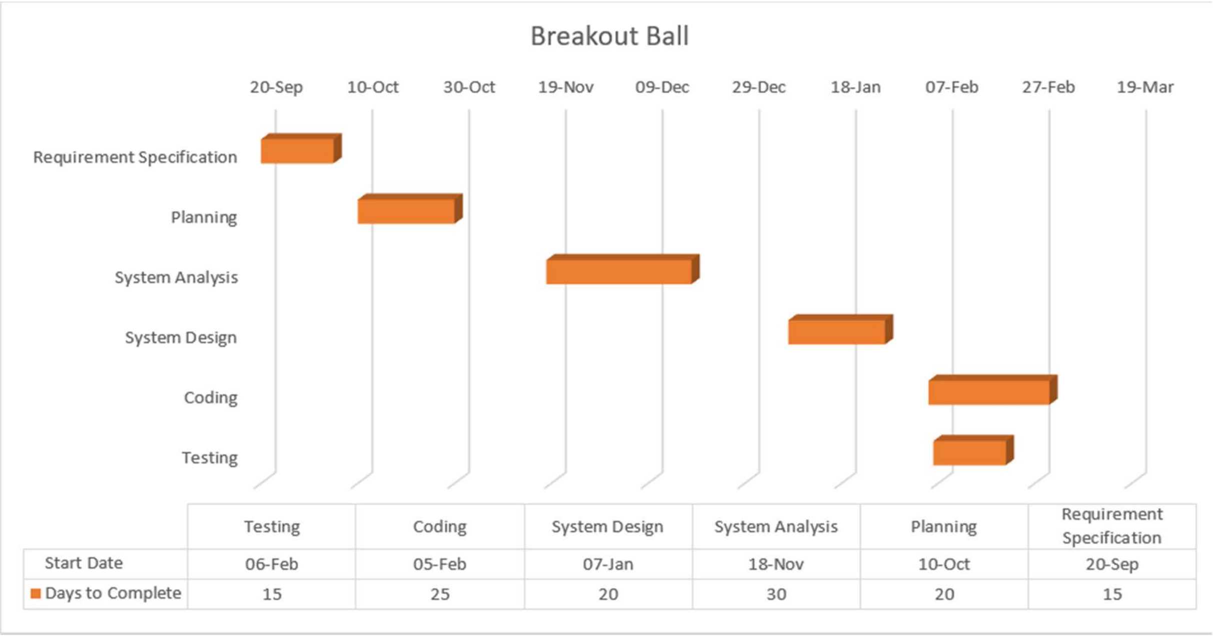
At the end iterations, a working product is displayed to the customer and the important stakeholders.

Agile model is being widely accepted in the software world recently. The main advantages of an Agile SDLC are

- Functionally can be developed rapidly and demonstrated.
- Suitable for fixed or changing requirement.
- Delivers early partial working solution.

Gantt Chart:

A Gantt Chart is a type of bar chart that illustrates a project schedule. This chart lists the tasks to be performed on the vertical axis, and the time interval on the horizontal axis. The width of the horizontal bars shows the duration of each activity.



	Testing	Coding	System Design	System Analysis	Planning	Requirement Specification
Start Date	06-Feb	05-Feb	07-Jan	18-Nov	10-Oct	20-Sep
■ Days to Complete	15	25	20	30	20	15

3.4 Hardware and Software Requirements

3.4.1 Hardware Requirement

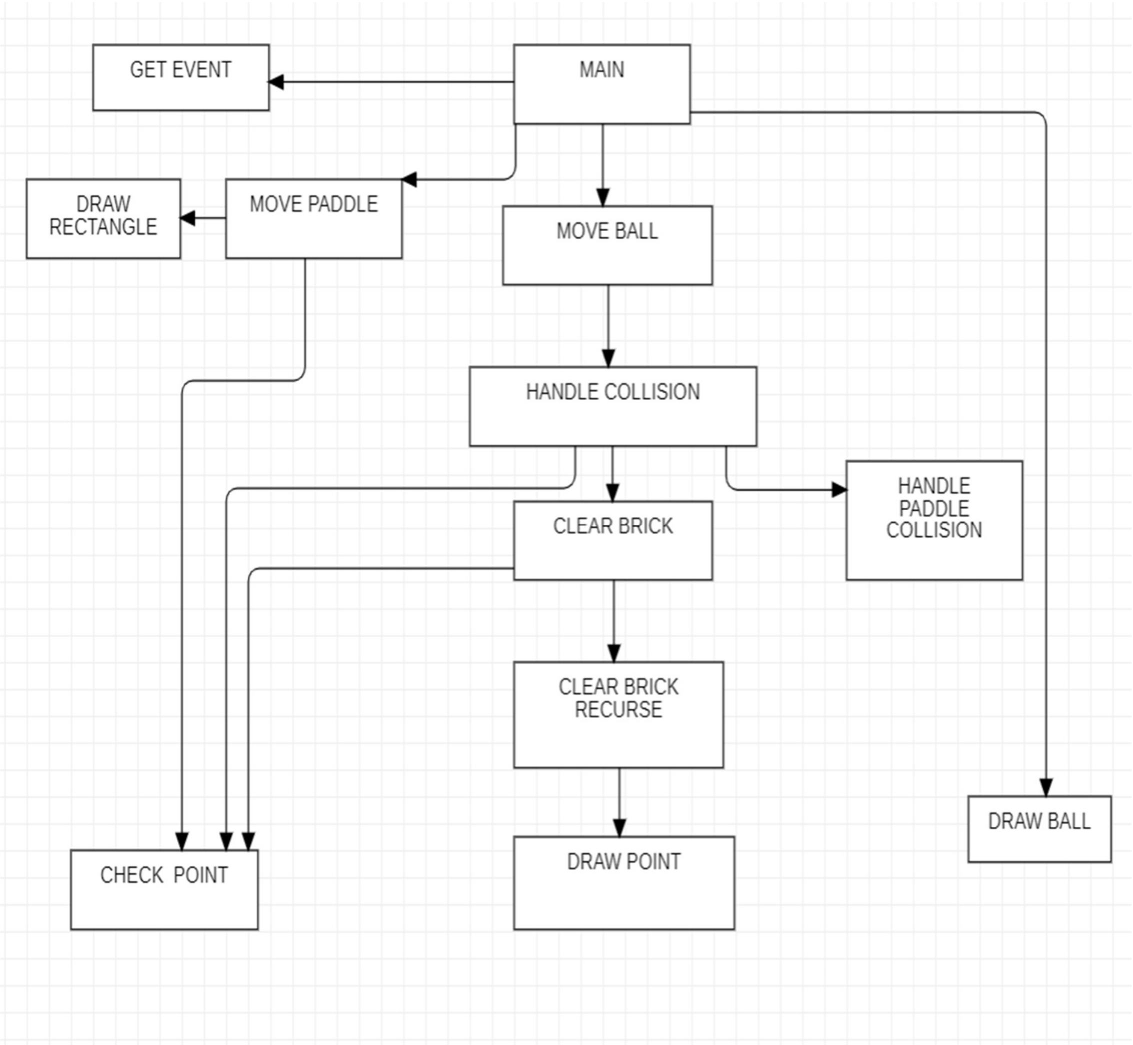
1. OS: Windows 7
2. Processor: Intel Pentium Dual-Core
3. Memory: 4 GB RAM
4. Graphics: AMD Radeon
5. Storage: 480 MB available space.

3.4.2 Software Requirement

1. You will need at least 480 MB of free disk space to install the game.
2. The cheapest graphics card you can play it in is an AMD Radeon HD 4650. Furthermore, an AMD Radeon R7 250 is recommended in order to run the game with the highest settings.
3. The minimum memory requirement is 4 GB of RAM.
4. To play this game you will need a minimum CPU equivalent to an Intel Core 2 Duo Q6867.

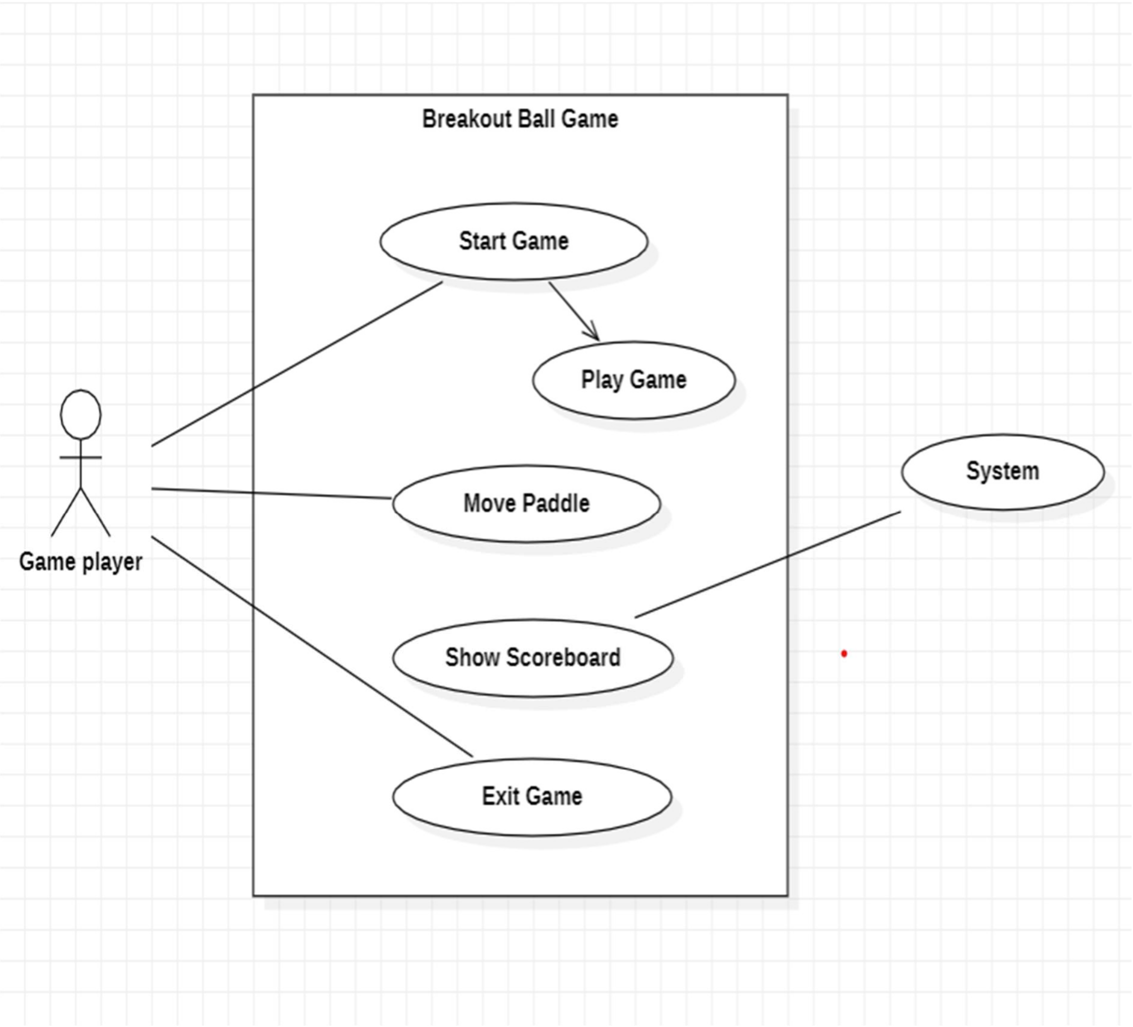
3.5 Conceptual Model

Game Flow Chart



3.6 Use Case Diagram

A use case diagram is a dynamic or behaviour diagram in UML. Use case diagrams model the functionality of a system using actors and use cases. Use cases are a set of actions, services, and functions that the system needs to perform. In this context, a "system" is something being developed or operated, such as web site. This "actors" are people or entities operating under defined roles within the system.



Chapter 4

Game Design

Game design is the study of how to make a game work by establishing game rules, game mechanics and gameplay. To make the game more entertaining rather than visually accepted, which is a part of game design because even if we produce a fancy game with good graphical work, if the game design isn't good enough, it won't interest the user.

4.1 Basic Modules:

- Software Implementation

A. Paddle control

The paddle will be controlled by a mouse. The mouse's moving to the left or right will correspond to the paddle's movement. The moving speed will also relate to how fast we move the mouse.

B. Bouncing ball

Software will be used to assign new locations of the ball as the ball bounce around walls (edges of the screen), bricks and paddle. When the angle of incidence changes, the angle of reflection changes too.

C. Bricks

For each brick, the software will have a status tracker. The bricks may have different radii. The actual sizes of bricks and gaps are left for your decision. There should be at least 15 bricks when the game starts. When a brick is hit by the ball, the brick is eliminated.

D. Score counting

Software will count and keep a track of the scores. The score will be displayed on the upright corner of the screen.

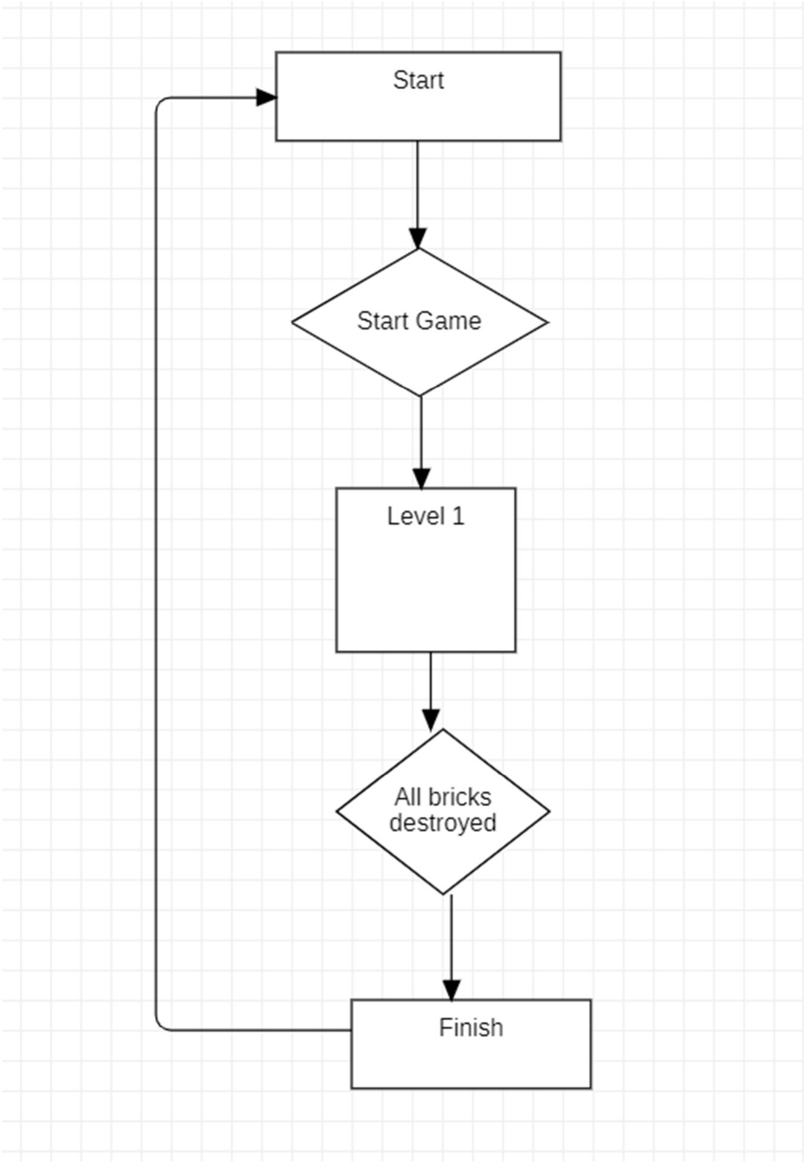
E. Paddle Speed and Size

Two important things to remember.

1. The greater the paddle’s horizontal movement speed, the easier it becomes for the player to reach the ball.
2. The wider the paddle striking surface, the easier it becomes for the player to hit the ball upon reaching it.

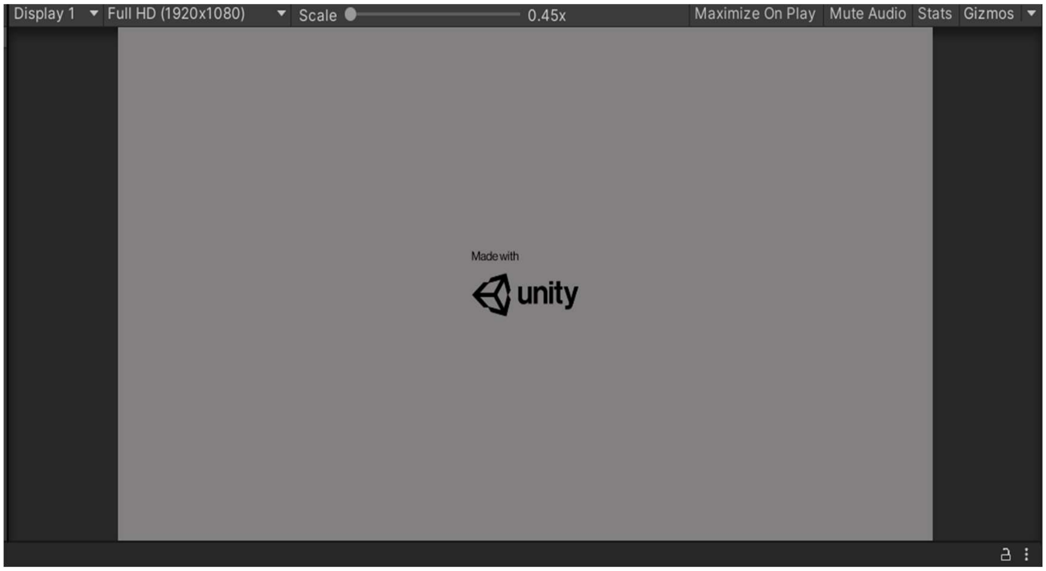
4.2 Procedural Design:

Activity Diagram

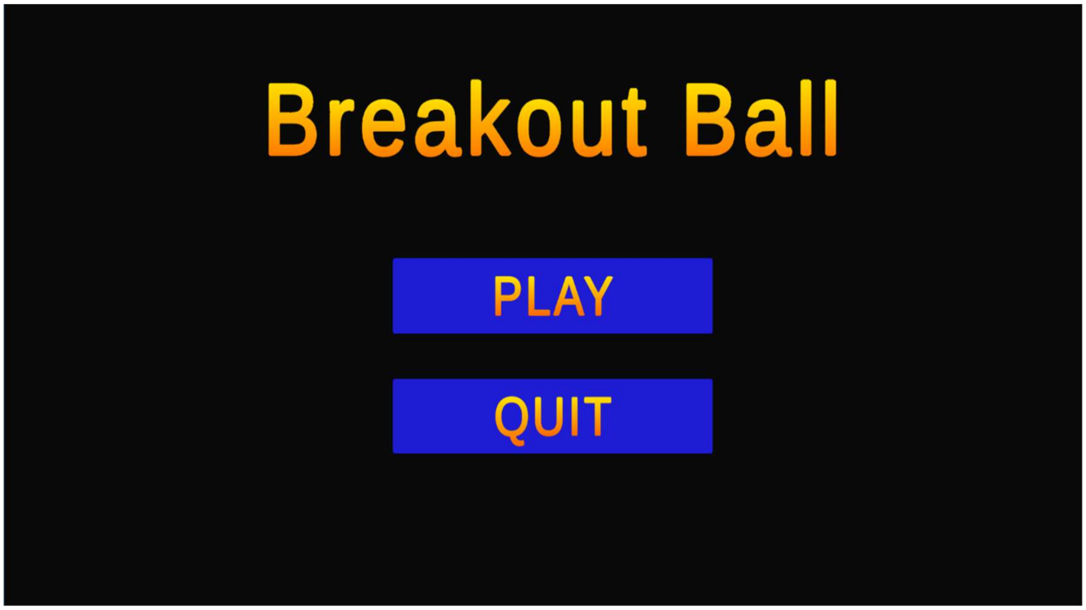


4.3 User Interface Design:

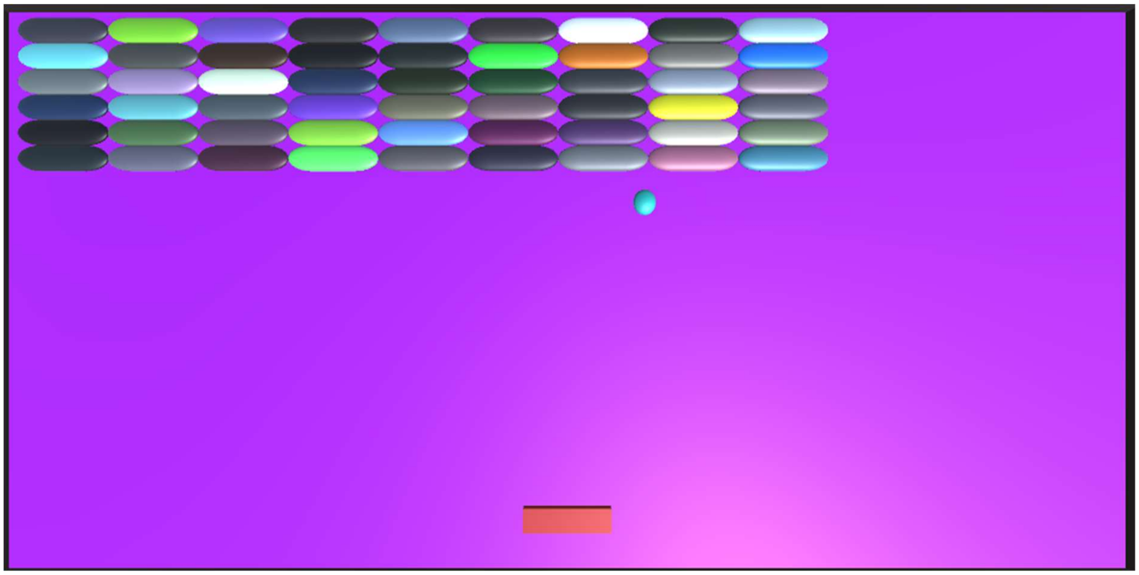
- Start Scene



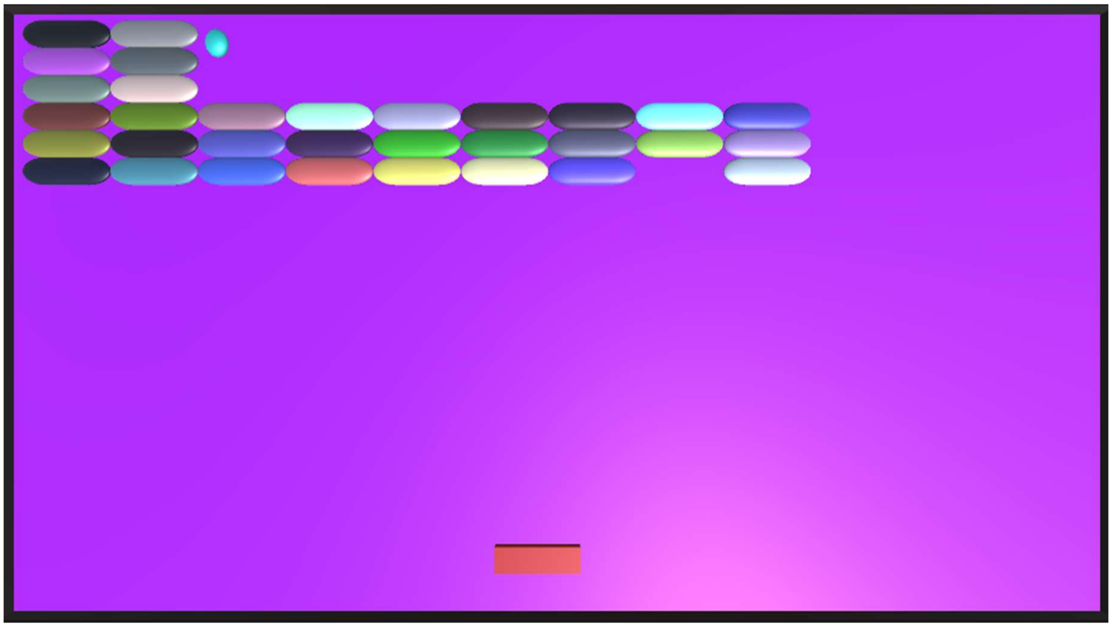
- Main menu



- Main User Interface



- Mid Game Interface



4.4 Test Cases Design

Software testing is defined as an activity to check whether the actual results match the expected results and to ensure that the software system is Defect free.

It involves execution of a software component or system component to evaluate one or more properties of interest.

Software testing also helps to identify errors, gaps or missing requirements in contrary to the actual requirements.

It can be either done manually or using automated tools. Some prefer saying Software testing as a White Box and Black Box Testing.

In simple terms, Software Testing means Verification of Application Under Test (AUT).

Testing is important because software bugs could be expensive or even dangerous. Software bugs can potentially cause monetary and human loss.

Types of Software Testing is classified into three categories.

Functional Testing

Non-Functional Testing or Performance Testing

Maintenance (Regression and Maintenance)

Analyse your test results thoroughly.

Do not ignore any test results. The final test result may be ‘pass’ or ‘fail’ but troubleshooting the root cause of ‘fail’ will give you the solution to the problem. Testers will be respected if they not only log the Bugs but also provide solutions.

Learn to maximize the Test Coverage each time you test any application. 100% test coverage might not be possible but still, try to reach near it.

In order to ensure maximum test coverage, Application is divided into smaller functional modules. Write test cases on such individual unit modules.

Also, if possible, break these modules into smaller parts. It is also known as Application Under Test (AUT).

Chapter 5

Implementation and Testing

5.1 Implementation Approach

In the Iterative model, iterative process starts with a simple implementation of a small set of the software requirements and iteratively enhances the evolving versions until the complete system is implemented and ready to be deployed.

An iterative life cycle model does not attempt to start with a full specification of requirements. Instead, development begins by specifying and implementing just part of the software, which is then reviewed to identify further requirements.

This process is then repeated, producing a new version of the software at the end of each iteration of the model. The advantage of this model is that there is a working model of the system at a very early stage of development, which makes it easier to find functional or design flaws.

Finding issues at an early stage of development enables to take corrective measures in a limited budget. The disadvantages with this SDLC model is that it is applicable only to large and bulky software development projects.

This is because it is hard to break a small software system into further small serviceable increments/modules.

The advantages of the Iterative and Incremental SDLC Model are as follows

- Some working functionality can be developed quickly and early in the life cycle.
- Results are obtained early and periodically.
- Parallel development can be planned.
- Progress can be measured.
- Less costly to change the scope requirements.
- Testing and debugging during smaller iteration is easy.
- Risks are identified and resolved during iteration; and each iteration is an easily managed milestone.
- Easier to manage risk – High risk part is done first.
- With every increment, operational product is delivered.
- Issues, challenges and risks identified from each increment can be utilized/applied to the next increment.
- Risk analysis is better.
- It supports changing requirements.
- Initial Operating time is less.
- Better suited for large and mission-critical projects.
- During the lifecycle, software is produced early which facilitates User valuation and feedback.

5.2 Coding Details

- Main Menu Scene

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

public class SceneLoader : MonoBehaviour
{
public void MainMenu()
{
SceneManager.LoadScene(0);
}

public void LoadGame()
{
//SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex + 1);
SceneManager.LoadScene(1);
}

public void QuitGame()
{
Application.Quit();
Debug.Log("Quit!");
}
}
```

- Bricks

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

public class BallMovement : MonoBehaviour
{
    public GameObject brick;
    public static List<GameObject> objects = new List<GameObject>();
    private float ballspeed = 2.0f;
    private bool dirx;
    private bool dirz;
    private float leftpos = -5.6f;
    private float uppos = 2.54f;

    void Start()
    {
        dirx = false;
        dirz = false;
        createbrick();

                }

private void createbrick()
    {
        for(int i = 0; i < 6; i++)
        {
            for(int j = 0; j < 9; j++)
            {
                GameObject obj = Instantiate(brick, new
Vector3(leftpos,0.15f,uppos),Quaternion.Euler(0,0,90));
```

```
                obj.GetComponent<Renderer>().material.color =
Random.ColorHSV();
                objects.Add(obj);
                leftpos += 1.0f;
            }
            leftpos = -5.6f;
            uppos -= 0.25f;

        }


              }
```

- Brick collision code

```
private void OnTriggerEnter(Collider other)
    {
        if (other.tag == "Brick")
        {
            Destroy(other.gameObject);
            dirz = !dirz;
            PlayerMovement.score += 10;
        }
        if (other.tag == "UpBorder")
        {
            dirz = true;
        }
        if (other.tag == "LeftBorder")
        {
            dirx = false;
        }
        if (other.tag == "RightBorder")
        {
            dirx = true;
        }
        if(other.tag == "DownBorder")
        {
            PlayerMovement.life -= 1;
            transform.position = new Vector3(0, 0.15f, 0);
            dirx = false;
            dirz = false;

        }
        if (other.tag == "Player")
        {
            dirz = false;
        }
    }
}
```

- Ball movement code

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

public class BallMovement : MonoBehaviour
{
    public GameObject brick;
    public static List<GameObject> objects = new List<GameObject>();
    private float ballspeed = 2.0f;
    private bool dirx;
    private bool dirz;
    private float leftpos = -5.6f;
    private float uppos = 2.54f;

    void Start()
    {
        dirx = false;
        dirz = false;
        createbrick();

    }
void Update()
    {
        if (PlayerMovement.gameplay == true)
        {
            if (dirx == true & dirz == true)
            {
                transform.position = new Vector3(transform.position.x - (ballspeed * Time.deltaTime), transform.position.y, transform.position.z - (ballspeed * Time.deltaTime));
            }
```

```
      if (dirx == true & dirz == false)
      {
        transform.position = new Vector3(transform.position.x - (ballspeed *
Time.deltaTime), transform.position.y, transform.position.z + (ballspeed * Time.deltaTime));
      }
      if (dirx == false & dirz == true)
      {
      transform.position = new Vector3(transform.position.x + (ballspeed *
Time.deltaTime), transform.position.y, transform.position.z - (ballspeed * Time.deltaTime));
      }
      if (dirx == false & dirz == false)
      {
         transform.position = new Vector3(transform.position.x + (ballspeed*
Time.deltaTime), transform.position.y, transform.position.z + (ballspeed* Time.deltaTime));
      }

    }



  }
private void createbrick()
  {
    for(int i = 0; i < 6; i++)
    {
      for(int j = 0; j < 9; j++)
      {
        GameObject obj = Instantiate(brick, new
Vector3(leftpos,0.15f,uppos),Quaternion.Euler(0,0,90));
        obj.GetComponent<Renderer>().material.color = Random.ColorHSV();
        objects.Add(obj);
        leftpos += 1.0f;
      }
      leftpos = -5.6f;
      uppos -= 0.25f;
```

```
        }


    }

private void OnTriggerEnter(Collider other)
    {
        if (other.tag == "Brick")
        {
            Destroy(other.gameObject);
            dirz = !dirz;
            PlayerMovement.score += 10;
        }
        if (other.tag == "UpBorder")
        {
            dirz = true;
        }
        if (other.tag == "LeftBorder")
        {
            dirx = false;
        }
        if (other.tag == "RightBorder")
        {
            dirx = true;
        }
        if(other.tag == "DownBorder")
        {
            PlayerMovement.life -= 1;
            transform.position = new Vector3(0, 0.15f, 0);
            dirx = false;
            dirz = false;

        }
        if (other.tag == "Player")
        {
            dirz = false;
        }
    }
}
```

- Player movement code

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
            using UnityEngine.UI;
public class PlayerMovement : MonoBehaviour
{
    private float PlayerSpeed = 2.8f;
    public Text Score;
    public Text Life;
    public Text RESULT;
    public Text TRY;
    public Text MENU;
    public static bool gameplay;
    public static float score;
    public static int life;
void Start()
    {
        score = 0;
        life = 3;
        gameplay = true;
    }
void Update()
    {
```

```
    if (gameplay == true)
    {
      if (Input.GetKey("a"))
        transform.position = new Vector3(transform.position.x - (PlayerSpeed * Time.deltaTime), transform.position.y, transform.position.z);
      if (Input.GetKey("d"))
        transform.position = new Vector3(transform.position.x + (PlayerSpeed * Time.deltaTime), transform.position.y, transform.position.z);

      Score.text = "Score:-" + score;
      Life.text = "YourLife:-" + life;

      if (transform.position.x <= -5.8f)

        transform.position = new Vector3(-5.8f, transform.position.y, transform.position.z);

      if (transform.position.x >= 5.8f)
        transform.position = new Vector3(5.8f, transform.position.y, transform.position.z);



    }
if (score == 540)
    {
      gameplay = false;
      RESULT.text = "YOU WIN";
      TRY.text = "TRY AGAIN";
      MENU.text = "MENU";

    }
    if(life <= 0)
    {
      gameplay = false;
      TRY.text = "TRY AGAIN";
      RESULT.text = "YOU LOSE";
```

```
            MENU.text = "MENU";
        }

    }

}
```

5.3 Testing Approach

5.3.1 Unit Testing

Unit testing, a testing technique using which individual using which individual modules are tested to determine if there are any issues by the developer himself. It is concerned with functional correctness of the standalone modules.

The main aim is to isolate each unit of the system to identify, analyse and fix the defects.

Unit Testing – Advantages:

- Reduces Defects in the Newly developed features or reduces bugs when changing in the existing functionality.
- Reduces Cost of the Testing as defects are captured in very early phase.
- Improves design and allows better refactoring of code.
- Unit Tests, when integrated with build gives the quality of the build as well.

5.3.2 Integrated Testing

- Integration Testing is a level of software testing where individual units are combined and tested in a group.
- The purpose of this level of testing is to expose the faults in the interaction between integrated units.
- **Big Bang** is an approach to integration testing where all the or most of the units are combined and tested at one go.

- **Top Down** is an approach where top-Level units are tested first, and lower-levels units are tested step by step after that.

- **Bottom Up** is an approach where the bottom level units are tested first, and the upper-level units are texted step after that.

- **Sandwich/Hybrid** is an approach where it is a combination of Top down and bottom-up approaches.

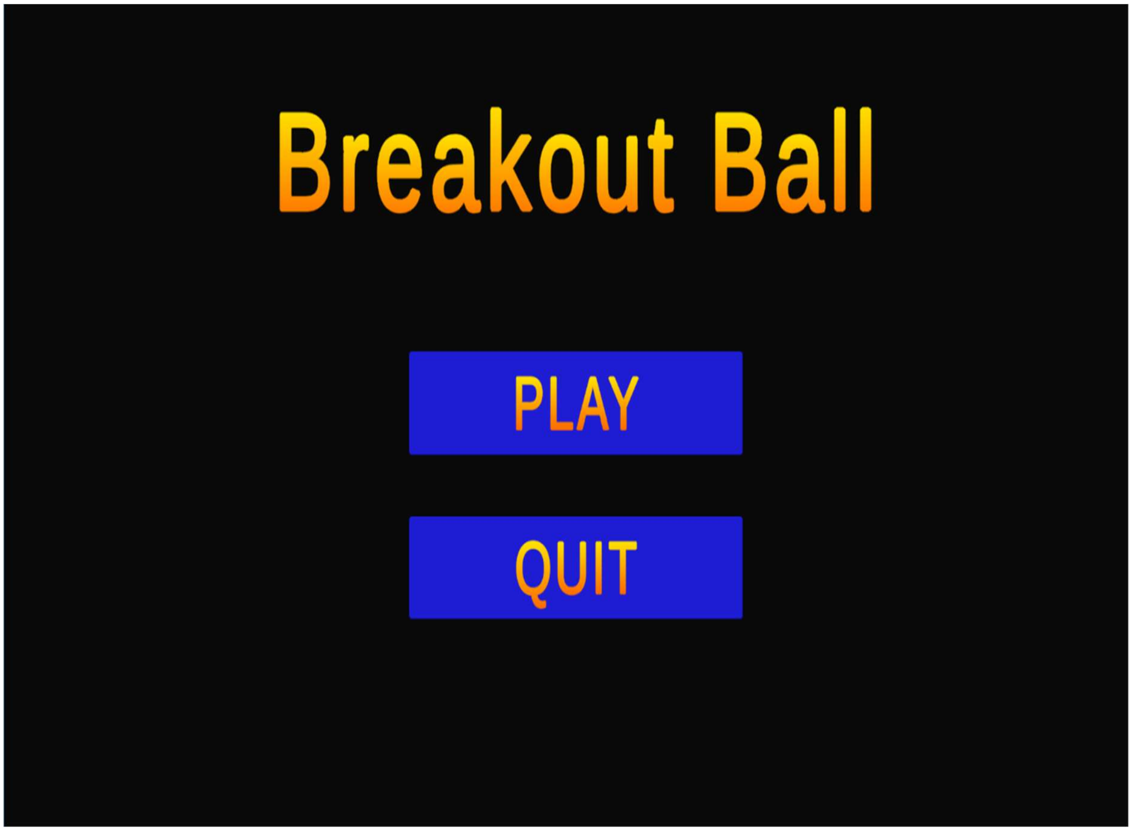
Chapter 6

Results and Discussions

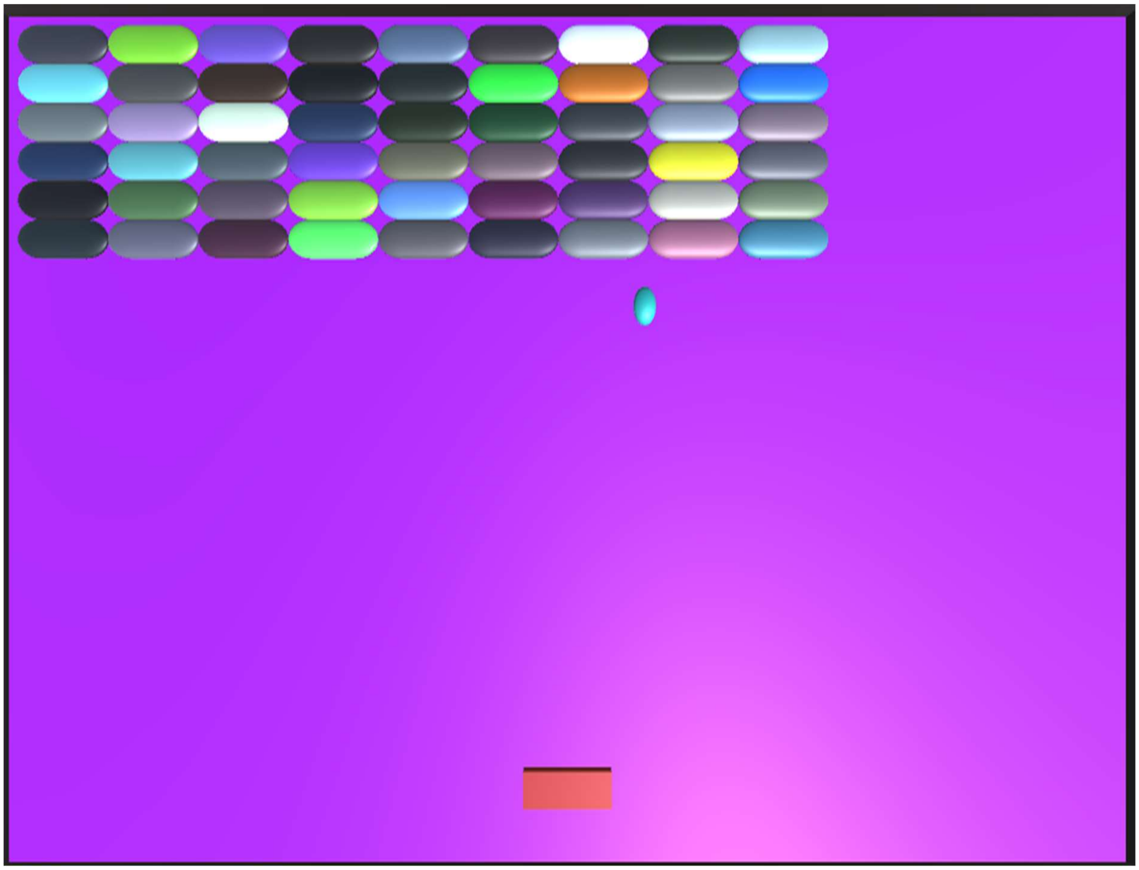
6.1 User Documentation:

- **Start Slide**

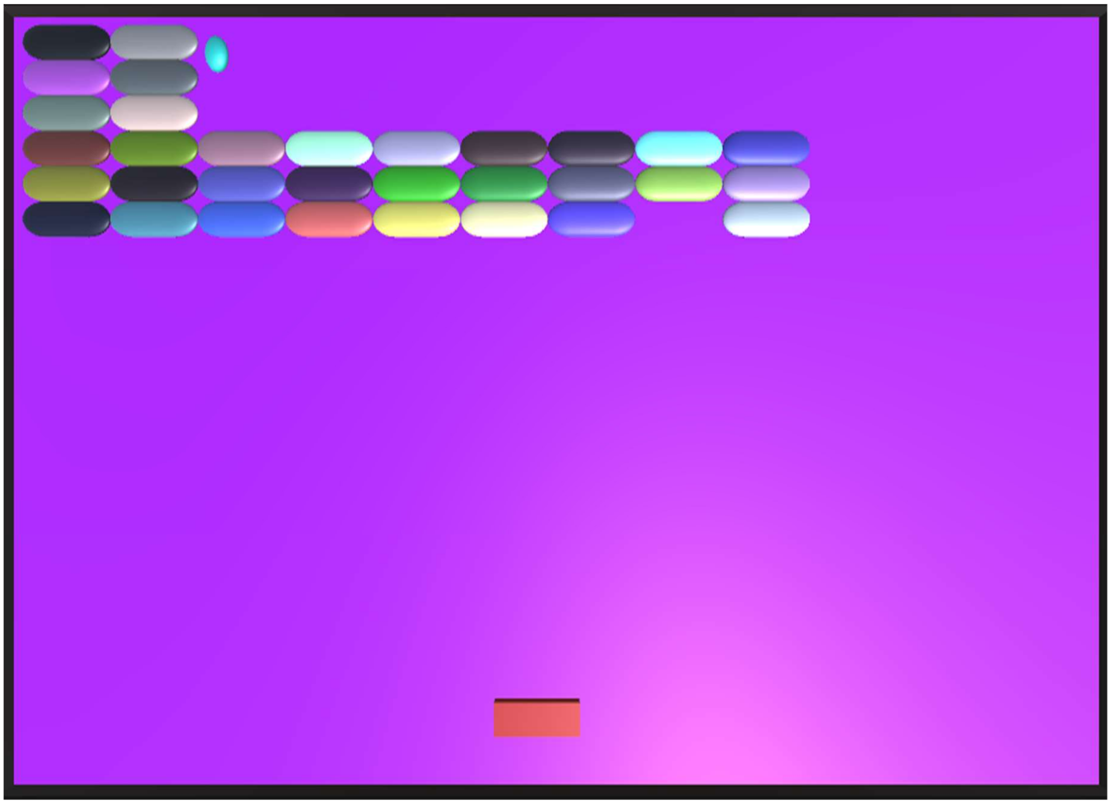
- **Main Menu**



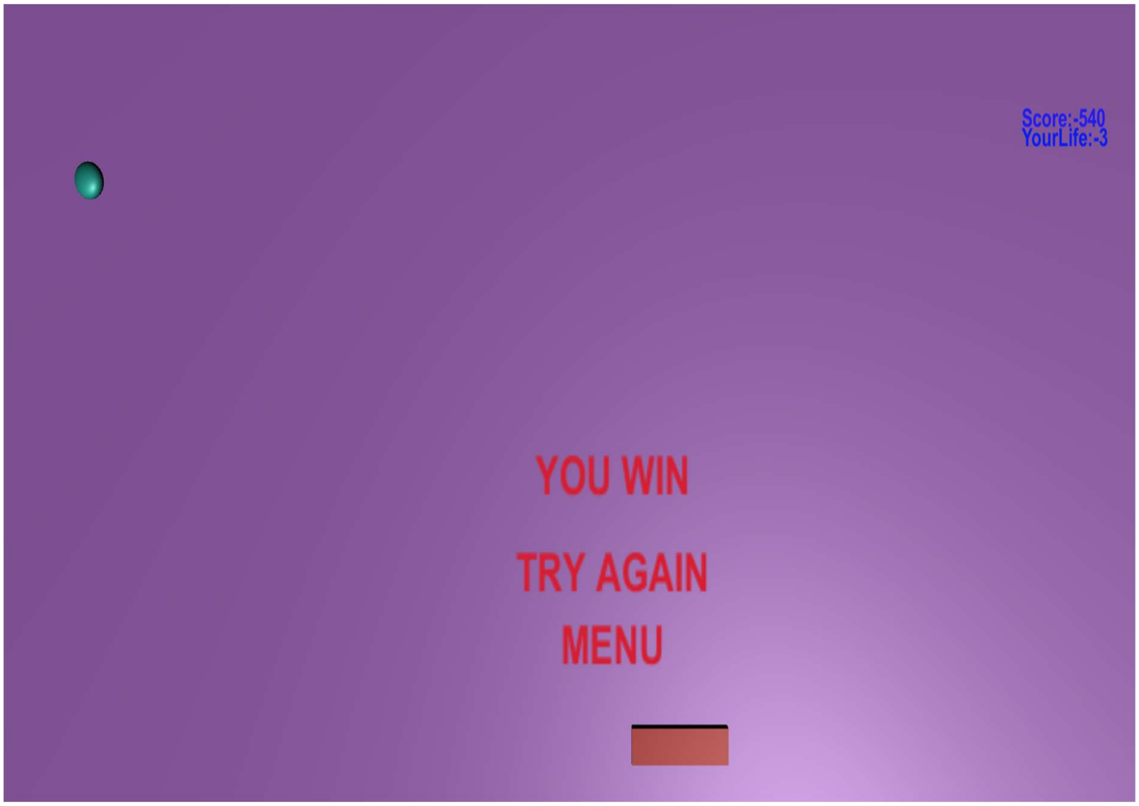
➢ Main User Interface



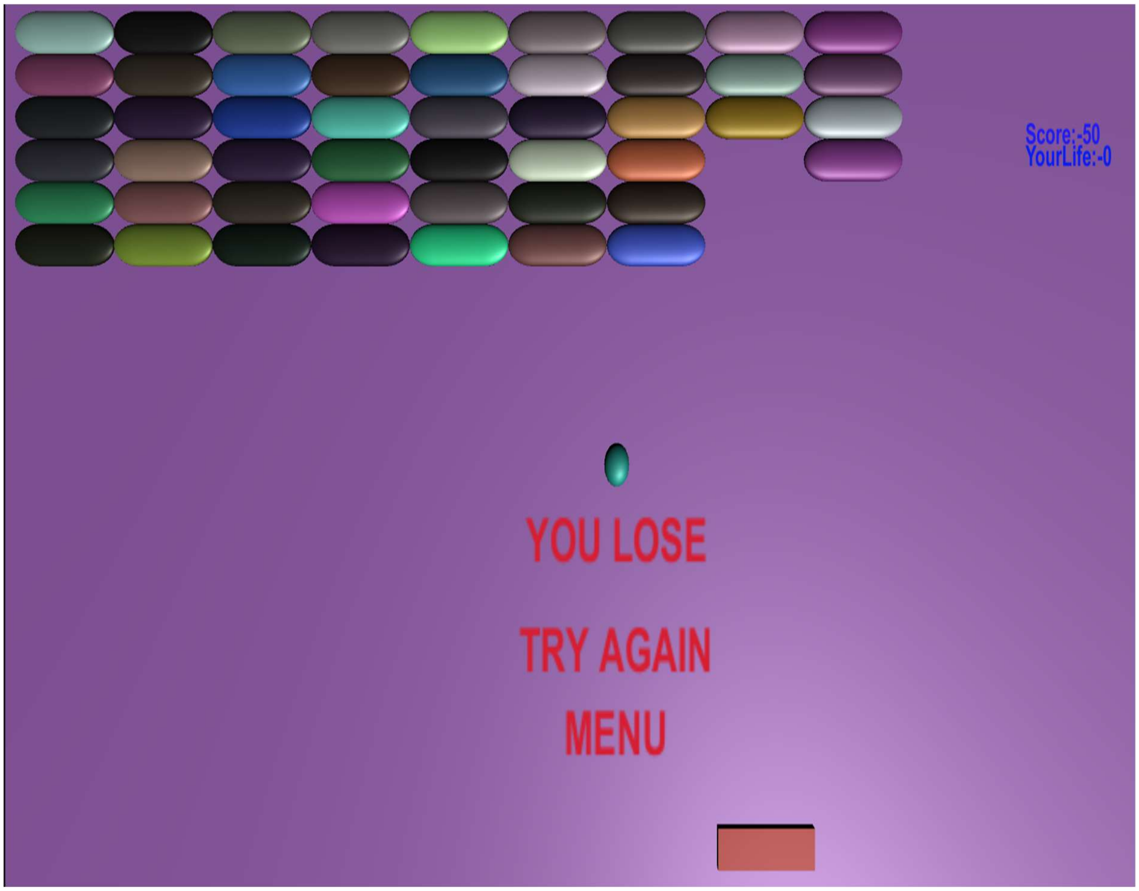
➢ Mid Game Interface



➢ Game Interface (when you win)



➢ Lose Game Interface (when you lose)



Chapter 7
Conclusion and Future Scope

Conclusion:

- This is a fantastic game for fans of arcade action.
- It is developed on Unity platform using “Microsoft Visual Studio”.
- The features of Breakout Ball are: -
  - Easy to access
  - User friendly
  - Easy to handle
- Breakout Ball successfully ran at the end and was a fun experience.

Future Scope:

Some of the plans for future for this application could possibly be:

- To make an IOS version of this app.
- To make the User-Interface more attractive and more user-friendly.
- To make use of modern UI designs and implement it.

References:

https://youtu.be/UgdszzkXga4

https://youtu.be/gCq0nchV4V0

https://youtu.be/NWG8v002oj4

https://youtu.be/EfqzczRaYCs

https://youtu.be/Bzjat5pskrM

https://youtu.be/wr1sqqcijhm

https://youtu.be/yuppqobjvvk

https://youtu.be/-gwja6dixv4

https://youtu.be/zc8ac_quxqy

https://youtu.be/qa012rujxwy