# CHESS [THULLA]

**Final Year Project**

**Session 2016-2020**

A project submitted in partial fulfillment of the degree of

BS in Computer Science


Department of Computer Science

Faculty of Computer Science & Information Technology

The Superior College, Lahore

FALL 2020

| | |
|---|---|
| Type (Nature of project) | [ ✓ ] **D**evelopment [ ] **R**esearch [ ] **R**&**D** |
| Area of specialization | |
| FYP ID | FYP19-Group-080 |

**Project Group Members**

| Sr.# | Reg. # | Student Name | Email ID | *Signature |
|---|---|---|---|---|
| (i) | BCSM-F16-113 | Zain-ul-Abidin | zainjutt@gmail.com | |
| (ii) | BCSM-f16-241 | Saeed Ahmed Sheikh | Bcsm-F16-241@superior.edu.pk | |
| (iii) | BCSM-F16-292 | Amroz Rasheed | Bcsm-f16-292@superior.edu.pk | |

*The candidates confirm that the work submitted is their own and appropriate credit has been given where reference has been made to work of others

## Plagiarism Free Certificate

This is to certify that, I Saeed Ahmed S/D of Shakeel Ahmed, group leader of FYP under registration no FYP-GROUP-080 at Computer Science Department, The Superior College, Lahore. I declare that my FYP report is checked by my supervisor.

Date: Name of Group Leader: _____ Signature: _____

Name of Supervisor: Mr. Umar Farooq Co-Supervisor:

Designation: Junior Lecturer Designation:

Signature: _____ Signature: _____

HoD: Dr. Arfan Jaffar

Signature: _____

# Project Report

## [Chess (Thulla)]

## Change Record

| Author(s) | Version | Date | Notes | Supervisor's Signature |
|---|---|---|---|---|
| Amroz Rasheed | 1.0 | | <Original Draft> | |
| Amroz Rasheed, Saeed Ahmed | | | <Changes Based on Feedback from Supervisor> | |
| Amroz Rasheed, Saeed Ahmed | | | <Changes Based on Feedback From Faculty> | |
| Zain ul Abdin, Amroz Rasheed | | | <Added Project Plan> | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |

# APPROVAL

**PROJECT SUPERVISOR**

Comments: _____

_____

Name:_____

Date:_____ Signature:_____

**PROJECT MANAGER**

Comments: _____

_____

Date:_____ Signature:_____

**HEAD OF THE DEPARTMENT**

Comments: _____

_____

Date:_____ Signature:_____

## Dedication

*This work is dedicated to Allah Almighty our creator, our best companion, our strong pillar, our source of inspiration, wisdom, knowledge and understanding. This work is also dedicated to my supervisor who inspired us to do a different thing because we are going to make a admin panel for a snooker game where as our supervisor advise us to make a unity based game as we learn many new things in it.*

## Acknowledgements

Special thanks to Allah Almighty and His most beloved Prophet, Hazrat Muhammad (Peace be upon Him). We are also really thankful to my supervisor, Sir, Umar Farooq , for his guidance and constant supervision as well as for providing necessary information regarding the project and also for their support in completing the project. We also want to thank to our parents, friends and teachers, who supported us and taught us that we can do anything if we decide.

# Executive Summary

The goal of this project is to implement a First Human Vs Human and Machine Vs Machine Thulla Game. We are three group members. We haven't read the Unity 2d course in our degree but we are fond of developing game that's why we choose this project for making this game we use a unity 2d tool and visual studio for scripting. In this game we have different mode which include up to 4 players mode in private/online Table mode exist which include game mode i-e Easy, Medium, Hard. If you win the game you will receive coins as rewards by which you can play a higher bet game. User will be found it more interesting as well he will achieve levels stepwise and in future, we will try to achieve more exceptional functionalities in our game in update version (login with social media etc...).

# Table of Contents

# List of Figures



# List of Tables

# Chapter 1

# Introduction

# **Chapter 1:** Introduction

In this chapter we will tell about background of this project, the challenges we face for this project, briefly explanation of our goal and objectives, the existing system which is similar to our game, the gap in existing games and our proposed solution. We briefly describe all the project plan of this game which include work breakdown structure, roles and responsibility matrix, Gantt chart and the final report of this game which tell how much time we needed to complete this game.

## 1.1. Background

At the beginning of the project we had no experience in developing a game. But, this time there was a big difference that we were be more determined than ever. As everyone knows, there are several types of computer games. First Human-vs-Human and Human-vs-computer game. A Chess[THULLA][THULLA] Game is a game which makes the player feel within the game world. There are mostly games in Play Store which are not too much attract the user to play a game again and again because of their environment are not in good qualities. If they provide a good quality environment then the user face the crash error problems and sometime mobile performance goes too slow due to high quality graphics. The user get bored after playing this type of game as he does not face difficulties so that's why we introduce 3 modes (easy, medium, hard) ,no user can select the difficulty level in the game. The gaming trend is increasing day by day in all over the world. And according to the research there is a 70% people in the world who love to play games. The gaming trend is growing especially in the kids and youth drastically. Most of the people like to play adventurous games and they become addicted whereas the brain boosting gaming trend is much better compare to the other games. That's why there is a need of such type games which make the people mind sharp and even they get something from game. We are going to develop cards Game in which user can play a game as individual or with friends

## 1.2. Motivations and Challenges

There are many challenges which we face while developing this game some are given below:

- Problems in designing environment for game.
- GUI designing Project Report
- Cards distribution controlling
- Sound controlling
- Memory consumption issues

## 1.3. Goals and Objectives

At the beginning of the project we had no experience in developing a game. But, this time there was a big difference that we were be more determined than ever. The main goal of this project is to learn all the concepts of unity 2d which will help us to build our own business. The purpose of the project is to design and implement a 2-dimensional game written in C# using Unity 2d game development. The project includes a complete level of game with documentation. The level will include everything that should be available in a Thulla game. The game will be a singleplayer. Goals of our project are:

- User friendly efficient and lucrative system.
- Minimum maintenance cost (may be graphics definition).
- Availability of expected requirements within the android device configuration.
- Easy to operate.
- The game with measured coding, professional thinking.
- Game should be enjoyable yet challenging.

## 1.4. Literature Review/Existing Solutions

WHY DO PEOPLE PLAY GAMES? Research on the motivations for games playing has been carried out by researchers across a number of disciplines. One of the earliest, and most cited, research works is by (Thomos Malone, 1952)(Malone 1981) who identified three main ways in which games were able to motivate players: fantasy, challenge and curiosity. Other research confirms these findings; for example, in

research using educational software, Amory et al (1988) identified curiosity ("what happens if I do this") as a common motive in playing a game. Presumably the fact that something does happen encourages players to proceed and the quality of what happens in terms of user engagement is the factor that keeps them playing. The TEEM data suggests that degree of difficulty is important here; for children to enjoy playing, the game must be neither too difficult nor too hard (McFarlane et al 2002). A key concept that frequently emerges in the literature is that of 'flow', first discussed by Csikszentmihalyi (1990). This is summarized by several researchers as "the state in which we are so involved in something that nothing else matters", which has clear relevance to research into games and play. Debate on the issue of 'flow' centers around how the 'state' can be created in an individual, and measuring how it might make a person more receptive to receiving, comprehending and using educational-based content and skills (we will go on to discuss in more detail how 'flow' might apply to the design of learning games in Section 4).A 2001 survey (ESA) produced four main reasons for gameplay, namely: 87% of most frequent computer and video game players said the number one reason they play games is because it's fun. Games are challenging (72%) Games are an interactive social experience that can be shared with friends and family (42%) Games provide a lot of entertainment value for the money (36%). Therefore, no clear consensus emerges on the reasons why people play digital games. This is unsurprising since the games themselves vary enormously and, as some researchers point out, the individuality of the player provides a sometimes complex set of reasons for game play. Poole (2000) notes that: "Videogames are powerful, but they are nothing without humans to play them. So, the inner life of videogames how they work is bound up with the inner life of the player."

### 1.4.1. Categorizing Games

Categorizations are discussed in or want (2000), who also illustrates the system employed by Herz (1997) which closely resembles that used by many in the contemporary games industry. The Herz system presents these major categories:

#### 1.4.1.1 Adventure Game

In most adventure games, the player solves a number of logic puzzles (with no time constraints) in order to progress through some described virtual world.

#### 1.4.1.2 Fighting Games

These involve fighting computer-controlled characters, or those controlled by other players.

#### 1.4.1.3 Puzzle Games

such as Tetris.

#### 1.4.1.4 Role-Playing Games

Where the human players assume the characteristics of some person or creature type

#### 1.4.1.5 Strategy Games

Such as commanding armies within recreations of historical battles and wars. Even with this taxonomy, there are exclusions; a small number of games will be released every year that defy categorization.

### 1.4.2. Existing Systems

#### 1.4.2.1. Bhabhi Thulla Online

When I install this game, there are too many errors which I face while playing this game the controller of this game is not so good it will hang sometimes when a player through a card . **1.4.2.2. Bhabhi.org**
This game is good in AI but it online available in desktop version and graphics is very bad and not user friendly. It features a lot of the same mechanics from the first. The graphics are above average. The game has a surprising amount of depth for being a mobile game. It's a freemium game and that's never ideal.

## 1.5. Gap Analysis

### 1.5.1. Where Are We Now?

We are now standing there where we are developing an our own simples environment which may be based on less textures and less attractive towards user , in this way the numbers of crashes, and memory consuming issue can be less, and user can play the game in easy manner . We will improve the game on our player's requirements so we can identify easily what type of environments that exactly wants, so our team will work more hard on the basis of their comments and feedback.

### 1.5.2. How Are We Going to Close the Gap?

We will close this gap by following these points:

- Comments and Feedback gather from players.
- More train our team for better 2d environments.
- Environments should be good in a view.

• Each objects in environments may be based on better textures and materials.

• The frequency of flickering colors on objects should be less.

• Each environment should consume less memory for any device.

These are basics gap for any games which can be faced by players or our end user, so covering these point may be beneficial for our game in market and our players will get more fun while playing that game.

## 1.6. Proposed Solution

"Chess[THULLA][THULLA]" is an puzzle based 2D first-person cards game where the player can play with friends or with computer in different difficulties level and enjoy with friends. This game is insane for a free game, this has incredible level of gameplay. I think no free game has ever offered a this type of graphics and AI ,In each and every game there should not be good performance with best quality so, we'll provide a game whose GUI will be user friendly and it's performance raise the user intension more towards the game. The player experiences the game world. This makes one feel one is a part of the game.

## 1.7. Project Plan

The Work Breakdown Structure of the project is shown below every phase is divided into many sub tasks. We decided to done all the task together which help us both to enhance our own skill and the task is done under supervision of our supervisor. All the task which we show in WBS is check by our supervisor.

### 1.7.1. Roles & Responsibility Matrix

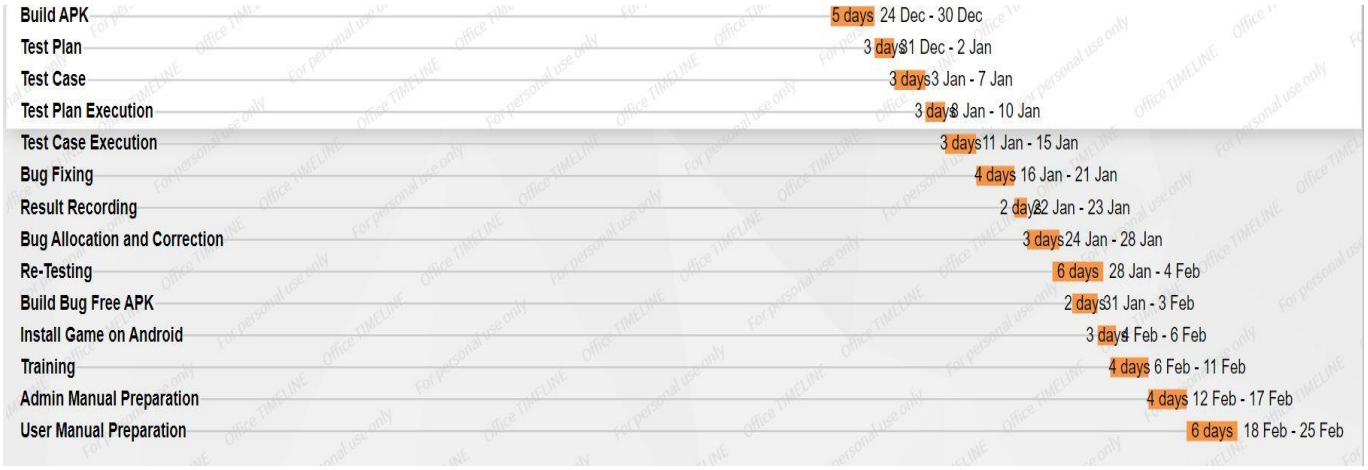
| WBS | WBS Deliverable | Predecessor Activity No | Duration (No. of Days) | Responsible Team Member |
|---|---|---|---|---|
| 1. Introduction | | | | |
| 1.1. | Background | None | 2 | Saeed Ahmed |

| | | | | |
|---|---|---|---|---|
| 1.2. | Motivation & Challenges | None | 3 | Zain ul Abidin |
| 1.3. | Goals and Objectives | 1.1, 1.2 | 3 | Amroz, Zain |
| 1.4. | Existing System Reviews | 1.3 | 2 | Zain .Saeed, Amroz |
| 1.5. | Gap Analysis | 1.4 | 2 | Saeed Ahmed Sheikh |
| 1.6. | Proposed Solution | 1.1, 1.2, 1.3, 1.4,1.5 | 2 | Zain |
| 1.7. | Project Plan | 1.6 | 6 | Zain, Saeed Ahmed, Amroz |
| 2. Software Requirements Specification | | | | |
| 2.1. | Introduction | 1 | 3 | Amroz,Zain |
| 2.2. | Overall Description | 1 | 2 | Saeed Ahmed |
| 2.2.1. | Product Perspective | 2.2 | 2 | Saeed Ahmed, Amroz |
| 2.2.2. | Product Functions | 2.2.1 | 4 | Saeed Ahmed |
| 2.2.3. | User Classes and Characteristics | 2.2.2 | 4 | Zain |
| 2.2.4. | Operating Environment | 2.2.2 | 4 | Saeed Ahmed |
| 2.2.5. | Design and Implementation Constraints | 2.2.3 | 3 | Zain |
| 2.2.6. | User Documentation | 2.2.1 - 2.2.5 | 4 | Amroz |
| 2.2.7. | Assumptions and Dependencies | 2.2.6 | 3 | Saeed Ahmed |
| 2.3. | External Interface Requirements | 2.2 | 6 | Zain,Saeed Ahmed, Amroz |
| 2.4. | System Features | 2.3 | 6 | Amroz, Zain |
| 2.5. | Nonfunctional Requirements | 2.3 | 3 | Saeed Ahmed |
| 3. Use Case Analysis | | | | |
| 3.1. | Use Case Model | 2 | 3 | Amroz, Saeed Ahmed |
| 3.2. | Use Case Description | 3.1 | 4 | Saeed Ahmed |
| 4. System Design | | | | |
| 4.1. | System Architecture Diagram | 3 | 3 | Saeed Ahmed |
| 4.2. | Domain Model | 3 | 3 | Amroz |
| 4.3. | Class Diagram | 4.2 | 3 | Amroz |

| | | | | |
|---|---|---|---|---|
| 4.4. | Sequence Diagram | 4.3 | 5 | Zain, Saeed Ahmed |
| 4.5. | Operation Contracts | 4.4 | 4 | Saeed ahmed,Amroz |
| 4.6. | Activity Diagram | 4.5 | 4 | Amroz , Zain |
| 4.7. | State Transition Diagram | 4.6 | 2 | Amroz,Zain |
| 4.8. | Component Diagram | 4.7 | 3 | Saeed Ahmed |
| 4.9. | Deployment Diagram | 4.8 | 3 | Saeed Ahmed, Amroz |
| 4.10. | Data Flow Diagram | 4.9 | 5 | Zain,Amroz |
| 5. Implementation | | | | |
| 5.1. | Flow Chart | 4 | 3 | Saeed Ahmed |
| 5.2. | Environment Design | 4 | 20 | Amroz,Zain |
| 5.3. | Graphics Design | 5.2 | 10 | Saeed Ahmed |
| 5.4. | UI Design | 5.3 | 10 | Amroz, Zain |
| 5.5. | Scripting | 5.4 | 30 | Saeed Ahmed, Amroz, Zain |
| 5.6. | Build APK | 5.1 – 5.5 | 2 | Saeed Ahmed, Amroz, Zain |
| 6. Testing and Evaluation | | | | |
| 6.1. | Use Case Testing | 5 | 5 | Amroz |
| 6.2. | Equivalence Partitioning | 5 | 5 | Saeed Ahmed |
| 6.3. | Boundary Value Analysis | 6.2 | 5 | Saeed Ahmed |
| 6.4. | Data Flow Testing | 6.1 | 5 | Amroz |
| 6.5. | Unit Testing | 5 | 6 | Zain |
| 6.6. | Integration Testing | 6.3 | 6 | Saeed Ahmed |
| 6.7. | Performance Testing | 6.6 | 4 | Amroz, Zain, Saeed Ahmed |
| 6.8. | Stress Testing | 6.7 | 4 | Saeed Ahmed, Amroz, Zain |
| 6.9. | Final APK | 6.1 – 6.8 | 1 | Saeed Ahmed, Amroz, Zain |
| 7. Conclusions | | | | |
| 7.1. | Summary | 6 | 2 | Saeed Ahmed |
| 7.2. | Achievement Report | 6 | 2 | Zain |
| 7.3. | Critical Reviews | 7.1 | 2 | Saeed Ahmed, Amroz |

| | | | | |
|---|---|---|---|---|
| 7.4. | Learnt Lessons Report | 7.2, 7.3 | 2 | Saeed Ahmed, Zain |
| 7.5. | Future Enhancements | 7.4 | 2 | Amroz, Zain, Saeed Ahmed |

### 1.7.2. Gantt Chart


## 1.8. Report Outline

This chapter is all about the:

- Background
- Motivations and Challenges

- Goals and objectives
- Literature Review
- Gap Analysis
- Proposed Solution
- Project Plan

# Chapter 2

# Software Requirement Specifications

# **Chapter 2:** Software Requirement Specifications

## 2.1. Introduction

### 2.1.1. Purpose

Games have become an integral part of everyday life for many people. A traditional game often presents a situation where “players engage in an artificial conflict, defined by rules and results in a quantifiable outcome. Such artificial conflicts are often represented as a puzzle or a challenge, and having the puzzle solved or the challenge resolved provides a real-world purpose to the game players. This type of games is sometimes referred to as “serious” games. However, this kind of traditional, or “serious games” has been increasingly replaced by electronic games, especially for the so-called “game generation”. This generation typically consists of “digital natives,” who, in contrast to the “digital immigrants” of the older generation, grew up playing a lot of games and who are trained in skills such as “dealing with large amounts of information quickly even at the early ages, using alternative ways to get information, and finding solutions to their own problems through new communication paths”. The Software Requirement Specification document is intended to give a complete overview of Chess [THULLA] including the game mechanics and user interface. The SRS document details all features upon which Chess [THULLA] have currently decided with references to the manner and importance of their implementation. This SRS describes the function, nonfunctional, safety, security and performance requirements of our game project. These include an overview of the project description, functional requirements of systems the project will run on, and characteristics of target users**.**

The purpose of this document is to set up the requirements for the development of Android 2d First person shooting game. The intended customer is really anyone with an android device. A shooting game is simple and fun to play making it available to anyone.

### 2.1.2. Document Conventions

**2D Graphics**: Graphic rendering technique featuring Second-dimensional objects.

**GUI**: Graphical User Interface.

**RLS**: Reload Systems.

**UI**: User Interface.

### 2.1.3. Intended Audience and Reading Suggestions

This project is a prototype for the Chess [THULLA] Game and this document is written for the programmers, designers, developers, testers, documentation writers. This document may be read from front to back for a complete understanding of this game .This has been implemented under the guidance of our supervisor. This document is useful for all of those who want to develop a cards game. The document is intended for developers, project managers, supervisor, users, testers, IT Experts and documentation writers. Any User easily read this document through Table of content.

### 2.1.4. Product Scope

Our Project Scope includes the list of specific project features, functions tasks following: Different levels of Environment We provide different environment on different levels which make our game more adventures and this attract our user to play our game.

**GUI**:

GUI of this game will include splash screen having game logo. After then a menu will appear, having options of all the game modes, coin purchases, and free coins after every defined time, a Spinner, Settings and game Exit etc.

**Private Table**:

In Private mode, the player will have to select the bet amount difficulty mode and number of players/opponents. Number of opponents will increase the difficulty, interest, adventure and thrill of the game.

**Online Table**:

In online table a user can connect with friends with WIFI HOTSPOT and can play with each other. The bet amount will be select by the player who made that table.

### 2.1.5. References

**Bhabhi Thulla Online**

When I install this game, there are too many errors which I face while playing this game the controller of this game is not so good it will hang sometimes when a player through a card . **Bhabhi.org**

This game is good in AI but it online available in desktop version and graphics is very bad and not user friendly. It features a lot of the same mechanics from the first. The graphics are above average. The game has a surprising amount of depth for being a mobile game. It's a freemium game and that's never ideal.

## 2.2. Overall Description

### 2.2.1. Product Perspective

Chess [THULLA] is Puzzle based Cards game which is a family-member of the Cards games category on Google Play-Store. The Game is uploaded to Google Play account and the user after making a search on the Cards based game can download the game. Once the game is downloaded and installed on the android devices, can be played. All the features of the game will be available to the user. While the game itself is not a continuation of any predecessor, its implementation is made possible through the Unity 2D game development tool and its ability to compile and build projects for use on the Android mobile device. As part of its inclusion in the Android framework it is guaranteed security against access from other applications. Since smartphones these days are full of tons of games therefore, one of our goal while designing the game is to be able to let the user understand the gameplay at the first look no matter how often and how least he opens the game, we are achieving this by showing the control guide screen every time at the beginning of the game, hence engaging the user every time with the game. We have tried to avoid creating the story primarily because the users find it really annoying instead our game model focuses on gaining the coins and levels by achieving the trophy.

### 2.2.2. Product Functions

The following is a summary of the major function implemented in the game. They are separated into categories based on those that are necessary for the game to function:

- **Splash Screen**

The splash screen of our game identify the logo of game it will take time and then start the game.

- **Main Menu**

The main menu will be displayed to user in all the settings are appeared,

- **Table Selection**

The user will select the table.

- **Bet Amount, Difficulty Selection, Number of players**

The user have to select the bet amount, difficulty level and number of players with you want to play.

- **Play Game**

The user start the game by click on play button after completion of other settings.

- **Pause Menu**

The player will pause the game when he was playing the game.

- **Resume**

The player can resume game.

- **Settings**

Setting will also be the main part of any game in setting option user can change all the setting of the game like sound option is used to control the sound of game and also the background music of the game and user can check all the controls of the game and learn how to play the game.

- **Home**

Home is also necessary in every game, so it will display in the game.

### 2.2.3. User Classes and Characteristics

User of the games will be the lover of the Cards games specially the category of the people who love to play Puzzled Game and want free Coins with the actual 2D experience. This category is the main target of our business because of the passion and love for cards and Puzzled. More frequent interaction with the game will generate good revenue as well. This category will love the environment and new way of introducing the

Cards game with different modes. Another category of users will be the category with respect to 'age'. This game caters the audience that has an average age of ten to fifteen years. Also the teenagers and users of age 20- 28 will love to play the game as well. Their frequency of use may rely on the level of interest and other factors. One category which is not generally considered is developers and tester who want to play the game for testing purposes as well as to improve the game in terms of features and functionality. New learner of the unity game developer may also take part in playing the games.

### 2.2.4. Operating Environment

Operating environment for Card Game "Chess [THULLA]" is as listed below:

- **Operating system**: Android.
- **Platform**: Unity and Android Studio
- **Language**: C#
- **Minimum Android Version**: Jelly Bean 4.1.

### 2.2.5. Design and Implementation Constraints

Although there are many engines and tools available for game coding, 2d modeling and 2D designing. One should know the pros and corns of software or application before implementing it so that future constraints could not create hurdles to make any game or impose any conditions.

- **Synchronization**: Android version should more than Jelly Bean.
- **Memory**: Device will have 2GB internal hard drive.
- **Language Requirements**: English

### 2.2.6. User Documentation

This Game will come with an "HELP" button, which will allow users to access the offline help manual. This manual will be updated with each new settings. Other user documentation includes one user manual for lowest level users.

### 2.2.7. Assumptions and Dependencies

The final destination of our game's operation will be the Android mobile device. However, Unity will be responsible for both the construction of the game and its integration within the Android framework. Moreover:

• The Game will be virtually be bug free and it will be stable.

• We assume the user will have the correct devices to play the game; which will be either a smart phone for the applications or a computer.

• We assume the user will have sufficient system requirements on which to run the Game.

## 2.3. External Interface Requirements

### 2.3.1. User Interfaces

The user interface of the game will use standards and GUI. Due to the varying age group of users (from younger interns to middle-aged doctors), the GUI needs to adapt to the age group's GUI and gaming preference. The game GUI styles is classic which include all of this:

**• Splash Screen**

The splash screen of our game identify the logo of game it will take time and then start the game.

**• Main Menu**

The main menu will be displayed to user in all the settings are appeared,

**• Level Selection**

The user select different game levels.

**• Gameplay UI**

Gameplay UI will include:

▪ cards ▪ Move

### 2.3.2. Hardware Interfaces

"Chess [THULLA]" is a mobile gaming application designed specifically for the Android platform and is functional on both mobile smart phones and tablets. The device should also contains a SD card for extra storage. "Chess [THULLA]" has been developed for Android Jellybean Version and all new releases android versions. The Android platform is graphically adaptable with 2 dimensional graphics library based on specifications as well as hardware orientation, scaling, pixel format conversion and accelerated 2D graphics.

### 2.3.3. Software Interfaces

The Game runs on Android more than version jellybean embedded operating system. The game features revolutionary touch screen software that makes it convenient to use. The "Chess [THULLA]" uses synchronization that is compatible with Android operating systems. The game contains an in-built image, text and audio files. The built-in characteristics can be updated with additional features by downloading them on the user's device.

## 2.4. System Features

#### 2.4.1. Splash Screen

#### 2.4.1.1. Description and Priority

The splash screen is the screen the player will see every time when start the game. On this screen the game logo. The priority of splash screen is medium.

#### 2.4.1.2. Stimulus/Response Sequences

Step 1: First user press the game icon from apps.

Step 2: Then after 1 second splash screen will displayed to user.

#### 2.4.1.3. Functional Requirements

REQ 1: The title screen must load and appear every time the game is launched.

REQ 2: The time limit of splash screen is 2 seconds

### 2.4.2. Main Menu

#### 2.4.2.1. Description and Priority

The player confronts with the main menu screen. This is the top most priority screen in which user needs to interact with to play game. Here user sees the coins achieved. User sees the “Rate Us” button and “Play” buttons. User can adjust the settings of the play mode and “sound on” and “sound off” functionality.

#### 2.4.2.2. Stimulus/Response Sequences

Step 1: if the player presses the Private/online table play button on the main-menu interface. Step 1.1: The main menu disappears and another panel for Bet amount, difficulty level, number of players selection appears.

Step 2: if the player presses the “Rate Us” button.

Step 2.1: User is directed to the online “rate us” platform.

Step 3: if the player presses the “Setting” button.

Step 3.1: User is prompted with a panel where user can adjust the settings of the “sound on” and “sound off” functionality.

#### 2.4.2.3. Functional Requirements

REQ-1: Main menu screen should contain background image showing the menu background. REQ-2: Main menu panel should contain the level view of user which user has achieved. In case, user is playing game at very first time, zero score should show.

REQ-3: Main menu panel should contain the level view of user which user has achieved even after exiting the game.

REQ-4: Main menu panel should contain the buttons: “Table selection”, “Setting” and “Rate Us”.

REQ-5: if the player presses the “Any table” button the main menu should disappear and another panel for car selection should appear which should contain the arms selection options.

REQ-6: if the player presses the “Setting” button the main menu should disappear and another panel for the setting of game should come on the screen where user should be able to on and off the sound of the game along with the ability to adjust the mode of the game.

### 2.4.3. Level Selection

#### 2.4.3.1. Description and Priority

The difficulty selection screen is the primary way for the player to choose between different modes. The game is separated into multiple modes. On modes Selection panel user have the ability to select the difficulty of his choice.

#### 2.4.3.2. Stimulus/Response Sequences

Step 1: Player selects the table.

Step 2: Game appears with the bet amount, difficulty and number of players selection panel which have multiple levels to show.

Step 3: Player chooses the bet amount.

Step 4: Player chooses the Difficulty mode.

Step 5: Player chooses the number of players.

Step 6: Player is presented with the game play.

#### 2.4.3.3. Functional Requirements

REQ-1: On pressing respective table button user should be shown bet amount, difficulty and number of players selection panel.

REQ-2: after selection game must start with respective settings.

REQ-3: If user has win a level, his level should be increase.

### 2.4.4. Pause Menu

#### 2.4.4.1. Description and Priority

The player should be able to pause anytime during game-play, and this screen fulfills that requirement. The pause menu also allows the player to navigate between game-play and title screens. The portable nature of the console renders player convenience paramount, so this feature must be included.

#### 2.4.4.2. Stimulus/Response Sequences

*Step 1: The player presses the pause button on the game-play interface.*

*Step 2: The level pauses, drawing up the pause menu which prompts the player with three options: "Resume," and "Home".*

*Step 3: The player presses one of the buttons, triggering its respective function.*

#### 2.4.4.3. Functional Requirements

*REQ-1: The "Resume Game" option must continue the game without any change to the character's vector or the state of the level from the moment of the pause action.*

*REQ-2: The "Home" option must start the game from main menu.*

### 2.4.5. THULLA

#### 2.4.5.1. Description and Priority

Whenever the opponent or user through a card on the table. The card counter of the respective player should decrease by one. Different symbol card, the thulla should occur with thulla written animation arrive.

#### 2.4.5.2. Stimulus/Response Sequences

Step 1: User presses the on the respective card.

Step 2: The current should throw on the table.

Step 3: The card counter decreases accordingly.

#### 2.4.5.3. Functional Requirements

REQ 1: Player should throw card on the table.

REQ 2: throwing card should have a sound of throw.

REQ 3: It should effect on the card counter.

### 2.4.6. Winner

#### 2.4.6.1. Description and Priority

The player who's cards counter having Zero cards in it should won the game and takes all the bet amount of all the users/players.

#### 2.4.6.2. Stimulus/Response Sequences

Step 1: after throwing a last card and after chall completed the empty card counter player wins.

#### 2.4.6.3. Functional Requirements

REQ 1: Card counter should be empty after the chall completes should be empty.

REQ 2: Endgame menu should appear.

## 2.5. Other Nonfunctional Requirements

### 2.5.1. Performance Requirements

#### 2.5.1.1. Response Time

There will not be a delay greater than 1 seconds between a user initiating an action, and the system performing it.

**For GUI**

When user press any button in GUI the system will response to user in one second. For **Gameplay**

- When player press the respective card .the throw time of that card on the table should be in 1 second.
- If player gets thulla, the cards on the table should add in the effective player in one and a half second.

### 2.5.2. Safety Requirements

"Chess [THULLA]" will not affect or damage any of the other applications installed on the player's phone. The game will also not cause any overheating of the player's phone; therefore, the phone's internal components will not be damaged. Game should not be played when the player's attention is divided among multiple tasks to prevent potential harm to the player.

### 2.5.3. Security Requirements

"Chess [THULLA]" will not ask for any personal information from the player and will thus be unable to compromise such information. There is no player authentication required to play "Chess [THULLA]". The player simply has to download the application in order to start playing "Chess [THULLA]". That being said, anyone who has access to the player's phone will have the ability to play "Chess [THULLA]".

If any unauthorized player acquires the original player's phone, that unauthorized player will be able to play "Chess [THULLA]". It is the responsibility of the Player to make sure that no unauthorized player/ person will have access to his or her phone.

## 2.5.4. Software Quality Attributes

### 2.5.4.1. Scalability

As this game is for android mobiles so it is scalable with android OS platform.

### 2.5.4.2. Reliability

As this game is for android mobiles so it is reliable with android OS platform.

### 2.5.4.3. Usability

All the interfaces and option in our game will be simple so every user can easily enjoy the game.

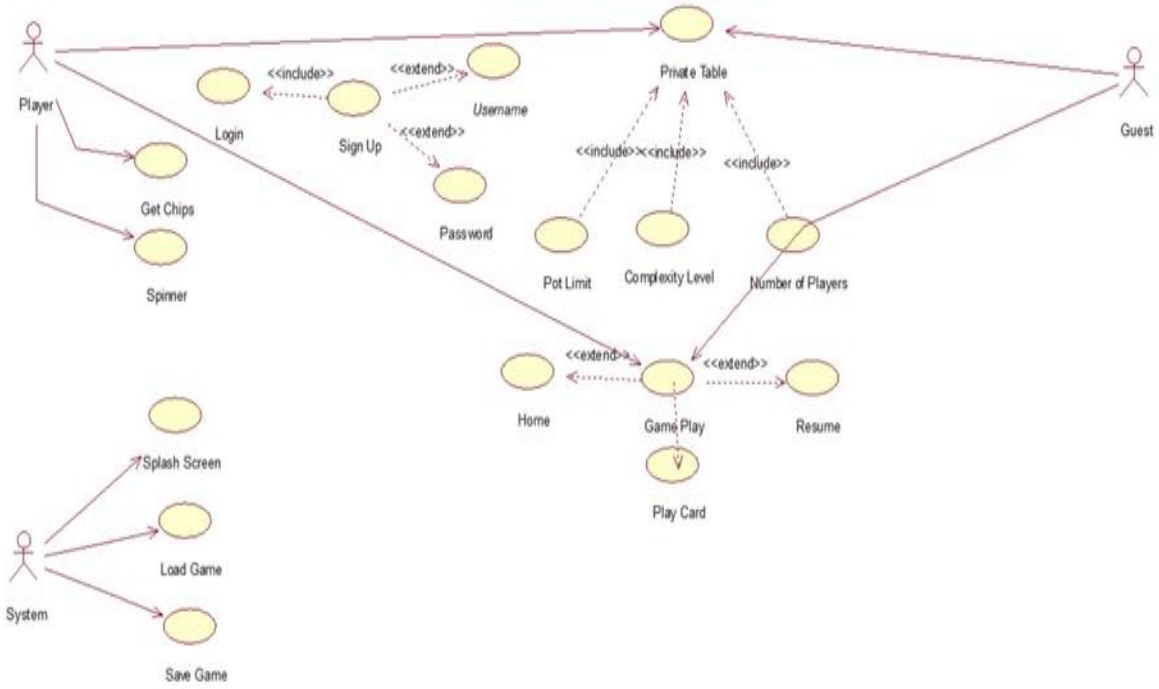
# Chapter 3

# Use Case Analysis

# **Chapter 3:** System Analysis

Use case analysis is a technique used to identify the requirements of a system (normally associated with software/process design) and the information used to both define processes used and classes (which are a collection of actors and processes) which will be used both in the use case diagram and the overall use case.

## 3.1. Use Case Model




## 3.2. Use Case Descriptions

## 3.2. Fully Dressed Use Cases

### 3.2.1. Level Selection

#### 3.2.1.1. ID

WC-01

#### 3.2.1.2. Description

The player will select the mode in order to play game.

#### 3.2.1.3. Primary Actor

User

#### 3.2.1.4. Pre-Conditions

- Game is must on.
- User must select the game mode option.

#### 3.2.1.5. Post Conditions

Gameplay starts.

#### 3.2.1.6. Main Success Scenario

- Open game.
- Touch “private table” button.
- Select mode.
- Loading screen appears.

#### 3.2.1.7. Alternatives

- Level of game menu not shown.
- Some other level load.

### 3.2.2. Quit

#### 3.2.2.1. ID

WC-02

#### 3.2.2.2. Primary Actor

When the user doesn’t want to play game, he/she will quit the game.

#### 3.2.2.3. Pre-Conditions

- Game must be on.

- User are on main menu.

#### 3.2.2.4. Post Conditions

Game will be quitted successfully.

#### 3.2.2.5. Main Success Scenario

- Open main menu.
- Press quit option.
- Game quitted.

#### 3.2.2.6. Alternatives

- Game Crashes.
- Device switches to halt state.

### 3.2.3. Play Game

#### 3.2.3.1. ID

WC-03

#### 3.2.3.2. Description

After selecting the mode, the player will play game.

#### 3.2.3.3. Primary Actor

User

#### 3.2.3.4. Pre-Conditions

- User must start the game.
- User must select the mode to play the game.

#### 3.2.3.5. Post Conditions

User play game successfully.

#### 3.2.3.6. Main Success Scenario

- First user press the game icon from apps.
- Then from main menu touch play button.
- Then select the mode.

#### 3.2.3.7. Alternatives

- Game will be crashed due to some system errors.

- Level may not be loaded because of low specification of mobile at this time game may crashed.

### 3.2.4. Play Card

#### 3.2.4.1. ID

WC-04

#### 3.2.4.2. Description

When the user turn comes, he has to play his/her card.

#### 3.2.4.3. Primary Actor

User

#### 3.2.4.4. Pre-Conditions

- Player must has any card.
- Game must be started.

#### 3.2.4.5. Post Conditions

The player play card successfully.

#### 3.2.4.6. Main Success Scenario

- User touches the card.
- The current card throw on table.

#### 3.2.4.7. Alternatives

- No card of that series.
- The player with big card of another series receive a thulla.

### 3.2.5. Pause

#### 3.2.5.1. ID

WC-05

#### 3.2.5.2. Description

The user must be able to pause the running game, whenever he/she want.

#### 3.2.5.3. Primary Actor

User

#### 3.2.5.4. Pre-Conditions

User must be in play mode of game.

#### 3.2.5.5. Post Conditions

Game will be in pause mode when user press pause button all the functionality of game will be paused like firing and health of player.

#### 3.2.5.6. Main Success Scenario

- When user will playing the game.
- Then press the pause button on the screen.
- Game successfully be in pause mode.

#### 3.2.5.7. Alternatives

- Game will be crashed due to some system error.
- All functionality of game may be not in pause state like health of player may be increase or decrease either the game is in pause state.
- Game may be not paused because pause button may not work.

### 3.2.6. Sound Control

#### 3.2.6.1. ID

WC-06

#### 3.2.6.2. Description

Setting will also be the main part of any game in setting option sound option is used to control the sound of game and also the background music of the game.

#### 3.2.6.3. Primary Actor

User

#### 3.2.6.4. Pre-Conditions

- User must be in the setting menu.
- User must press the sound button of the setting.

#### 3.2.6.5. Post Conditions

The sound is adjusted.

#### 3.2.6.6. Main Success Scenario

- From Main menu of the game user press setting button.

- Then press the sound button.
- Then he can control the sound of the game.

#### 3.2.6.7. Alternatives

- Sound of game not be changed when user volume up or down from sound bar.
- Game will be crashed due to some system errors.
- Sound bar may not displayed 4. Sound bar may not work properly.

### 3.2.7. Restart Game

#### 3.2.7.1. ID

WC-07

#### 3.2.7.2. Description

This use case define that player can restart the game when he/she want to play the game from beginning.

#### 3.2.7.3. Primary Actor

User

#### 3.2.7.4. Pre-Conditions

User must be in pause mode of game.

#### 3.2.7.5. Post Conditions

Game will successfully restarted when user press restart button.

#### 3.2.7.6. Main Success Scenario

- When user will be in pause mode.
- Then press the restart button on the screen.
- Game will restart from beginning.

#### 3.2.7.7. Alternatives

- All functionality of game may be not work properly after restarting of game.
- Game will be crashed due to some system error.

### 3.2.8. Resume

#### 3.2.8.1. ID

WC-08

#### 3.2.8.2. Description

The player will resume the game when he/she wants to continue the game.

#### 3.2.8.3. Primary Actor

User

#### 3.2.8.4. Pre-Conditions

User must be in pause menu.

#### 3.2.8.5. Post Conditions

Game will successfully continue when user press resume button all the functionality of game will behave efficiently like firing and health of player.

#### 3.2.8.6. Main Success Scenario

- When user will be in pause mode.
- Then press the resume button on the screen.
- Game will be continue where user leave.

#### 3.2.8.7. Alternatives

- All functionality of game may be in pause state like health of player either the game is in running state.
- Game will be crashed due to some system error.
- Game may be not continue because resume button may not work.

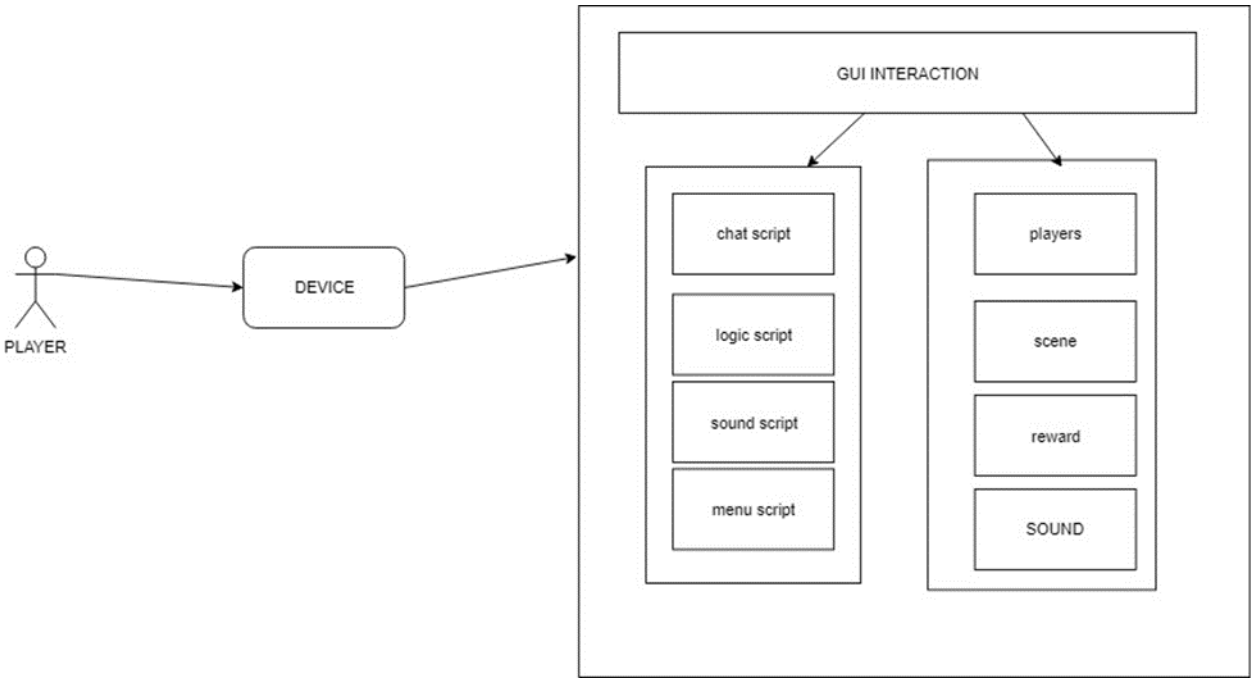
# Chapter 4

# System Design

# **Chapter 4:** System Design

Systems design is the process of defining the architecture, modules, interfaces, and data for a system to satisfy specified requirements. Systems design could be seen as the application of systems theory to product development.

## 4.1. Architecture Diagram



## 4.2. Domain Model

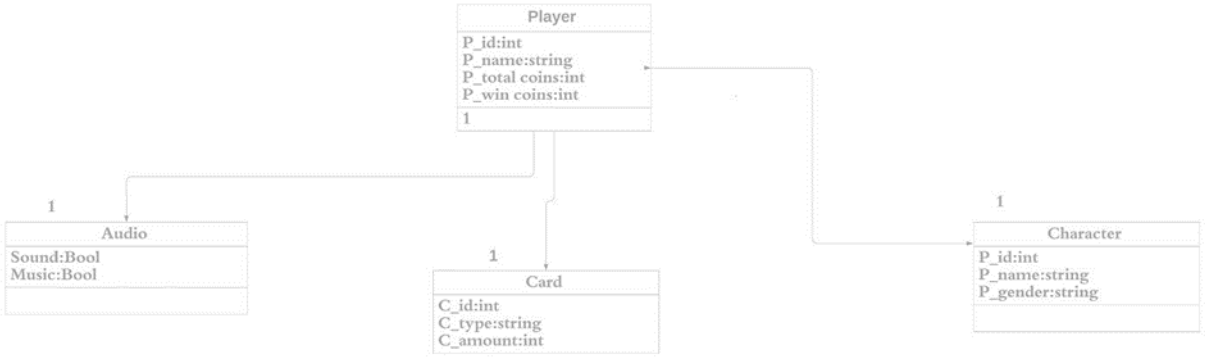
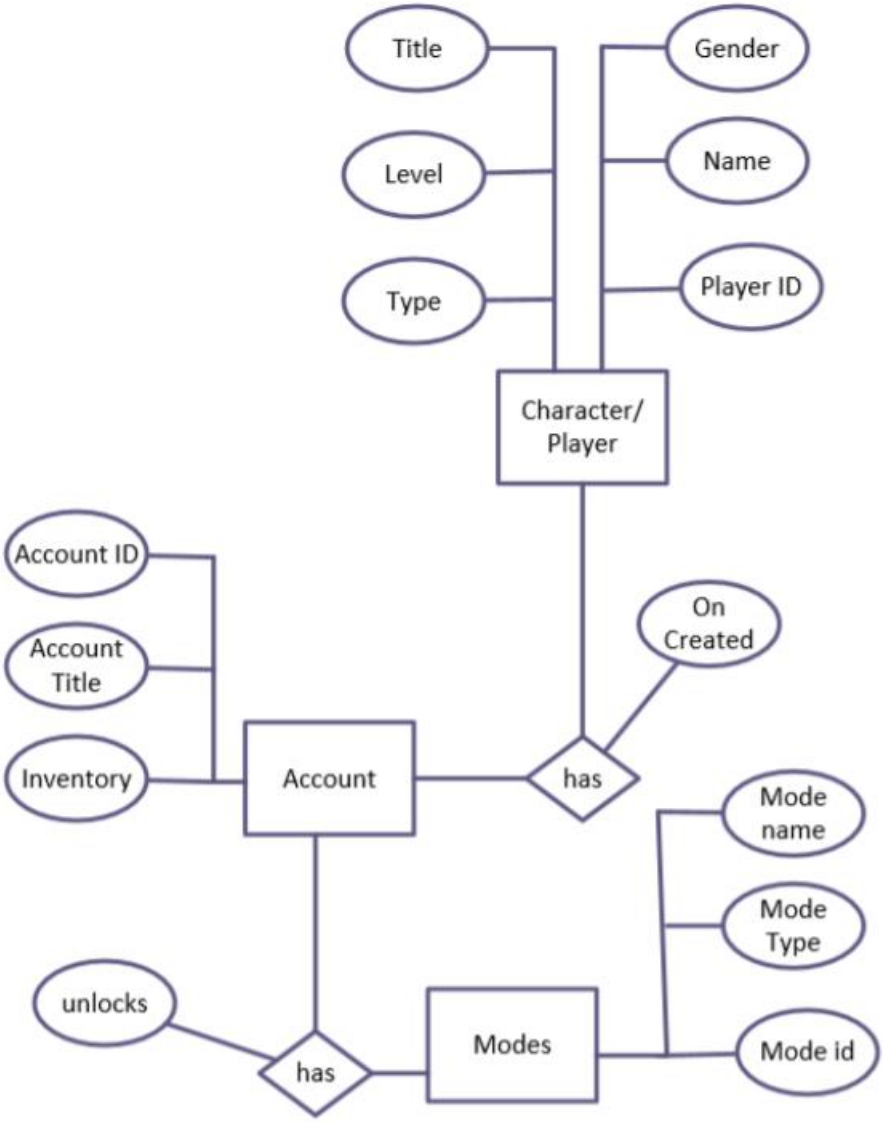



**Figure 4. 2: Domain Model**

## 4.3. Entity Relationship diagram:



## 4.4. Class Diagram

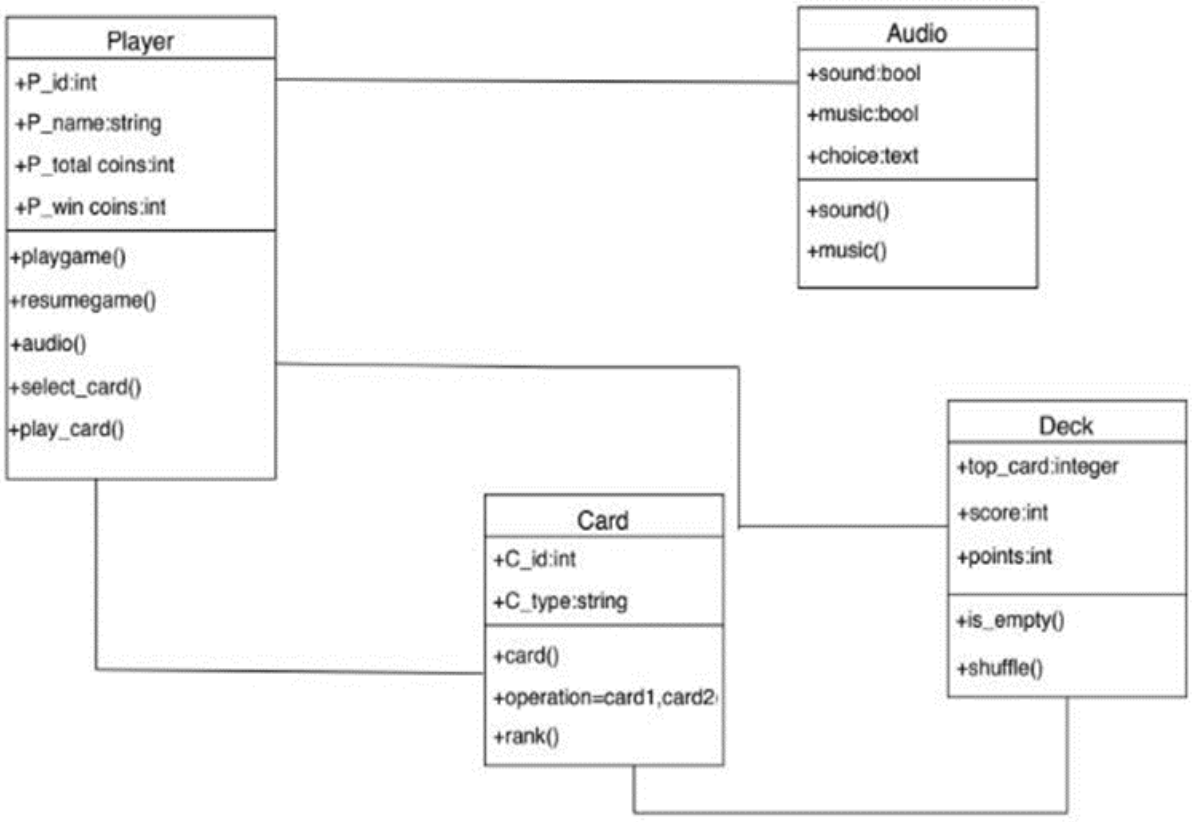
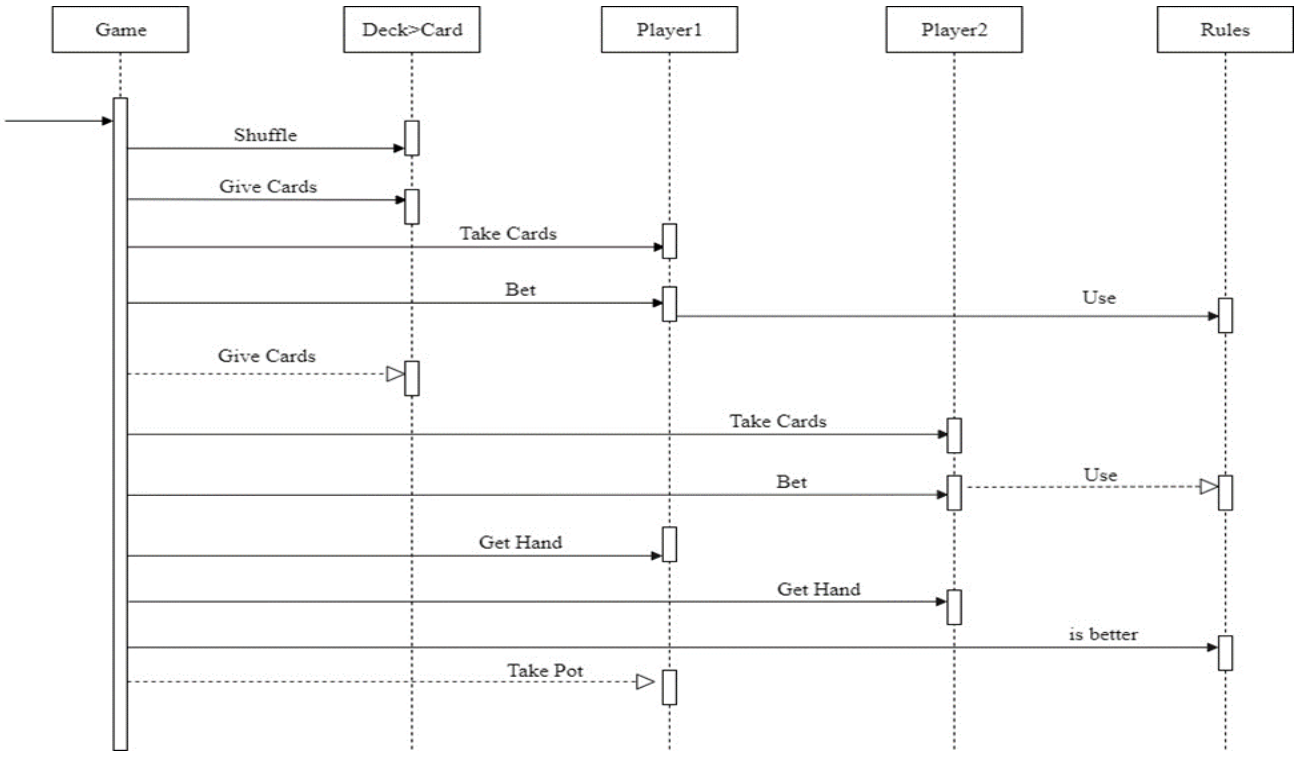


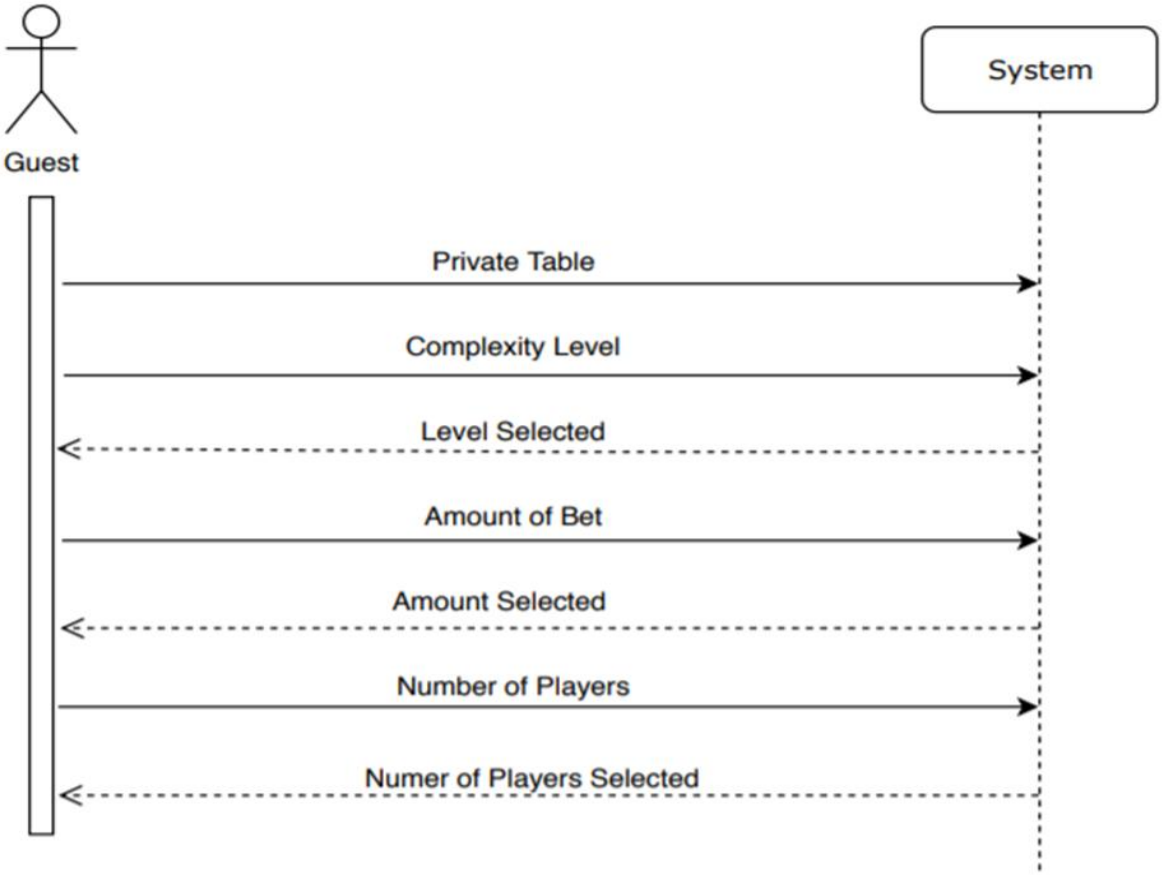

**Figure 4. 3: Class Diagram**

## 4.5. Sequence / Collaboration Diagram

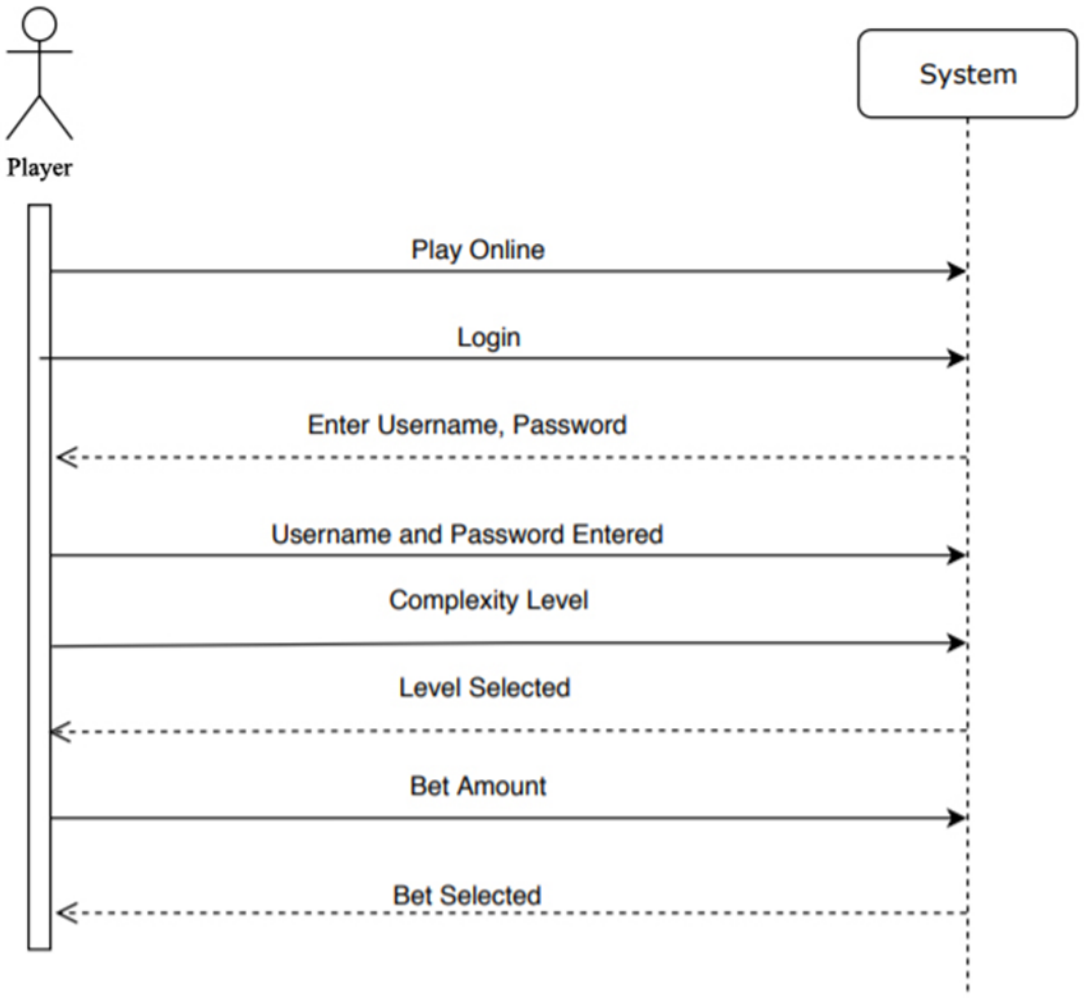


## 4.6. Activity Diagram

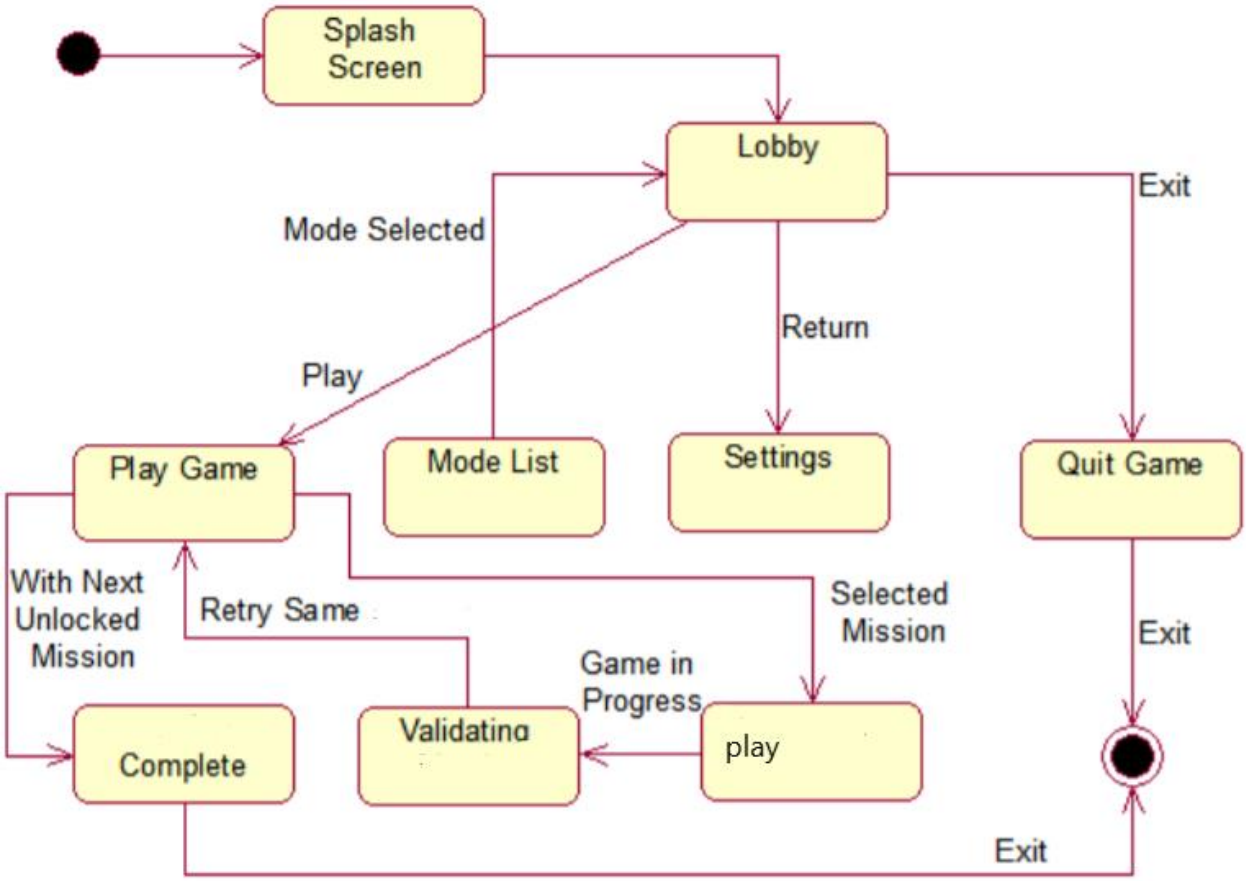
## 4.6.1Guest Sequence Diagram



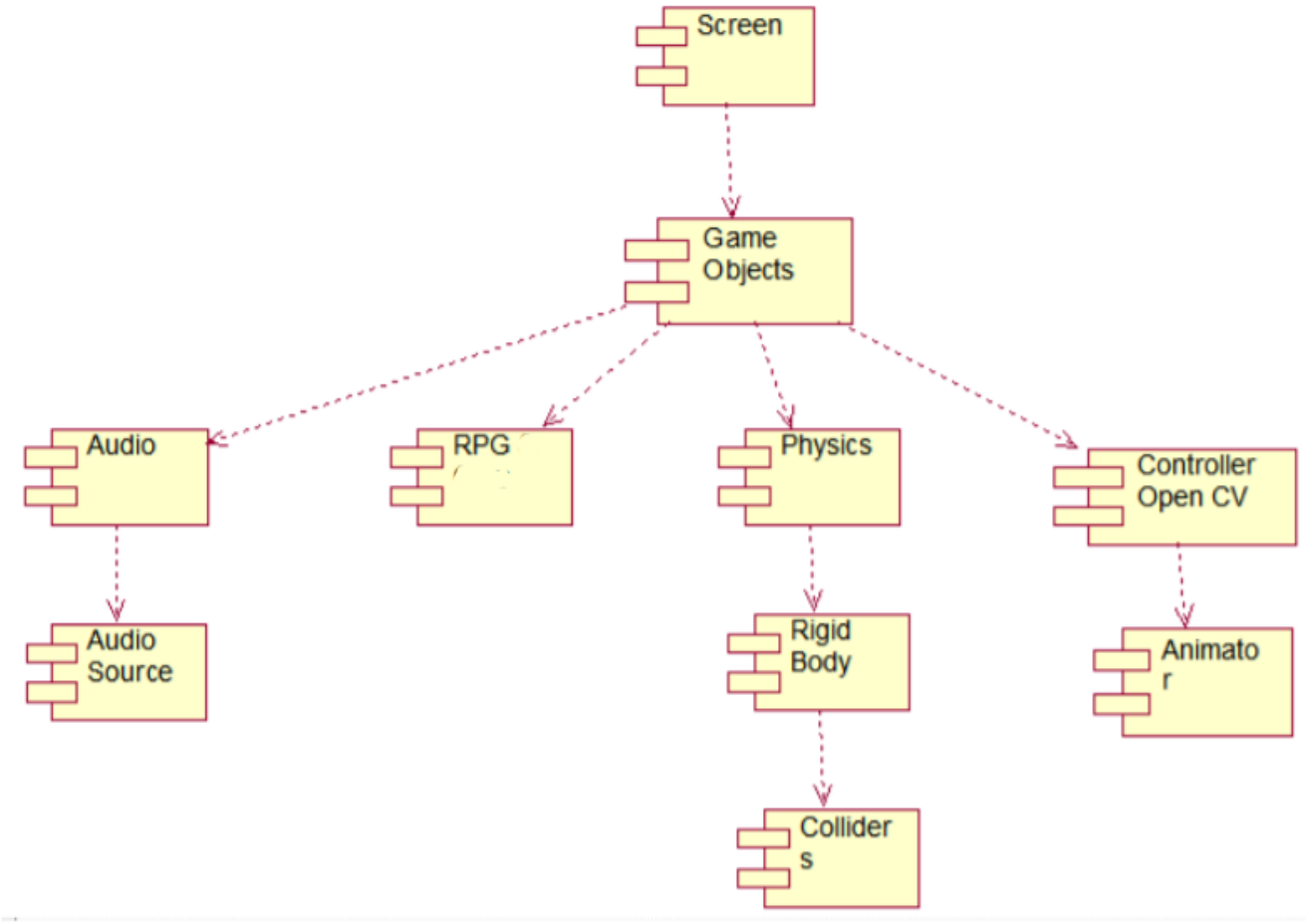
## 4.6.2Player Sequence Diagram



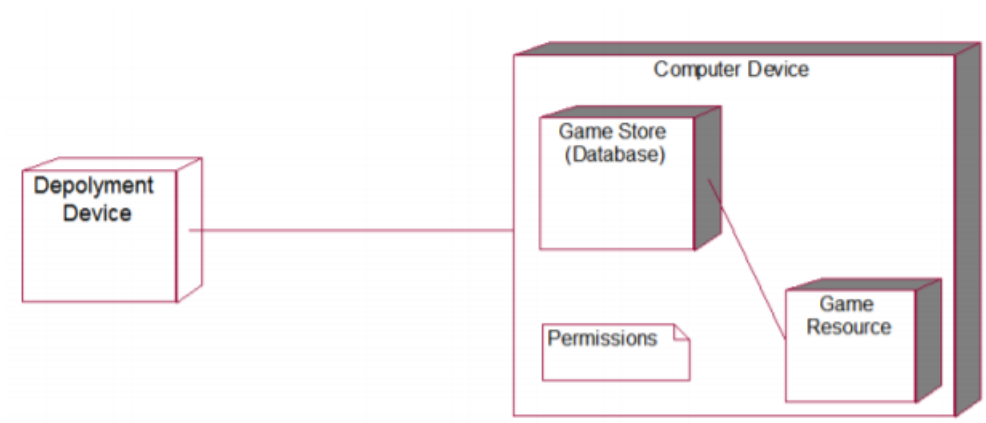
## 4.7.State transition diagram:



## 4.8. Component diagram:



## 4.9. Deployment Diagram:




## 4.10. Operation Contracts

**Table 4. 1: Operation Contracts**

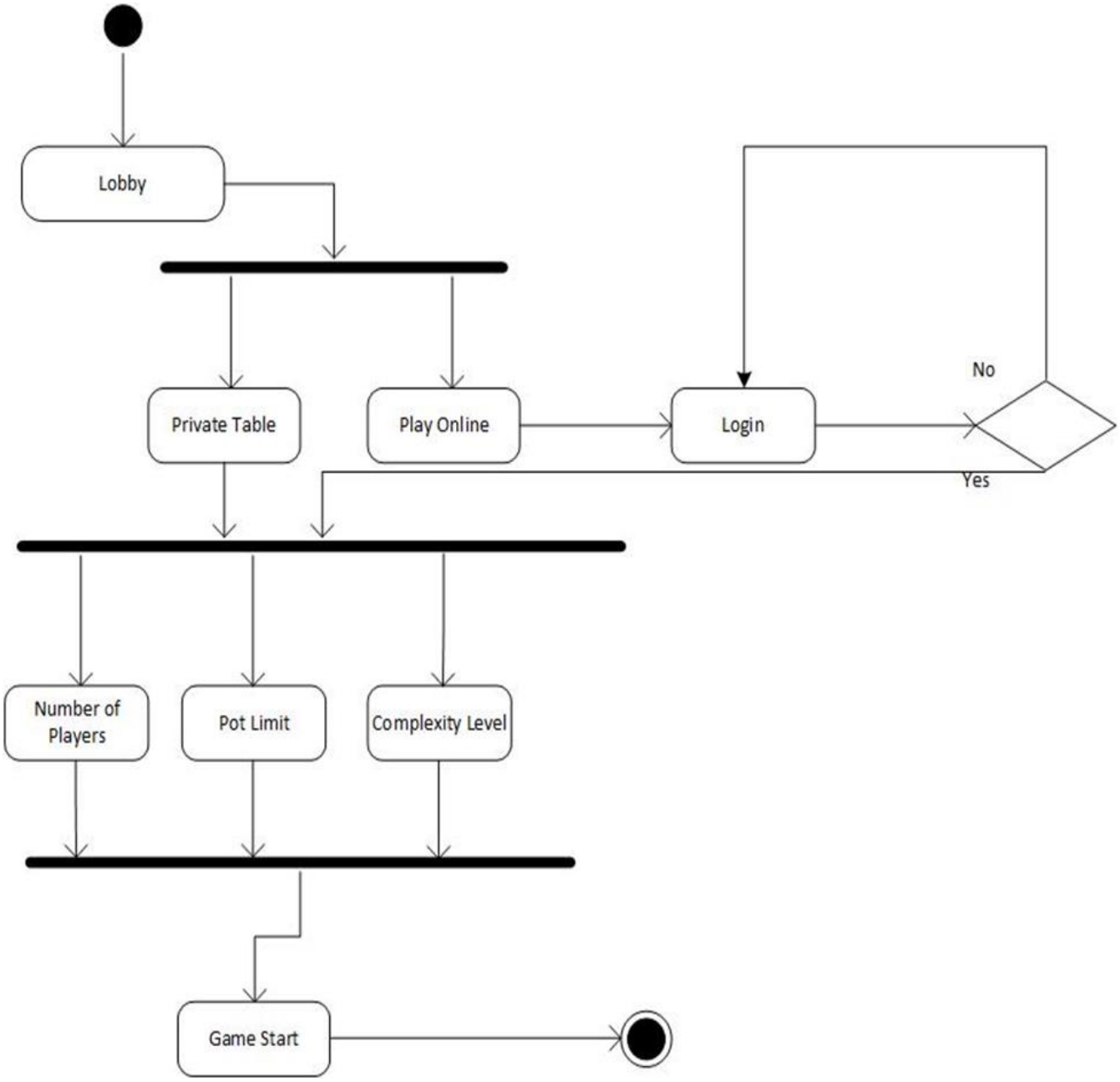
| | |
|---|---|
| **Operation** | chall() |
| **Cross reference** | Play |
| **Pre-Condition** | The game must be in running state |
| **Post Condition** | The health power will be low |

| | |
|---|---|
| | Player will be able to select the option for get high health |
| | |
| **Operation** | isLose() |
| **Cross reference** | Play |
| **Pre-Condition** | Game must be in running state |
| **Post Condition** | Health power will be reduce<br>Message will be displayed You Lose<br>The shooter will be become slow due to low power<br>In case of any other selection, the game will be shift to main menu |
| | |
| **Operation** | IsWin() |
| **Cross reference** | Play |
| **Pre-Condition** | Game must be in running state |
| **Post Condition** | Score will be displayed<br>The message will be displayed You win<br>The game will shift the next level<br>In case of any other selection, the game will be shift to main menu |
| | |
| **Operation** | isEmpty() |
| **Cross reference** | Play |
| **Pre-Condition** | Game must be in running state<br>Shoot command must be passed |
| **Post Condition** | Card slot is empty |
| | |
| **Operation** | Thulla() |
| **Cross reference** | Play |
| **Pre-Condition** | Game must be in running state<br>Thulla command must be passed |
| **Post Condition** | The player will be throw card<br>Thulla player should get the thulla card |
| | |
| **Operation** | No Of cards slot |

| | |
|---|---|
| **Cross reference** | Play |
| **Pre-Condition** | Game must be in running state<br>The character must have cards |
| **Post Condition** | If the cards of the player is empty |

| | |
|---|---|
| | |
| | |
| **Operation** | selectMode() |
| **Cross reference** | Select Mode |
| **Pre-Condition** | Game must be in playing state<br>Select option must be selected |
| **Post Condition** | The menu will be displayed<br>The player will be able to select the Mode option |
| | |
| **Operation** | Select mode() |
| **Cross reference** | Select mode |
| **Pre-Condition** | Level option must be displayed<br>User must be able to select the level |
| **Post Condition** | The level menu will be displayed<br>The player will be able to select the level before to start the game. |

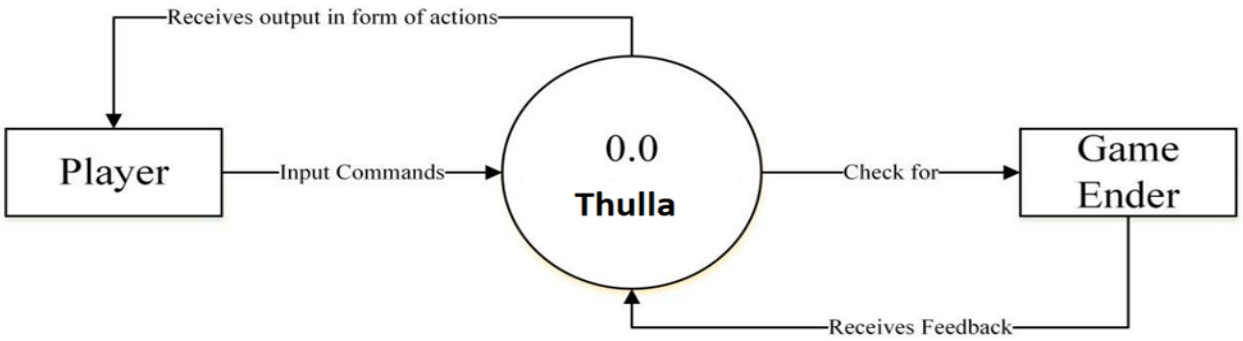
## 4.11. Component Diagram




## 4.12. Data Flow diagram

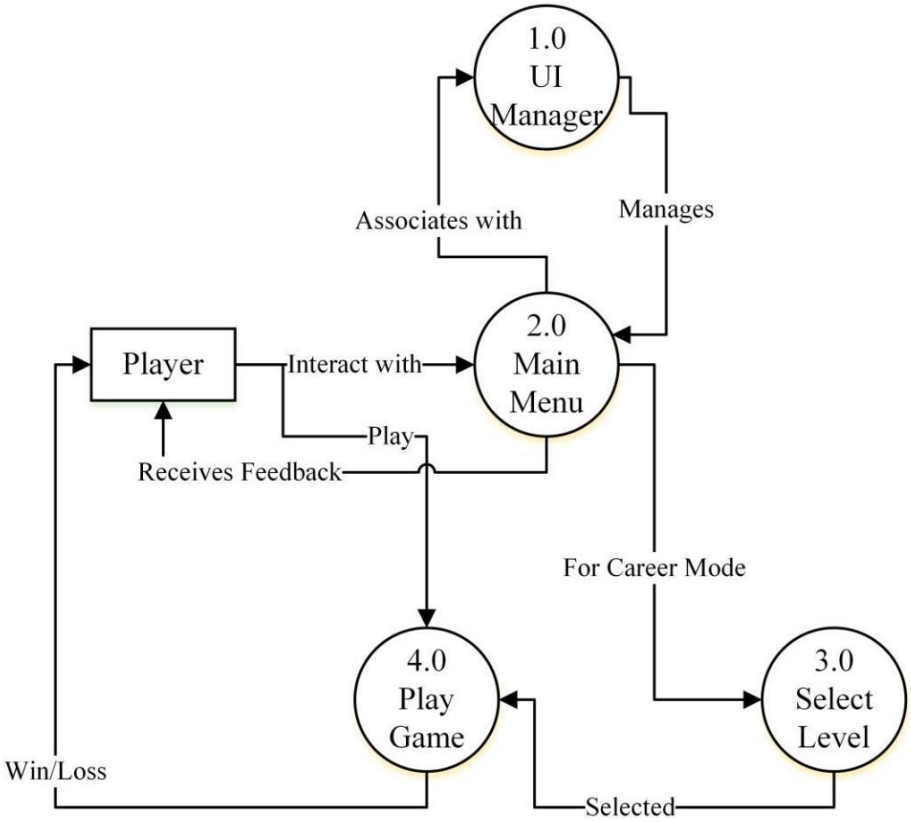
# Level 0

**5. Figure 4. 20: Data Flow Diagram (Level 0)**




# Level 1

**Figure 4. 21: Data Flow Diagram (Level 1)**



# Chapter 5

# Implementation

# **Chapter 5:** Implementation

This chapter covers the implementation of our project features and implementation flow control of our game. This chapter also describes the components, libraries and web services used in our game project. The deployment environment, tools and techniques and coding standards are explained in this chapter that we are using in out game project.

## 5.1. Important Flow Control/Pseudo codes

Following is the Pseudo code of the game that helps to understand how it works and what will be the flow of code:

**START**

**1.** The game is being opened by the Player and
**Game splash screen** starts.

**1.1** ELSE the Player skips the splash screen.

**2.** The **Lobby** Screen appears.

**2.1** IF the Player clicks **Play** and the flow transfers to
**Process ID: 6.Game Initiates.**

**2.2** ELSE the Player clicks **Exit Game.**

**2.3** ELSE the Player clicks **Setting.**

**2.3.1** IF the Player **Sets Volume.**

**2.3.2** ELSE IF the Player **Sets Music**.

**2.3.3** ELSE IF the player **Sets Controls.**

**3.** The **Modes list** of the game appears the mission list (completed and locked) and story mode.

**4.** The coins Selection screen appears that
let Player select the coin and number of players.

**4.1** ELSE IF Player selects the coin that
is availabl, flow transfers to Process ID: 6
Game Initiates.

**4.2** ELSE IF Player selects the story mode flow transfers to **Process ID: 6 Game Initiates.**

**5.** Player plays the game with
keyboard/mouse.

**7.** Player completes the game or story.

**7.1** IF Player lost in the game, **you became Bhabhi** Message
Appears.

**7.2** ELSE IF Player completes the game,
coins get add in the coin list as
completed and flow transfers to Process

**ID: 6. Game Initiates.**

**8.** Player **Pause the game.**

**8.1** IF player clicks the Pause button, and
then clicks the Resume button, flow
transfers to **Process ID: 6. Game Initiates.**

**8.2** ELSE IF player clicks the Pause button,
and then click the Lobby button, flow

transfers to Process

**ID:2 Lobby**

**9.** Player Quits the game.

**9.1** IF Player clicks “No”, flow transfers to Process ID:

**10 Game Menu**.

**10.1** IF Player clicks “Yes”, game Exits

## 5.2. Components, Libraries, Web Services and stubs

Following are the fundamental building blocks in Unity that will make us able to create Game plays for “Bhabhi Thulla”:

### 5.2.1 Components

Components define and control the behavior of Game Objects they are attached to.

Following are different Unity Components that will use to implement our game

- Collider
- Joint
- Animator
- Canvas
- Sprite
- Rigid Body
- Drag
- Scroll
- Particle System
- Prefabs

### 5.2.2 Game Objects

Every kind of content in Unity begins with a Game Object. Any object in our game is a **Game Object:** characters (Avatar, public), special effects, game environment (tables, cards, background) etc.

### 5.2.3 Variables

Components have any number of editable properties that can be tweaked via the Inspector window in the editor, and

### 5.2.4 Libraries

The unity libraries we are used in our game First we are using unity libraries

1. System
2. Unity engine
3. System.Collections
4. DLLTest
5. System.Collections.Generic
6. UnityEngine.Ui
7. UnityEngine.SceneManagement
8. UnityEngine.Audio

### 5.2.5 Scenes

Everything that runs in the game "Bhabhi thulla" exists in a scene. When this game is packaged for a platform, the resulting game is a collection of one or more scenes, plus any platform-dependent code we add. A scene, for example, is a level in a game, though we can have multiple levels in one scene file by just moving the player to different points in the scene.

-There will be 1 scene for introduction game play

-6 scenes for Menu: a scene for menu and five more scenes for five different options of menu (Play, Mission selection, Settings, Exit)

-1 scene for Lobby Selection

-1 scene for mode Selection

-1 scene for Main Game

-1 scene for Loading

-1 scene for Splash Screen

-and so, on.

-Apply physics,

-Respond to user input, and much, much more.

### 5.2.6 C# Scripts

With the use of C# scripts, made us enable to implement own game logic and behavior by simply applying them to the game objects of this game. For the purpose, **Visual Studio** will be used, connected with Unity. The script Components let us to do many things:

#### 5.2.6.1 Loading screen scripts

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


#pragma warning disable 0414

public class GDK_LoadingScreen : MonoBehaviour {


	[SerializeField]float DelayTime;


	[SerializeField]Image LoadingBar;

	bool EnableAd;


	float FadeDuration = 0.5f;
	AsyncOperation Async = null;
	bool LevelLoadComplete = false;
	Scenes CurrentScene;
	public string nextlevel;
	void Awake () {
		InitializeAds ();
	}

	void Start () {
		InitializeScene ();
		StartCoroutine (StartLevelLoad ());
	}

	void InitializeAds () {
```

```
    }

        void InitializeScene () {
                FadeDuration = GDK_Settings.GetFadeDuration ();
                //LoadingBar.fillAmount = 0;
                if (GameManager.Instance.IsSFXControllerAvailable ()) {
                        GDK_SoundController.Instance.PlayMusicLoop (SoundType.MenuMusic);
                }
        }

        void Update () {
                if (Async != null && !Async.isDone) {
                        if (LoadingBar.fillAmount < 0.8f) {
                                LoadingBar.fillAmount = Mathf.Lerp (LoadingBar.fillAmount,
Async.progress, Time.deltaTime);
                        } else if (LoadingBar.fillAmount >= 0.8f && LoadingBar.fillAmount != 1f) {
                                LoadingBar.fillAmount = Mathf.Lerp (LoadingBar.fillAmount, 1,
Time.deltaTime * 5.0f);
                                if (LoadingBar.fillAmount >= 0.98f && Async.progress >= 0.9f &&
!LevelLoadComplete) {
                                        LevelLoadComplete = true;
                                        StartCoroutine (ActivateLevel ());
                                }
                        }
                }
        }

        IEnumerator StartLevelLoad () {
                yield return new WaitForSeconds (DelayTime);
                if (GameManager.Instance.online == 0)
                {
                        Async = SceneManager.LoadSceneAsync("GamePlay_");
                }
                else
                {
                        Async = SceneManager.LoadSceneAsync("GamePlay_ 2");

                }
                Async.allowSceneActivation = false;
                yield return Async;
        }

        IEnumerator ActivateLevel () {
                if (GameManager.Instance.NextSceneToLoad.Equals (nextlevel)) {
                        if (GameManager.Instance.IsSFXControllerAvailable ())
                                GDK_SoundController.Instance.PlayMusicLoop (SoundType.GamePlayMusic);
                }
                Fader.Instance.FadeIn (FadeDuration);
                yield return new WaitForSeconds (FadeDuration + 0.2f);
                Async.allowSceneActivation = true;
                Fader.Instance.FadeOut (FadeDuration);
        }
}
```

### 5.2.6.2 Main Menu Scripts:

```
using UnityEngine;
using System.Collections;
using UnityEngine.UI;
using VoxelBusters.NativePlugins;
using Sirenix.OdinInspector;
using DG.Tweening;
[HideMonoScript]
public class GDK_MainMenu : MonoBehaviour {
[BoxGroup("Advertisements-Meta")] [GUIColor(1f, 1f, 0f)] [SerializeField]
        private bool ShowInterstitial;
        [BoxGroup("Scene Change - Meta")]
        [ReadOnly]
        [SerializeField]float FadeDuration = 0.5f;
        [BoxGroup("Scene Change - Meta")][ReadOnly]
        [SerializeField] bool StartLoadNextScene = false;
        [BoxGroup("Scene Change - Meta")]
        [SerializeField] bool EnableLoadingScreen = false;
        [BoxGroup("ModeSetting")]
        [SerializeField]Scenes NextScene;
        [BoxGroup("ModeSetting")][ShowIf("EnableSurvivalMode",true)]
        [SerializeField]Scenes SurvivalModeScene;
        [BoxGroup("ModeSetting")]
        [SerializeField]bool EnableSurvivalMode;
        [BoxGroup("Share - Meta")]
        [SerializeField]string ShareMessage;
        [FoldoutGroup("OtherReferences")]
        [SerializeField]GameObject GDPRScreen;
        [FoldoutGroup("OtherReferences")]
        [SerializeField]GameObject SettingsScreen;
        [FoldoutGroup("OtherReferences")]
        [SerializeField]GameObject ExitScreen;
        Scenes CurrentScene;
    [BoxGroup("GameMiniScreens - Data")]
    public RectTransform ProfileDataScreen;
    [BoxGroup("GameMiniScreens - Data")]
```

```
public RectTransform PlayofflineScreen;
[BoxGroup("GameMiniScreens - Data")]
public RectTransform SettingScreen;
[BoxGroup("GameMiniScreens - Data")]
public RectTransform AchiementsScreen;
[BoxGroup("GameMiniScreens - Data")]
public RectTransform SpinnerFreeScreen;
[BoxGroup("GameMiniScreens - Data")]
public RectTransform InfoScreen;
[BoxGroup("GameMiniScreens - Data")]
public RectTransform GameSettingScreen,CoinStoreScreen,ChangeNamePanel,ErrorPanelForNoCoins;
public GameObject fadeScreen;
[BoxGroup("SetUpPrivateTable - Data")]
public Slider BetAmountSlider;
[BoxGroup("SetUpPrivateTable - Data")]
public Text BetAmount;
[BoxGroup("Profile Data Variables")]
public Text OnScreenUserName, OnScreenUserNameInMainMenu;
[BoxGroup("Profile Data Variables")]
public Slider OnScreenUserRank, OnScreenUserRankOnMainMenu;
[BoxGroup("Profile Data Variables")]
public Text ProfileuserCoins,MainScreenUserCoins;
[BoxGroup("Profile Data Variables")]
public Text ProfileuserLevel;
[BoxGroup("Profile Data Variables")]
public Text ProfileuserTotalgamePlayed;
[BoxGroup("Profile Data Variables")]
public Text ProfileuserTotalgameWon;
[BoxGroup("Profile Data Variables")]
public Text ProfileuserTotalgameLose;
void Awake () {
                GameManager.Instance.InitializeGame ();
                IntializePrivacy();
                InitializeAds ();
        }
        void Start () {
```

```
                        GameManager.Instance.Gameplayers = GamePlayers.none;
                        InitializeScene ();
                        InitializeSound();
                    SaveLoadManager.LoadGame();
                    SetupProfileData();
                }
                public void SetupProfileData()
                {
                        float gamewon = SaveData.Instance.gameWon;
                        float gameplayed = SaveData.Instance.GamePlayed;
                        GameManager.Instance.EntryFee = 0;
                        GameManager.Instance.PrizeAmount = 0;
                    OnScreenUserName.text = SaveData.Instance.userName.ToString();
                        OnScreenUserNameInMainMenu.text = SaveData.Instance.userName.ToString();
                        OnScreenUserRank.value = (gameplayed / 100);
                        OnScreenUserRankOnMainMenu.value = (gameplayed / 100);
                        ProfileuserCoins.text = SaveData.Instance.UserCoins.ToString();
                        MainScreenUserCoins.text = SaveData.Instance.UserCoins.ToString();
                        ProfileuserLevel.text = SaveData.Instance.Level.ToString();
                                if (SaveData.Instance.GamePlayed == 0)
                            ProfileuserTotalgameLose.text = "0 %";
                        else
                        ProfileuserTotalgameLose.text = ((gamewon / gameplayed) *100).ToString() + "%";

                    ProfileuserTotalgameWon.text = SaveData.Instance.gameWon.ToString();
                    ProfileuserTotalgamePlayed.text = SaveData.Instance.GamePlayed.ToString();
}
                    void Update () {
                        //if (Application.platform == RuntimePlatform.Android) {
                        //      if (Input.GetKey (KeyCode.Escape)) {
                        //              OpenExitScreen ();
                        //      }
                        //}
                    }
        void InitializeSound()
                    {
```

```
                if (GameManager.Instance.IsSFXControllerAvailable ())
                        GDK_SoundController.Instance.PlayMusicLoop (SoundType.MenuMusic);
        }
void IntializePrivacy()
        {
    #if  USE_ADMOB || USE_ADCOLONY || USE_UNITYADS
                if (!Advertisements.Instance.UserConsentWasSet())
                {
                        PrivacyDialogue();
                }
                #endif
        }
        void InitializeAds () {
                CurrentScene = Scenes.MainMenu;
if (GameManager.Instance.FirstLaunch) {
                        GameManager.Instance.FirstLaunch = false;
                } else {
    #if  USE_ADMOB || USE_ADCOLONY || USE_UNITYADS
                        if (GDK_Settings.IsBannerEnabled())
                        {

        Advertisements.Instance.ShowBanner(GDK_Settings.GetBannerPosition(),GDK_Settings.GetBannerType());
                        }

                        if (ShowInterstitial)
                        {
                                Advertisements.Instance.ShowInterstitial();
                        }

        #endif
                }
        }

        void InitializeScene () {
                FadeDuration = GDK_Settings.GetFadeDuration ();
                SettingsScreen.SetActive (false);
```

```
                ExitScreen.SetActive (false);
        }
        public void SoundOnOff()
        {
                if (GameManager.Instance.SoundOn)
                {
                        GameManager.Instance.SoundOn = false;
                        SoundOff();
                }
                else
                {
                        SoundOn();
                        GameManager.Instance.SoundOn = true;
                }
        }
        public void VibrationOnOff()
        {
                if (GameManager.Instance.VibrationOn)
                        GameManager.Instance.VibrationOn = false;
                else
                        GameManager.Instance.VibrationOn = true;
        }
public void SetupEntryFee(int i)
        {
                                if (SaveData.Instance.UserCoins >= i)
                {
                        CurrencySound();
                        GameManager.Instance.EntryFee = i;
                        StartGame();
                }
                else
                {
                        OpenErrorPanel();
                        ErrorOccurSound();
                }
        }
```

```
        public void SetupPrize(int j)
        {
                GameManager.Instance.PrizeAmount = j;

        }
        public void OpenProfileScreen()
{
   ButtonPressed();
   ProfileDataScreen.DOAnchorPos(Vector2.zero, 0.5f, true);
   fadeScreen.SetActive(true);
}
public void closeProfileScren()
{
   ButtonPressed();
   ProfileDataScreen.DOAnchorPos(Vector2.right * -1997f, 0.5f, true);
   fadeScreen.SetActive(false);
}
        public void OpenChangeNamePanel()
        {
                ButtonPressed();
                ChangeNamePanel.gameObject.SetActive(true);
                ChangeNamePanel.GetComponentInChildren<InputField>().text = SaveData.Instance.userName;
        }
        public void OpenErrorPanel()
        {
                //ButtonPressed();
                Debug.Log("Error for no money");
                ErrorPanelForNoCoins.gameObject.SetActive(true);
        }
        public void CloseErrorPanel()
        {
                ButtonPressed();
                ErrorPanelForNoCoins.GetComponentInChildren<Animator>().SetTrigger("close");
                StartCoroutine(actionforcloseError());
        }
        IEnumerator actionforcloseError()
```

```
        {
                yield return new WaitForSeconds(0.4f);
                ErrorPanelForNoCoins.gameObject.SetActive(false);
        }
        public void SaveChangeNamePanelandClose()
        {
                ButtonPressed();
                SaveData.Instance.userName = ChangeNamePanel.GetComponentInChildren<InputField>().text;
                SaveLoadManager.SaveGame();
                SaveLoadManager.LoadGame();
                ChangeNamePanel.GetComponentInChildren<Animator>().SetTrigger("close");
                StartCoroutine(CloseNameChangeAction());
                SetupProfileData();
                //ChangeNamePanel.GetComponentInChildren<InputField>().text = SaveData.Instance.userName;
        }
        IEnumerator CloseNameChangeAction()
        {
                yield return new WaitForSeconds(0.5f);
                ChangeNamePanel.gameObject.SetActive(false);

        }
        //public int offlineOnline;
        public GameObject PrivateTableTitle, OnlineTableTitle;
        public void OpenPlayOfflinegameScreen(int i)
{
                  GameManager.Instance.online = i;
                if (i == 0)
                {
                        PrivateTableTitle.SetActive(true);
                        OnlineTableTitle.SetActive(false);
                }
                else
                {
                        PrivateTableTitle.SetActive(false);
                        OnlineTableTitle.SetActive(true);
                }
```

```
                ButtonPressed();
                PlayofflineScreen.DOAnchorPos(Vector2.zero, 0.5f, true);
                fadeScreen.SetActive(true);
            }
    public void closePlayOfflineScreen()
    {
      ButtonPressed();
      PlayofflineScreen.DOAnchorPos(Vector2.up *-1445f, 0.4f, true);
      fadeScreen.SetActive(false);
    }

    public void OpenFreeSpiner()
    {
      ButtonPressed();
      SpinnerFreeScreen.DOAnchorPos(Vector2.zero, 0.5f, true);
      fadeScreen.SetActive(true);
    }
    public void closeSSpinerScreen()
    {
      ButtonPressed();
      SpinnerFreeScreen.DOAnchorPos(Vector2.up * -1545f, 0.4f, true);
      fadeScreen.SetActive(false);
    }
    public void OpenGameSettings()
    {
      ButtonPressed();
      GameSettingScreen.DOAnchorPos(Vector2.zero, 0.6f, true);

      fadeScreen.SetActive(true);
    }

        public void closeGameSettings()
    {
      ButtonPressed();
      GameSettingScreen.DOAnchorPos(Vector2.right * 1645f, 0.4f, true);
      fadeScreen.SetActive(false);
```

```
}
    public void OpenCoinStoreScreen()
    {
        ButtonPressed();
        //CoinStoreScreen.DOAnchorPos(Vector2.zero, 0.6f, true);
        CoinStoreScreen.gameObject.SetActive(true);
        //fadeScreen.SetActive(true);
    }
    public void closeCoinStoreScreen()
    {
        ButtonPressed();
        CoinStoreScreen.GetComponentInChildren<Animator>().SetTrigger("close");
        StartCoroutine(closeCoinStoreScreenAction());

        //CoinStoreScreen.DOAnchorPos(Vector2.right * 1645f, 0.4f, true);

        //fadeScreen.SetActive(false);
    }
    IEnumerator closeCoinStoreScreenAction()
    {
        yield return new WaitForSeconds(0.5f);
        CoinStoreScreen.gameObject.SetActive(false);
    }
    public void OpenAchiementsScreen()
{
  ButtonPressed();
  AchiementsScreen.DOAnchorPos(Vector2.zero, 0.2f, true);
  fadeScreen.SetActive(true);
}
public void CloseAchiementsScreen()
{
  ButtonPressed();
  AchiementsScreen.DOAnchorPos(Vector2.up * -1445f, 0.2f, true);
  fadeScreen.SetActive(false);
}
public void OpeninfoScreen()
```

```
    {
        ButtonPressed();
        InfoScreen.DOAnchorPos(Vector2.zero, 0.4f, true);
        fadeScreen.SetActive(true);
    }
    public void closeInfoScreen()
    {
        ButtonPressed();
        InfoScreen.DOAnchorPos(Vector2.right * -2118f, 0.4f, true);
        fadeScreen.SetActive(false);
    }
   public void setPlayers(int i)
    {
        ButtonPressed();
        if (i == 0)
        {
            GameManager.Instance.Gameplayers = GamePlayers.three;
        }
        if (i == 1)
        {
            GameManager.Instance.Gameplayers = GamePlayers.Four;
        }
        if (i == 2)
        {
            GameManager.Instance.Gameplayers = GamePlayers.Five;

        }
                    if (i == 3)
                    {
                        GameManager.Instance.Gameplayers = GamePlayers.six;
}

            //      Debug.Log(GameManager.Instance.Gameplayers);
            }
    public void StartGame()
    {
```

```
                    //ButtonPressed();
                                if (GameManager.Instance.Gameplayers != GamePlayers.none)
                                {
                                        //  Debug.Log("Start the game");
                                        PlayofflineScreen.DOAnchorPos(Vector2.up * -1445f, 0.4f, true);
                                        fadeScreen.SetActive(false);

                            Fader.Instance.FadeIn(FadeDuration).LoadLevel("LoadingScreen").FadeOut(FadeDuration);
                    }
                                else
                                {

                                        ErrorOccurSound();                      }
}
                    public void PlayLevelsMode () {
                                if (!StartLoadNextScene) {
                                        ButtonPressed ();
                                        StartLoadNextScene = true;
                                        GameManager.Instance.SessionStartScene = CurrentScene;
                                        GameManager.Instance.CurrentGameMode = GameMode.LevelMode;
                                        LoadNextScene ();
                                }
                            }
        public void PlaySurvivalMode () {
                                if (!StartLoadNextScene) {
                                        ButtonPressed ();
                                        StartLoadNextScene = true;
                                        GameManager.Instance.SessionStartScene = CurrentScene;
                                        GameManager.Instance.CurrentGameMode = GameMode.SurvivalMode;
                                        LoadNextScene ();           }
                            }
                            public void ShareGame () {
                                #if USES_SHARING
                                string store = "";
                                #if UNITY_ANDROID
                                if (GDK_Settings.IsAmazonDevice ()) {
```

```
                    store = "Amazon AppStore";
                } else
                    store = "Google Play Store";
                #endif
                #if UNITY_IPHONE
                store = "App Store";
                #endif
                ShareSheet _shareSheet = new ShareSheet ();
                _shareSheet.Text = Application.productName + " - " + store + "\n" + ShareMessage;
                _shareSheet.URL = GDK_Settings.GetRateUsURL ();
                NPBinding.UI.SetPopoverPointAtLastTouchPosition ();
                NPBinding.Sharing.ShowView (_shareSheet, FinishedSharing);
                #endif
        }
private void FinishedSharing (eShareResult _result) {
                if (!_result.Equals (eShareResult.CLOSED)) {
                    NPBinding.UI.ShowAlertDialogWithSingleButton ("Share Successfull", "Thank you for
        sharing our game.", "Ok", null);
                }
        }
    public void OpenSettingsScreen () {
                ButtonPressed ();
                SettingsScreen.SetActive (true);
        }

        public void CloseSettingsScreen () {
                ButtonPressed ();
                SettingsScreen.SetActive (false);
        }

        public void OpenExitScreen () {
                ButtonPressed ();
                ExitScreen.SetActive (true);
        }

        public void CloseExitScreen () {
```

```
                ButtonPressed ();
                ExitScreen.SetActive (false);
        }

        public void ExitGame () {
                Application.Quit ();
        }

    public void PrivacyDialogue()
    {
                GDPRScreen.SetActive(true);
    }

        public void OpenPrivacyUrl()
        {
                }
public void SetUserConsent()
        {
                }

    public void OpenMoreGames () {
                ButtonPressed ();
                Application.OpenURL (GDK_Settings.GetMoreGamesURL ());
        }

        public void RateGame () {
                if (!GDK_Settings.IsAmazonDevice ()) {
                        NPBinding.Utility.RateMyApp.AskForReviewNow ();
                } else {
                        Application.OpenURL (GDK_Settings.GetRateUsURL ());
    }
        }

    //      public void LikeUs () {
    //              Application.OpenURL (GDK_Settings.GetFacebookURL ());
    //      }
```

```
public void RemoveAds () {
        ButtonPressed ();
        GDK_InAppController.Instance.BuyInAppProduct (0);
    }

    public void RestorePurchases () {
        ButtonPressed ();
        GDK_InAppController.Instance.RestorePurchases ();
    }

    public void ResetGame () {
        SaveData.Instance = null;
        SaveLoadManager.ResetSavedGame();
        SaveData.Instance = new SaveData ();
        SaveLoadManager.LoadGame();
    }

    void ButtonPressed () {
        if (GameManager.Instance.IsSFXControllerAvailable ())
            GDK_SoundController.Instance.PlaySingleSFX (SoundType.ButtonPressed);
    }
    void SoundOn()
    {
        if (GameManager.Instance.IsSFXControllerAvailable())
            GDK_SoundController.Instance.TurnOnSound();
    }
    void SoundOff()
    {
        if (GameManager.Instance.IsSFXControllerAvailable())
            GDK_SoundController.Instance.TurnOffSound();
    }
    void CurrencySound()
    {
        if (GameManager.Instance.IsSFXControllerAvailable())
            GDK_SoundController.Instance.PlaySingleSFX(SoundType.VirtualItemPurchased);
```

```
}
void ErrorOccurSound()
{
	if (GameManager.Instance.IsSFXControllerAvailable())
		GDK_SoundController.Instance.PlaySingleSFX(SoundType.errorSound);

	if(GameManager.Instance.VibrationOn)
		Handheld.Vibrate();

}
void LoadNextScene () {
	Scenes nextScene = NextScene;
	switch (GameManager.Instance.CurrentGameMode) {
	case GameMode.LevelMode:
		nextScene = NextScene;
		break;
	case GameMode.SurvivalMode:
		nextScene = SurvivalModeScene;
		break;
	}
	if (EnableLoadingScreen) {
		GameManager.Instance.NextSceneToLoad = nextScene;
		Fader.Instance.FadeIn (FadeDuration).LoadLevel (Scenes.LoadingScreen.ToString
()).FadeOut (FadeDuration);
	} else
		Fader.Instance.FadeIn (FadeDuration).LoadLevel (nextScene.ToString ()).FadeOut
(FadeDuration);
	}
}
```

### 5.2.6.3 Player selection scripts:

```
using Sirenix.OdinInspector;
using UnityEngine;
using UnityEngine.UI;

public class GDK_PlayerSelection : MonoBehaviour {

	#region Base-Classes

	[System.Serializable]
```

```
public class Selection_UI {

	[BoxGroup("Player Selection UI")]
	public Text PlayerName;
	[BoxGroup("Player Selection UI")]
	public Text PlayerInfoText;
	[BoxGroup("Player Selection UI")]
	public GameObject PlayerInfoObject;
	[BoxGroup("Player Selection UI")]
	public Text CoinsAmount;
	[BoxGroup("Player Selection UI")]
	public Button PurchaseButton;
	[BoxGroup("Player Selection UI")]
	public Text PurchaseAmount;
	[BoxGroup("Player Selection UI")]
	public Image[] AtrributeFillBars;
	[BoxGroup("Player Selection UI")]
	public Button PlayLevel;
	[BoxGroup("Player Selection UI")]
	public Button NextPlayer;
	[BoxGroup("Player Selection UI")]
	public Button PreviousPlayer;
}

[System.Serializable]
public class PlayerAttributes {

	public string PlayerName;

	public GameObject PlayerObject;
	[Tooltip ("Text to display when this Player is locked.")]
	[Multiline]
	public string PlayerInfo;
	[Range (0, 100)]

	public int[] Attributes;

	public ItemPurchaseTypes PurchaseType;

	[Tooltip ("Price of player in coins.")]
	[ShowIf("PurchaseType", ItemPurchaseTypes.VirtualItem)]
	public int VirtualItemPrice;

	[Tooltip ("Price of player in real currency.")]
	[ShowIf("PurchaseType", ItemPurchaseTypes.CurrencyItem)]
	public float CurrencyItemPrice;

	[Tooltip ("Item index of player in-app in NPSettings.")]
	[ShowIf("PurchaseType", ItemPurchaseTypes.CurrencyItem)]
	public int CurrencyItemIndex;
	[ShowIf("PurchaseType", ItemPurchaseTypes.LevelBased)]
	public int LevelRequired;

	public string GetPlayerName () {
		return this.PlayerName;
	}
```

```
	public bool LevelBasedUnlock () {
		return this.PurchaseType.Equals (ItemPurchaseTypes.LevelBased);
	}

	public bool VirtualItemBasedUnlock () {
		return this.PurchaseType.Equals (ItemPurchaseTypes.VirtualItem);
	}

	public bool CurrencyBasedUnlock () {
		return this.PurchaseType.Equals (ItemPurchaseTypes.CurrencyItem);
	}
}

#endregion


[BoxGroup("Player Selection UI")]
[SerializeField]Scenes PreviousScene;
[BoxGroup("Player Selection UI")]
[SerializeField]Scenes NextScene;
[BoxGroup("Player Selection UI")]
[SerializeField]bool EnableLoadingScreen;

[BoxGroup("Player Selection UI")]
[SerializeField]Selection_UI SelectionUI;

[BoxGroup("Player Selection UI")]
[SerializeField]PlayerAttributes[] Players;

bool LockPlayers;

int SequenceID;
float FadeDuration = 0.5f;
int currentPlayer;
bool StartLoadNextScene = false;
Scenes CurrentScene;

void Awake () {
	GameManager.Instance.InitializeGame ();
	CurrentScene = Scenes.PlayerSelection;
}

void Start () {

	InitializeScene ();
	InitializePlayers ();
	GetPlayerInfo ();

	if (GameManager.Instance.IsSFXControllerAvailable ())
		GDK_SoundController.Instance.PlayMusicLoop (SoundType.MenuMusic);


}
```

```
	void InitializePlayers () {
		GDK_Settings.SetMaxPlayers(Players.Length);
		if (SaveData.Instance.isPlayerUnlocked.Count == 0) {
			for (int i = 0; i < Players.Length; i++) {
				if (i == 0)
					SaveData.Instance.isPlayerUnlocked.Add (true);
				else
					SaveData.Instance.isPlayerUnlocked.Add (false);
			}
		} else if (SaveData.Instance.isPlayerUnlocked.Count < Players.Length) {
			for (int i = 0; i < Players.Length; i++) {
				if (i > SaveData.Instance.isPlayerUnlocked.Count - 1)
					SaveData.Instance.isPlayerUnlocked.Add (false);
			}
		}
		SaveLoadManager.SaveGame();

	}

	void InitializeScene () {
		UpdateCoinsAmount ();
		FadeDuration = GDK_Settings.GetFadeDuration ();
		LockPlayers = GDK_Settings.GetPlayerLockStatus();
	}

	void GetPlayerInfo () {
		for (int i = 0; i < Players.Length; i++) {
			if (i == currentPlayer) {
				Players [i].PlayerObject.SetActive (true);
				SelectionUI.PlayerName.text = Players [i].PlayerName;
			} else if (i != currentPlayer) {
				Players [i].PlayerObject.SetActive (false);
			}
		}
		if (SelectionUI.AtrributeFillBars.Length != Players [currentPlayer].Attributes.Length)
{
			Debug.LogError ("Quanitity of UI attribute fillbars and Player # " +
(currentPlayer + 1) + " atrributes do not match !");
		} else {
			for (int i = 0; i < SelectionUI.AtrributeFillBars.Length; i++) {
				if (i <= Players [currentPlayer].Attributes.Length) {
					SelectionUI.AtrributeFillBars [i].fillAmount = Players
[currentPlayer].Attributes [i] / 100.0f;
				}
			}
		}

		if (!LockPlayers) {
			SelectionUI.PlayerInfoObject.SetActive (false);
			SelectionUI.PurchaseButton.gameObject.SetActive (false);
			SelectionUI.PlayLevel.interactable = true;
		} else if (LockPlayers) {
			if (IsPlayerUnlocked ()) {
				SelectionUI.PlayerInfoObject.SetActive (false);
				SelectionUI.PurchaseButton.gameObject.SetActive (false);
				SelectionUI.PlayLevel.interactable = true;
```

```
                } else if (!IsPlayerUnlocked ()) {
                        SelectionUI.PlayerInfoObject.SetActive (true);
                        SelectionUI.PlayerInfoText.text = Players [currentPlayer].PlayerInfo;
                        SelectionUI.PlayLevel.interactable = false;
                        switch (Players [currentPlayer].PurchaseType) {
                        case ItemPurchaseTypes.LevelBased:
                                SelectionUI.PurchaseButton.gameObject.SetActive (false);
                                break;
                        case ItemPurchaseTypes.VirtualItem:
                                SelectionUI.PurchaseButton.gameObject.SetActive (true);
                                SelectionUI.PurchaseAmount.text = Players
[currentPlayer].VirtualItemPrice.ToString ();
                                break;
                        case ItemPurchaseTypes.CurrencyItem:
                                SelectionUI.PurchaseButton.gameObject.SetActive (true);
                                SelectionUI.PurchaseAmount.text = Players
[currentPlayer].CurrencyItemPrice.ToString ();
                                break;
                        }
                }
        }
        SetSelectionButtonsState ();
    }

    bool IsPlayerUnlocked () {
        switch (Players [currentPlayer].PurchaseType) {
        case ItemPurchaseTypes.None:
            return true;
        case ItemPurchaseTypes.LevelBased:
            if (SaveData.Instance.LevelUnlocked >= Players [currentPlayer].LevelRequired)
                return true;
            else
                return false;
        case ItemPurchaseTypes.VirtualItem:
        case ItemPurchaseTypes.CurrencyItem:
            return SaveData.Instance.isPlayerUnlocked [currentPlayer];
        default:
            return true;
        }
    }

    void SetSelectionButtonsState () {
        if (currentPlayer == 0) {
            SelectionUI.PreviousPlayer.interactable = false;
            SelectionUI.NextPlayer.interactable = true;
        } else if (currentPlayer == Players.Length - 1) {
            SelectionUI.PreviousPlayer.interactable = true;
            SelectionUI.NextPlayer.interactable = false;
        } else {
            SelectionUI.PreviousPlayer.interactable = true;
            SelectionUI.NextPlayer.interactable = true;
        }
    }

    public void PurchasePlayer () {
        switch (Players [currentPlayer].PurchaseType) {
```

```
			case ItemPurchaseTypes.VirtualItem:
				if (SaveData.Instance.UserCoins >= Players [currentPlayer].VirtualItemPrice) {
					SaveData.Instance.UserCoins -= Players [currentPlayer].VirtualItemPrice;
					SaveData.Instance.isPlayerUnlocked [currentPlayer] = true;
					if (GameManager.Instance.IsSFXControllerAvailable ())
						GDK_SoundController.Instance.PlaySingleSFX
(SoundType.VirtualItemPurchased);
					NPBinding.UI.ShowAlertDialogWithSingleButton ("Player Unlocked", " You
have unlocked " + Players [currentPlayer].PlayerName + ".", "Ok", null);
				} else {
					NPBinding.UI.ShowAlertDialogWithSingleButton ("Purchase Failed", " You
need " + Players [currentPlayer].VirtualItemPrice.ToString () + " coins to buy this item.", "Ok",
null);
				}
				break;
			case ItemPurchaseTypes.CurrencyItem:
				GDK_InAppController.Instance.BuyInAppProduct (Players
[currentPlayer].CurrencyItemIndex);
				break;
			}
			SaveLoadManager.SaveGame();
			UpdateCoinsAmount ();
			GetPlayerInfo ();
		}

		public void PlayerPurchased () {
			SaveData.Instance.isPlayerUnlocked [currentPlayer] = true;
			SaveLoadManager.SaveGame();
			GetPlayerInfo ();
		}

		void UpdateCoinsAmount () {
			SelectionUI.CoinsAmount.text = SaveData.Instance.UserCoins.ToString ();
		}

		public void SelectPrevious () {
			ButtonPressed ();
			currentPlayer--;
			GetPlayerInfo ();
		}

		public void SelectNext () {
			ButtonPressed ();
			currentPlayer++;
			GetPlayerInfo ();
		}

		public void NextButton () {
			if (!StartLoadNextScene) {
				ButtonPressed ();
				StartLoadNextScene = true;
				GameManager.Instance.CurrentPlayer = currentPlayer + 1;
				GameManager.Instance.SessionStartScene = CurrentScene;
				if (EnableLoadingScreen) {
					GameManager.Instance.NextSceneToLoad = NextScene;
```

```
                        Fader.Instance.FadeIn (FadeDuration).LoadLevel
(Scenes.LoadingScreen.ToString ()).FadeOut (FadeDuration);
                    } else
                        Fader.Instance.FadeIn (FadeDuration).LoadLevel (NextScene.ToString
()).FadeOut (FadeDuration);
                }
        }

        void ButtonPressed () {
                if (GameManager.Instance.IsSFXControllerAvailable ())
                        GDK_SoundController.Instance.PlaySingleSFX (SoundType.ButtonPressed);
        }

        public void BackButton () {
                if (!StartLoadNextScene) {
                        ButtonPressed ();
                        StartLoadNextScene = true;
                        LoadPreviousScene ();
                }
        }

        void LoadPreviousScene () {
         if (GameManager.Instance.CurrentGameMode.Equals(GameMode.LevelMode)) {
             Fader.Instance.FadeIn (FadeDuration).LoadLevel (PreviousScene.ToString ()).FadeOut
(FadeDuration);
         }
         else
         {
             Fader.Instance.FadeIn (FadeDuration).LoadLevel (Scenes.MainMenu.ToString()).FadeOut
(FadeDuration);
         }

        }
}
```

## 5.3 **Deployment Environment**

We can deploy our game in any environment but firstly we deploy it computer systems.User play game on their PC's. After that we can potentially deploy our games to a greater variety of devices from tablets to android device.

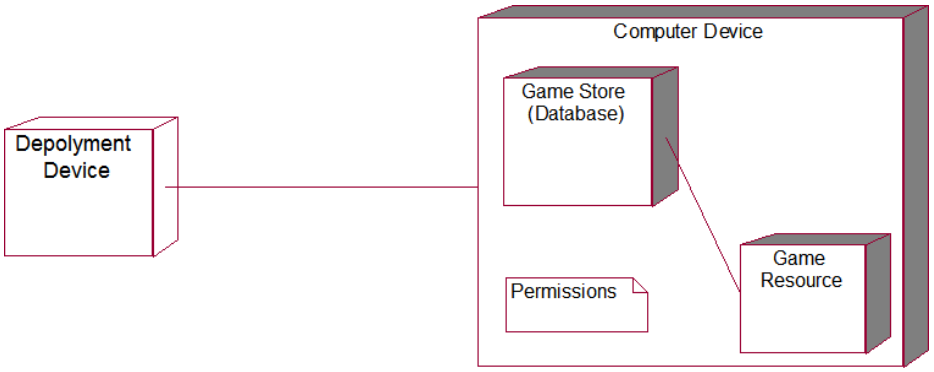



**Figure 14: Deployment Environment Diagram**

## 5.4. Tools and Techniques

**Product:** Unity Technologies

**Tool:** Unity 3d

**Usage:** Game Engine

**Work Experiment:** Backend and frontend Activity

**Product:** Adobe

**Tool:** Adobe photoshop

**Usage:** Create broachers and templates

**Work Experiment:** To create broachers, templates and flayers for repor

**Product:** Microsoft Corporation

**Tool:** Microsoft visual studio

**Usage:** Game coding

**Work Experiment:** game coding in c#

## 5.5. Best Practices / Coding Standards

**Naming Conventions**

The naming conventions in unity we used

**Variables:** begins with the lowercase letters. The variables are used to store the game information about any aspects of any state of game.

**Classes:** Classes are declared with the uppercase letter. The classes are the collection of functions.

**Functions:** Functions are starts with uppercase letters. Functions are collection of code and they are written once and they can be used as they need in program.

**Save:** The scene by selecting File->Save As and give the game a name

**Indentation**

¬ Emphasize the body of a control statement such a loop or a select statement
¬ Define the body of a conditional statement
¬ Define a new scope block
¬ Three spaces minimum used to indent

**Inline Comments:** Inline comments explaining the functionalities and codes of the game development

**Use of Braces**
¬ Braces are used to delimit the bodies of conditional statements , control constructs, and blocks of scope.

¬ Braces shall be used even when there is only one statement in the control block.

**Compiler Warnings**

¬ Compiler and linker warnings shall be treated as errors and fixed because they often indicate problems which may affect the behavior, reliability and portability of the code.

¬ The options fully enable the compiler's code-checking features.

**Line Length:** Line lengths cannot exceed 72 columns.

**Spacing:**

¬ A keyword followed by a parenthesis should be separated by a space.
¬ A blank space should be appear after each comma in an argument list.
¬ Casts should be made followed by a blank space.
**Wrapping Lines:**
¬ Break after a comma
¬ Break after an operator
¬ Prefer higher level breaks to lower level breaks

**Variable Declarations:** Variable declarations should always be preceded by multiple lines.

1. Coding for efficiency
2. Coding for readability
3. Meaningful error messages
4. Reasonably sized functions and methods

## 5.6. Version Control

Unity provides us services to build our game as a team in a different ways but rather than using this we individually complete our tasks in our separate machines and the finally joined all this and build game in a one single machine in which our complete project is up and running. For further any update we fix this issue in the same way we did before.

# Chapter 6

# Testing and Evaluation

# **Chapter 6:** Testing and Evaluation

This chapter includes some test cases for the game to check if the game works properly in various situations. We are giving four test examples for eight different situations here. In this whole chapter we check each aspect of game according to its performance and appearance. Testing each part of the game part give us a detailed report of our game which gives us a better understanding of game and its life cycle.

## 6.1. Use Case Testing

In use case testing we generally test our game covering all aspects.

| | |
|---|---|
| User Interface(UI) | Test if there is no overlapping of objects |
| | TEST for animation, movement of cards, graphics, Zoom In/Out (all gestures) etc. |
| | There should not be any clipping (cut background) |
| | Cards should not move out of the screen/specified area |
| | Test for enable and disable images/icons/buttons etc |
| | Font displayed (color, size etc) |

| | |
|---|---|
| Performance during game | TEST the loading time of a game |
| Core structure of game | TEST game area, game logic |
| | Lobby options |
| | Test when player 2's turn is on Player 1 is not able to do actions (should not be able to forfeit also) |
| Memory leak | Check the game when device memory is low |
| Check for localization | Should be Support of different languages |

## 6.2. Equivalence partitioning

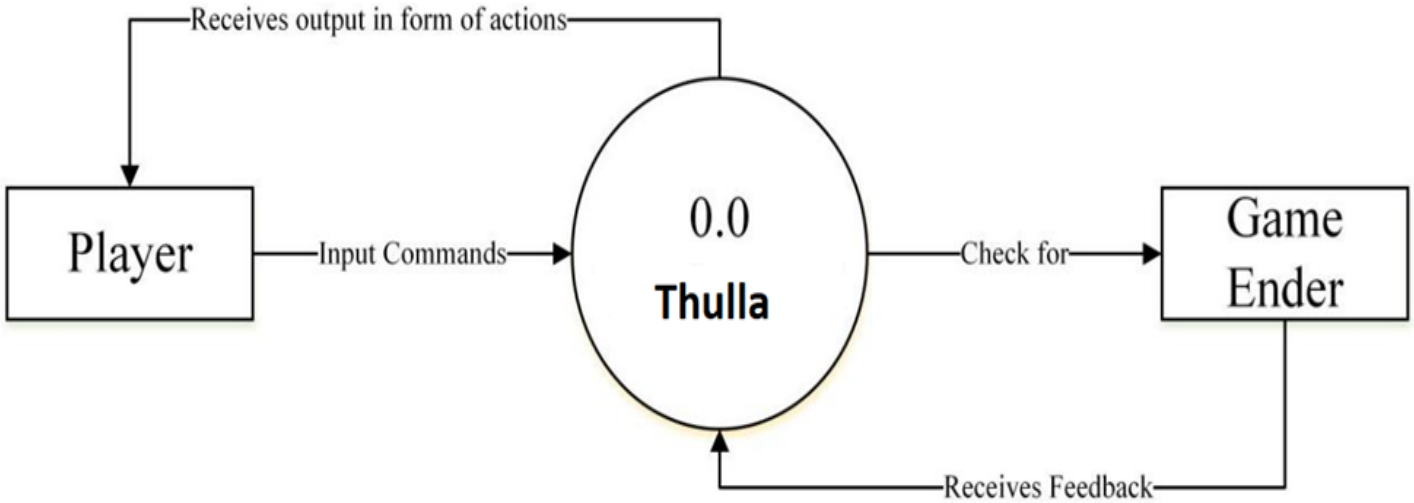
| **Invalid** | **Valid** | **Invalid** |
|---|---|---|
| Player's Card < 1 | Card in between 1-26 | Player's Card > 26 |
| Move Card < 1 | Move Card = 1 | Move Card >1 |
| Total Cards < 52 | Total Cards = 50 | Total Cards < 52 |

| | | |
|---|---|---|
| Chaal Time > 5.0 | Chaal Time = 5.0 | Chaal Time < 5.0 |
| Spawn Points in level > 2 | Spawn Points in level = 2 | Spawn Points in level < 2 |
| | | |

## 6.3. Boundary value analysis

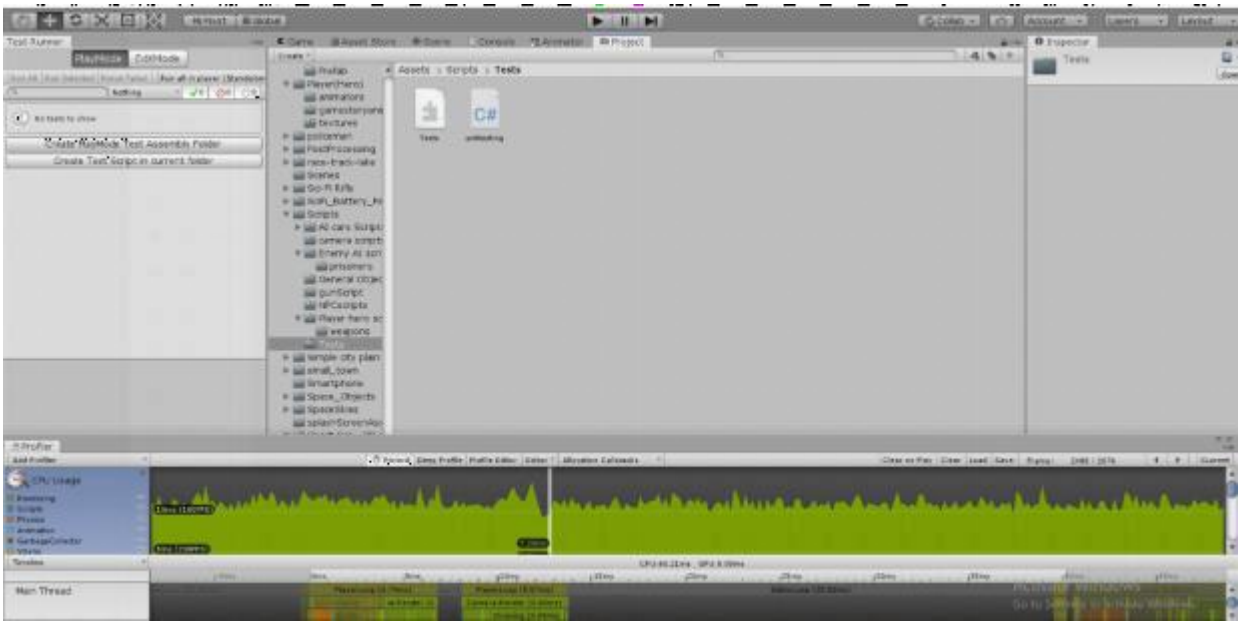
| | | |
|---|---|---|
| Invalid (min-1) | Valid (min, +min, -max, max) | Invalid (max+1) |
| Player's Card (-1) | Card in between (1-26) | Player's Card (27) |
| Move Card (2-99) | Move Card = 1 (1) | Move Card (2) |
| Total Card (-51) | Shot Speed (52) | Shot Speed (53) |
| Chaal Time (4.9s) | Chaal Time (5.0s) | Chaal Time(5.1s) |
| Spawn Points in level (1) | Spawn Points in level (2) | Spawn Points in level (3) |

## 6.4. Data flow testing




## 6.5. Unit testing

With the help of unity testing you can avoid from stupid bugs and write a sustainable code for your game. By using unit testing you get the early feedback through series of tests and avoid any type of regression in game and en-rooted to object oriented design and create during documentation of your code.



Unity itself provides the features of unit testing of game in its tool. So we ran our script tests for the betterment and find there is no error to be fixed in this code all the scripts to be tested and unity made them clear.

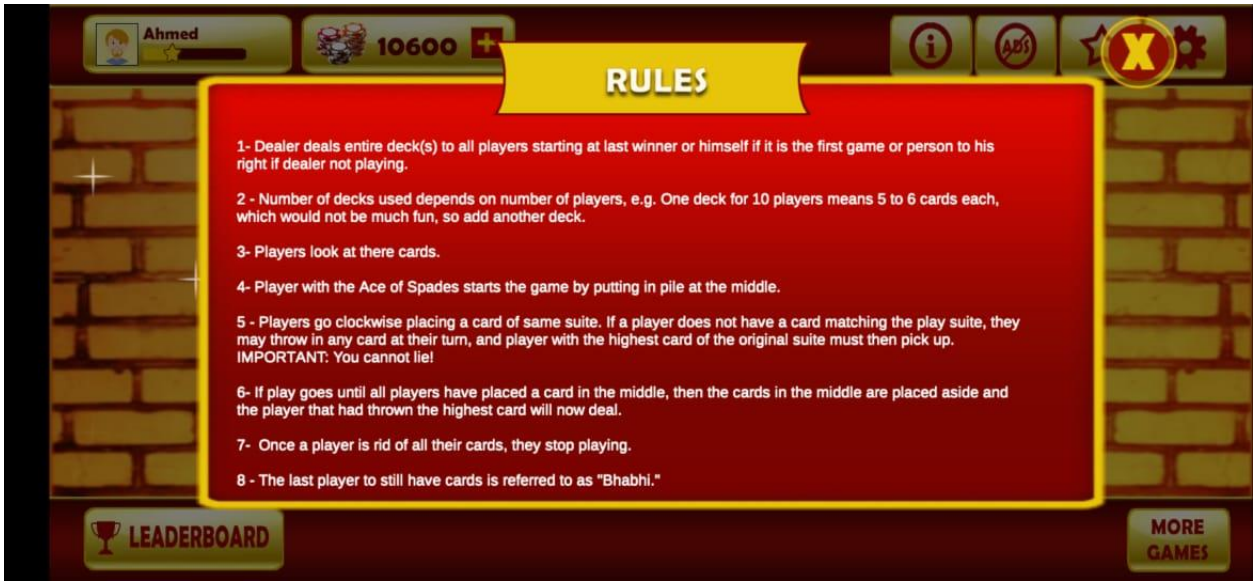
## 6.6. Integration testing

Our game consists of following scenes we test all the scenes step by step.

Mode Section Gives Two Types of Game Play Modes.


There are 2 modes of play user can play in both modes in private mode user didn’t need any internet connection in online mood user required internet connection.


In private table user can select number of players and amount of bet itself, the player who finish on top will win double of its bet amount and the player who finish on last will lost its bet amount.



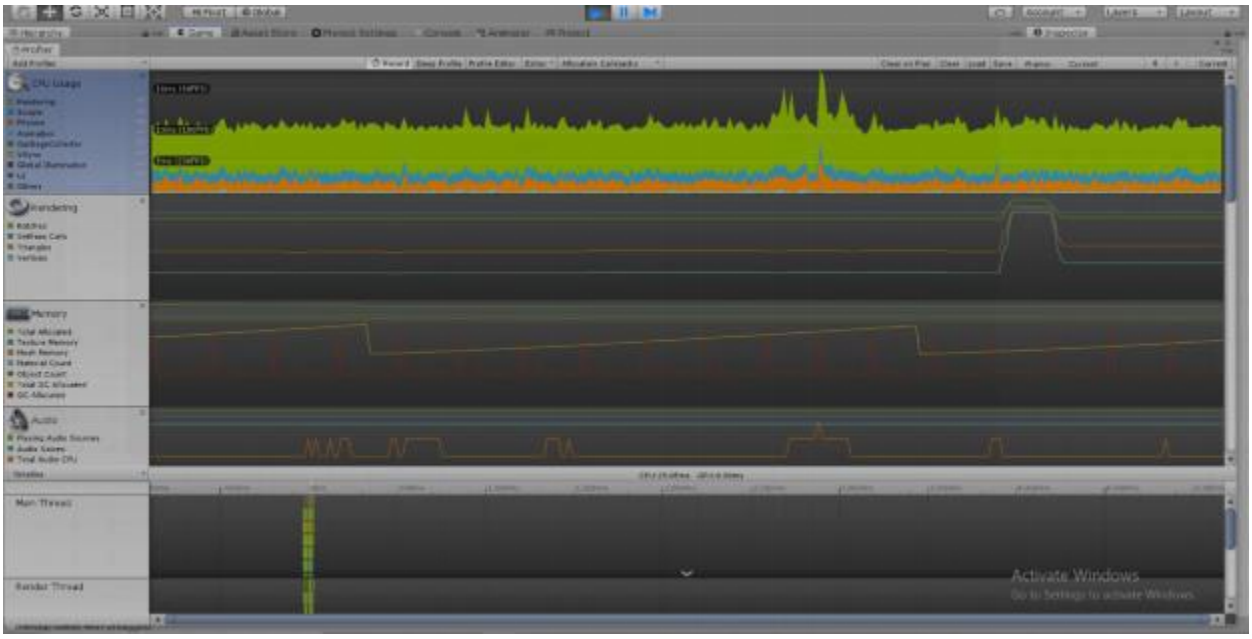
User have to aware by rules of game, every player have equal numbers of cards. Player with the ace of spades starts the game by putting in pie at the middle. The last player to still have cards is referred as Bhabhi.


User can turn on/off sound and vibration of game by going into settings mode.

User can change it's username by going into profile mode and can also check total game play, game won and success rate.

## 6.7. Performance testing

For testing the performance of our game unity itself provide a feature to test the performance of your game using PROFILER. It can test the gpu and cpu requirement of your game and displays the result right away when you play it in the unity. We can test the CPU usage, rendering unit, memory consumption and many other related things.



## 6.8. Stress Testing

In this testing we test the strength of a software by pushing to its limits so that we can came to know about how much this software goes far.

# Chapter 7

# Summary, Conclusion and Future Enhancements

# **Chapter 7:** Summary, Conclusion & Future Enhancements

## 7.1. Project Summary

The Summary of this project is to implement a First Human Vs Human and Machine Vs Machine Thulla Game. We are three group members. We haven't read the Unity 2d course in our degree but we are fond of developing game that's why we choose this project for making this game we use a unity 2d tool and visual studio for scripting. In this game we have different mode which include up to 4 players mode in private/online Table mode exist which include game mode i-e Easy, Medium, Hard. If you win the game you will receive coins as rewards by which you can play a higher bet game. User will be found it more interesting as well he will achieve levels stepwise and in future, we will try to achieve more exceptional functionalities in our game in update version (login with social media etc...).

## 7.2. Achievements and Improvements

During this process of game development we learnt a lot of skills like :-

- Unity Engine Understanding
- Working with a 3D objects
- Creating an Environments
- Best Coding practices
- Physics
- Dimensions

- C# scripting
- Object oriented coding
- Building and running Mobile games in various environments
- Exposure in a gaming field
- Logic behind gaming
- UI/UX designing
- Working with computer vision in a Unity 3D environment

these are the achievements and improvements we think we learn so far from this final project and that takes our interest in gaming to take pursue our future with this field.

## 7.3. Critical Review

Chess (Thulla) is first of it's kind that provide the single player using Mobile vision feature so its takes time to be completely stable. In sense of its unique controls, player should have hardware according to above mentioned requirements otherwise players will be face some disturbance.

## 7.4. Lessons Learnt

We absorb very much from this project. This project sharpens our skills in Unity Game Engine, designing tools and many management concepts as well as how to deal with a problem and how to stick for finding the solution of any problem until you fond. As well as technical skills this project also enhance our personal development skills such as team working, dedication. We learnt various types of techniques in gaming development for the creation of game. Our passion of making a unique game that contains the enjoyment element in an innovative environment and we succeeded it. We continue our work after this to make it through worldwide. This project is a

stepping stone for us in the beginning we don't even know that if we accomplished it or not but due to our hardworking and research we do it and it's an achievement for us that motivates us to do better and more.

## 7.5. Future Enhancements/Recommendations

So generally, there are many enhancements we want to put in our game i.e.

- Enhance the visual of game
- Adding various tables in game
- Adding a double bet mode
- Increase the performance of game
- Adding different inventories
- Adding more crates
- Adding new features
- Create a competition between players.

# Appendices

# **Appendix A:** User Manual

In the appendix section we describe the different phases of user interface and also describe how user can use our project. Secondly we describe our promotional plan and promotional materials like broucher, banner, standee and other marketing materials. We also describe other interfaces of our game project.

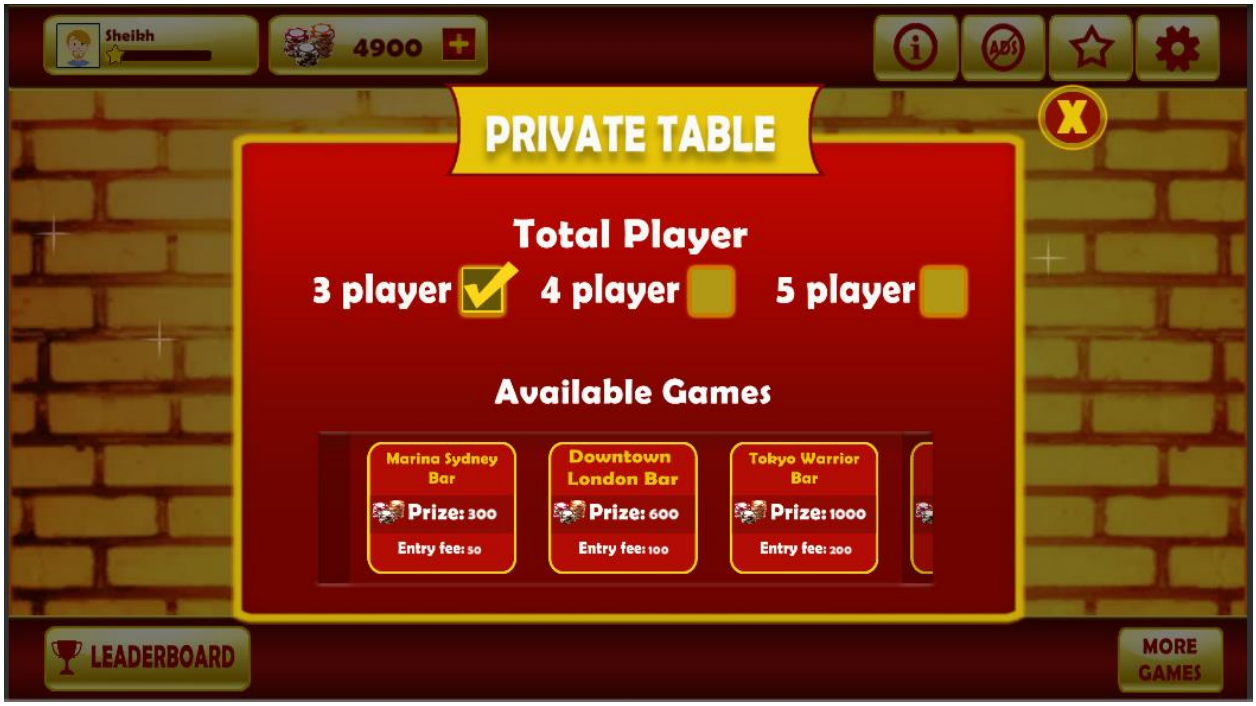
# **Appendix A:** MAIN

## Splash Screen:

This is the first page user see when ever he start the game.

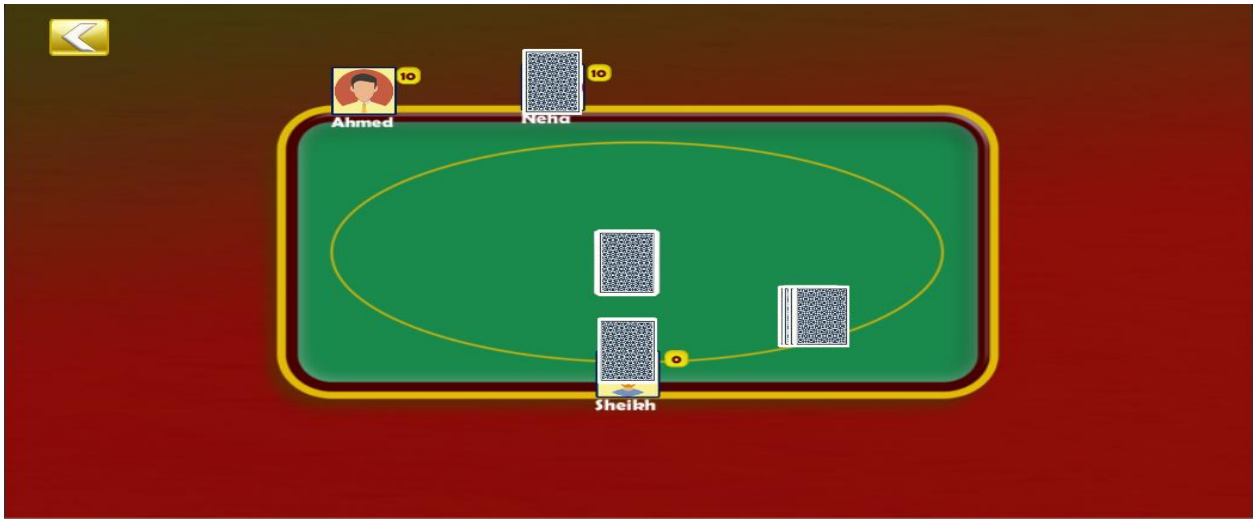
### A.1.1. Table Creation

In this screen the user select number of players and bet amount to create a personalized table.



#### A.1.1.1. Main Game

In this screen game allow a user to play a game with computer




# Appendix B: Administrator Manual

There is no Administrator manual for this game

## B.1. First Level heading

N\A

### B.1.1. Second level heading

N\A

#### B.1.1.1. Third level heading

N\A

# **Appendix C:** Information / Promotional Material

## C.1. Broacher

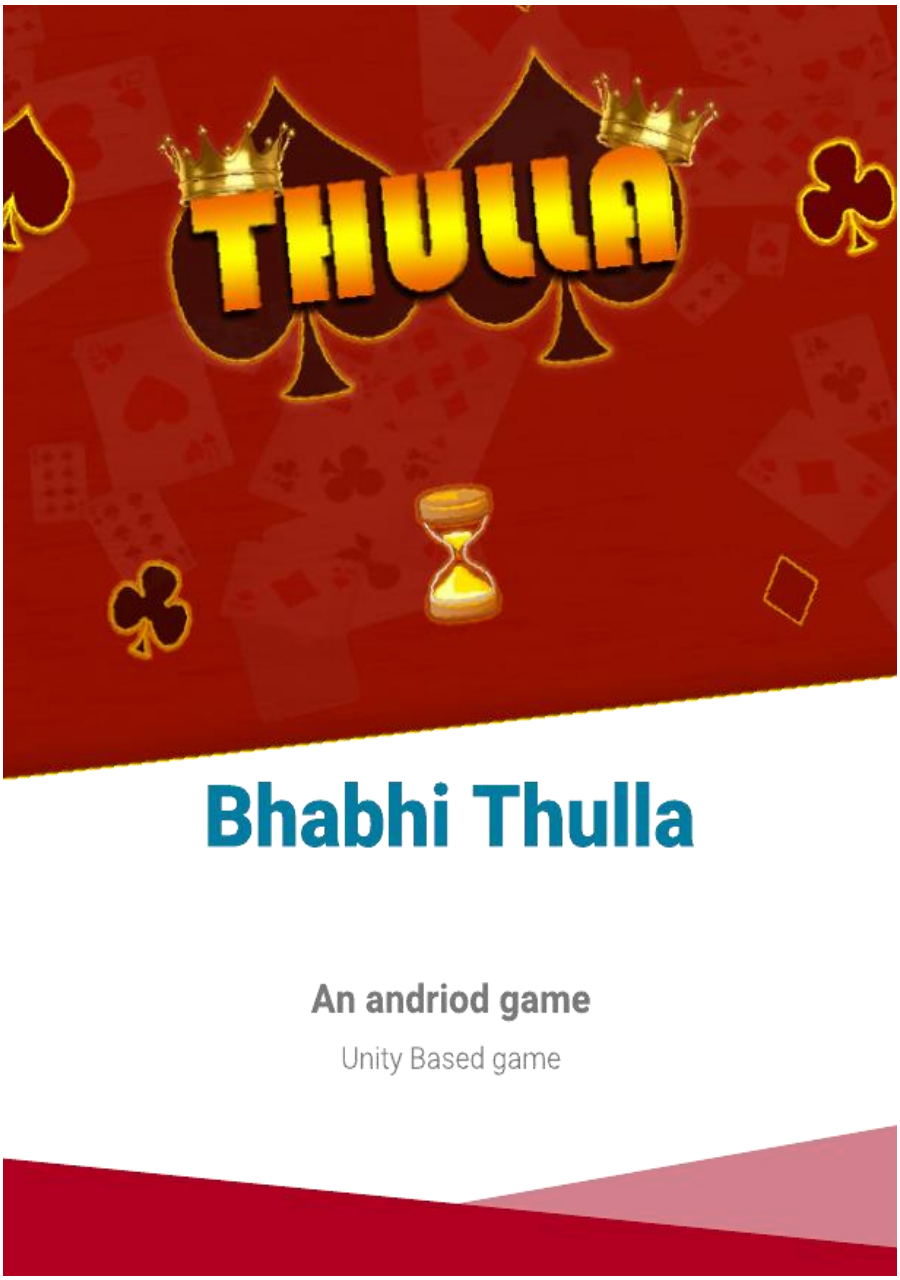
## C.2. Flyer


# Bhabhi Thulla

**An andriod game**

Unity Based game

## C.3. Standee



## C.4. Banner


Chintu
2.4 k
BUY COINS
Boot: 100
#1 Prize: 400
Jatin
1.8 k
Sanjay
2.5 k
User
49.9 k
02 : 13
NO INTERNET CONNECTION REQUIRED
PLAY OFFLINE

# Reference and Bibliography

# Reference and Bibliography


- IEEE Software Engineering Standards Committee, “IEEE Standard 830-2016, IEEE Recommended Practice for Software Requirements Specifications”, October 20, 2016.
- Project report on game development. [online] Academia.edu. Available at:https://www.academia.edu/13719598/Project_report_on_game_development [Accessed on 1 Jan. 2019].
- http://paperplanetools.com. (2018). Assets for Unity by Paper Plane Tools. [online]

# Index

# Index

**[C]**



**[D]**



**[E]**

Thulla