Village Survival Action RPG

1.1.1 Project Code

<Project code assigned by the Project Office>

1.1.2 Project Advisor

<Prof.Saad Razzaq> (internal)
<Mam Maham> (external)

1.1.3 Project Manager

<Prof.Saad Razaq>

1.1.4 Project Team

BSCSF18M020 Iqra Mushtaq
BSCSF18M026 Alia Saleem
BSCSF18M032 Rukhma Khalid

1.1.5 Submission Date

<12/1/2022>

Table of Contents

1. *Abstract*

*Gaming is one of the most important fields of programming. Especially in this era in which most of the population from children to mature people are playing games. Games of different genres like adventure, puzzle, strategy, like the walking dead, clash of clans, candy crush etc. In our daily life, we see and play different games. In these games, we notice some problems i.e., lack of graphics, performance, ads, static environment and ambiguity in understanding the story of the game etc. The name of our proposed project is "**Village Survival Action RPG** '. This is a single player game with multi-platform game project i.e., android, window. Village Survival Action RPG is a third person game which will provide a complete story to help user/player to understand about levels. The **Goal** of the game is to protect the village, which is a game environment, containing farms, crops from different types of enemies like goblins, spiders etc. Those enemies attacked villages and farms to take control of the village. Their target will be destruction of farms and crops. It has **dynamic enemy's locations and AI based** enemies. In this game the player is being placed in the boundary of the village, the player kills his enemies with different weapons to save himself and saves his toughness. If the player loses his health, he will die. In this game the death of the enemy will be according to his health and we have multiple levels and the enemies vary at each level like 1st level goblins, 2nd level containing spiders and L3 have human-like enemies etc. The life of the enemy and the protectors depends on how they bear attacks. Each level has different themes like how we will survive at each level. This should be a good source of entertainment for all users. This will be a game with good graphics and an attractive interface.*

2. *Background and Justification*

We play a lot of games but we face a lot of difficulties while playing them. As the player does not know how to use the controls, he is not fully guided. And the controls are static, which eliminates interest in the game. The full story of the game is not shown so that the player does not understand the game and what its mission is.

1.1.6 Example

1. *Project Zarb*
2. *Battle against terrorists*
3. *Battle Rage*

Are the games which have static enemies and have fewer hurdles that make the game less interesting and also have the same environment in each level. Battle Rage has dynamic enemies but these enemies only attack when the player is in his range or radar and it has only one control which eliminates interest in-game.

Following are the common issues we had observed in the above-mentioned games. Graphics

- *Static Enemies*
- *Hurdles*
- *Fewer Controls*
- *Incomplete Story*
- *First Person*

3. Project Methodology

We are going to use the waterfall model. First, we gather requirements (Story, Models, Characters, and other objects) then move on to designing (Modeling of objects). After that, we head towards deployment (Implementation of story and integration of designed models) and then move towards testing (Finding error in our game). There are the following Hardware & Software requirements for this project given below:

1.1.7 Hardware Requirements for Development

Laptop or PC

Android Device
At least 1 GB rams.
Hard disk 16 GB.

1.1.8 Software Requirements for Development

***Operating System**: Windows 7 or the latest Version, Android 4.6 or higher*

***Unity 3D:** Unity3D is a powerful cross-platform 3D engine and a user-friendly development environment. Easy enough for the beginner and powerful enough for the expert.*

***Photoshop:** is a raster graphics editor developed and published by Adobe Inc. for Windows and Mac OS.*

***Visual Code:** we are using visual code for backend coding and scripting*

4. Project Scope

***Village Survival Action RPG** is a third person game which will provide a complete story to help user/player to understand about levels. It has better graphics and gaming experience then other games. Players can sharpen their mind by facing different hurdles every time while playing different missions. It has dynamic enemy's locations and AI based enemies. Players get entertained through a better graphical environment and through a good UI design.*

Here are the functional requirements of proposed game

Functional Requirements

- ***Game Start:***
  Users can start the game by pressing the play button.
- ***Game Reset:***
  User can reset the game score
- ***Game Quit/ Game Exit**:*
  Quit button is used to quit the game
- ***Sound/ Sound Controller**:*
  User can increase or decrease the volume of sound
- ***Help about Game Controls**:*
  User can get the help about the controls of game
- ***Blocking Attack***
  Player/Enemy can block the attack from another entity.

- ***Life Management:***

  Life management of players will go through different hurdles like the number of attacks hits the player.
- ***Level management:***

  Each level should be maintained in a conditional way that you can only play the next level only after completing the previous level. In short, the levels will be dependent on each other
- ***Enemies Location/Map:***

  There should be a map in the game that can specify all the locations of enemies.
- ***Player Success***

  Player can achieve success after destroying all enemies
- ***Player Death***

  The Player will die if its health is zero.
- ***Level Failure***

  The level will fail if the player dies before enemies.
- ***Enemy Death***

  Enemy will die if its health is zero.
- ***Score management***

  Score can be managed in our project through different conditions like the player completed the mission before half of the given time to complete the objectives.
- ***Weapons Management***

  Players can select a weapon from different weapons.
- ***Game Manager***

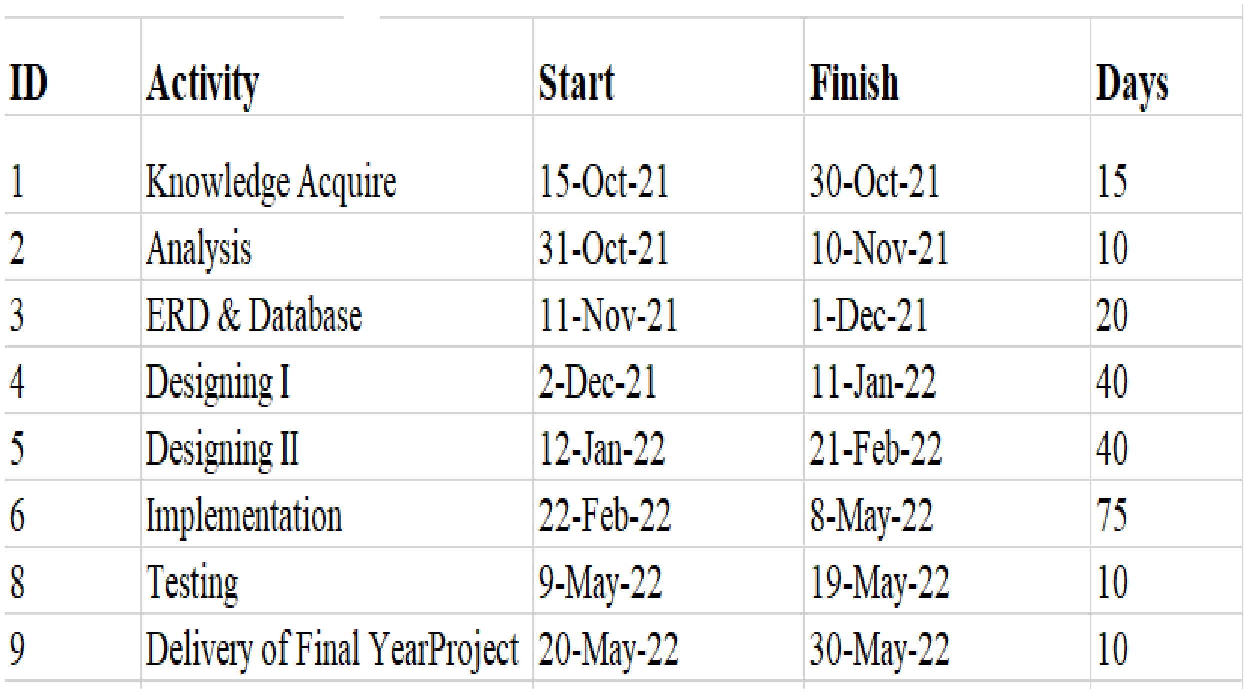
  It will manage game flow and manage all game Objects.
- ***MiniMap***

  Location of enemies will be displayed on the map.

Here are some limitations:

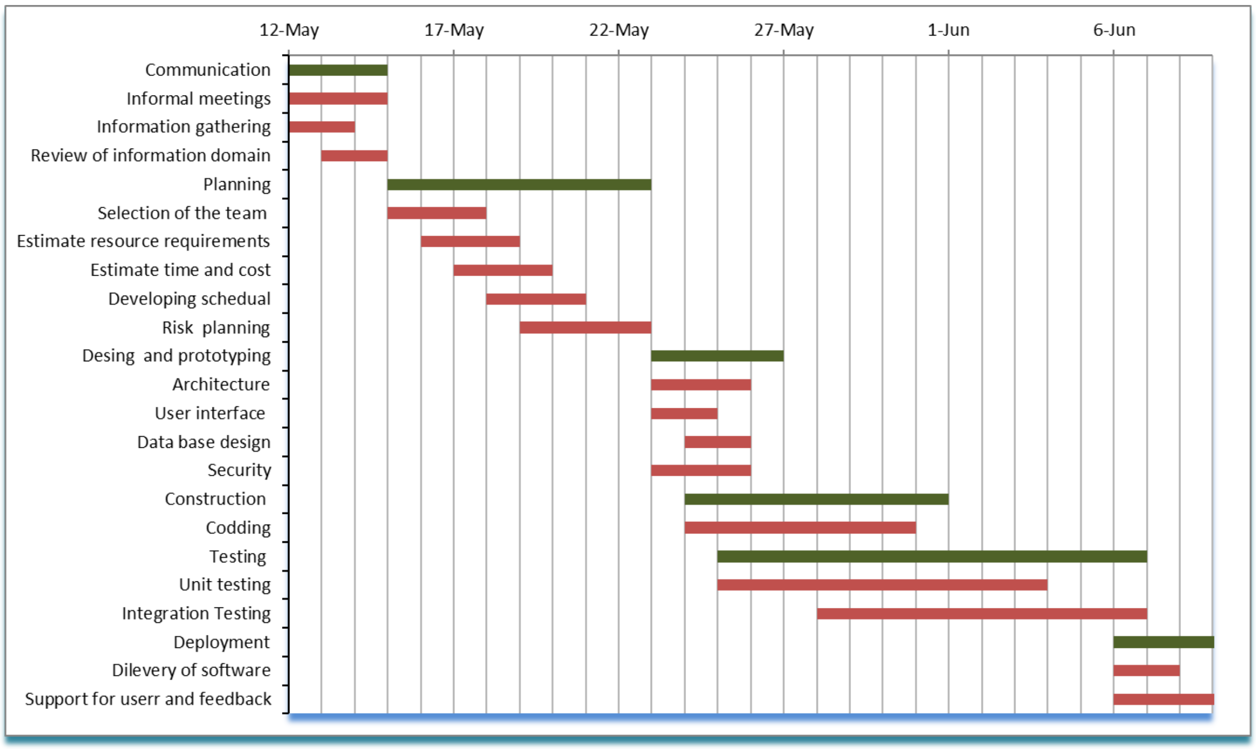
- *Game must be offline*
- *Android device must be required to run the game*
- *Android version must be greater than 4.6*
- *Player must complete the 1st mission to reach the next mission*

5. High level Project Plan:



ID	Activity	Start	Finish	Days
1	Knowledge Acquire	15-Oct-21	30-Oct-21	15
2	Analysis	31-Oct-21	10-Nov-21	10
3	ERD & Database	11-Nov-21	1-Dec-21	20
4	Designing I	2-Dec-21	11-Jan-22	40
5	Designing II	12-Jan-22	21-Feb-22	40
6	Implementation	22-Feb-22	8-May-22	75
8	Testing	9-May-22	19-May-22	10
9	Delivery of Final YearProject	20-May-22	30-May-22	10

Gantt chart


12-May
17-May
22-May
27-May
1-Jun
6-Jun
Communication
Informal meetings
Information gathering
Review of information domain
Planning
Selection of the team
Estimate resource requirements
Estimate time and cost
Developing schedual
Risk planning
Desing and prototyping
Architecture
User interface
Data base design
Security
Construction
Codding
Testing
Unit testing
Integration Testing
Deployment
Dilevery of software
Support for userr and feedback

Software Requirements Specifications

1.Introduction

1.1Purpose of Document

The purpose of this document is to present a detailed description of our final year project which is a game. This description relates to the functionality, constraints, performance, attribute and the game interface. e. The document is intended for all the stakeholder's customers and the developer. The audience that is going to play the game offline is assumed to have basic knowledge of accessing its controls.

.

1.2 Project Overview

*The name of our proposed project is "**Village Survival Action RPG** '. This is a single player game with multi-platform game project i.e., android, window. Village Survival Action RPG is a third person game which will provide a complete story to help user/player to understand about levels. The **Goal** of the game is to protect the village, which is a game environment, containing farms, crops from different types of enemies like goblins, spiders etc. Those enemies attacked villages and farms to take control of the village. Their target will be destruction of farms and crops. It has **dynamic enemy's locations and AI based** enemies. In this game the player is being placed in the boundary of the village, the player kills his enemies with different weapons to save himself and saves his toughness. If the player loses his health, he will die. In this game the death of the enemy will be according to his health and we have multiple levels and the enemies vary at each level like 1st level goblins, 2nd level containing spiders and L3 have human-like enemies etc. The life of the enemy and the protectors depends on how they bear attacks. Each level has different themes like how we will survive at each level. This should be a good source of entertainment for all users. This will be a game with good graphics and an attractive interface.*

1.3 Scope

***Village Survival Action RPG** is a third person game which will provide a complete story to help user/player to understand about levels. It has better graphics and gaming experience then other games. Players can sharpen their mind by facing different hurdles every time while playing different missions. It has dynamic enemy's locations and AI based enemies. Players get entertained through a better graphical environment and through a good UI design. Here are some limitations:*

- *Game must be offline*
- *Android device must be required to run the game*
- *Android version must be greater than 4.6*
- *Player must complete the 1st mission to reach the next mission*

2. Overall System Description

2.1User characteristics

- ***User Controls***

*Users can move at **360 angle** by using a joystick,by using **playercontroller class.***

- *Weapon Selection*

*Users can select weapon ,swapp the player by using **InApp_ purchase class.***

- ***Levels Selection***

*Users can select levels by using the **LevelsManagement class.***

- ***Attack***

The player can attack by using the ***Mlee_Generator class.***

- ***Score_ management***

Score panel will be handled by using ***Score Manager class***

- ***AIControllers***

All intelligence algorithms use in our game are handled by ***AIControllers class***

- ***Main_Menu***

User interface panel is managed by ***MainMenu class i.e*** *start, pause, quit,and resume the game*

2.2Operating environment

Operating environment is the combination of hardware and software, essential for project development. There are the following Hardware & Software requirements for this project given below:

Hardware Requirements for Development

- *Laptop or PC*
- *Android Device*
- *At least 1 GB ram.*
- *Hard disk 16 GB.*

Software Requirements for Development

- ***Operating System****: Windows 7 or the latest Version, Android 4.6 or higher*
- ***Unity 3D:*** *Unity3D is a powerful cross-platform 3D engine and a*
- *user-friendly development environment. Easy enough for the*
- *beginner and powerful enough for the expert.*
- ***Photoshop:*** *is a raster graphics editor developed and published by*
- *Adobe Inc. for Windows and Mac OS.*
- ***Visual Code:*** *we are using visual code for backend coding and scripting*

System constraints

- *Game must be offline*
- *Android device must be required to run the game*
- *Android version must be greater than 4.6*
- *Player must complete the 1st mission to reach the next mission*

3.External Interface Requirements

3.1 Software Interfaces

Software Name	***Description***
Operating System	*Android 4.6 or higher* *Windows 7 or higher* *Proposed Game will be run on these 2 operating systems*
Unity 3D	*This is a tool which will develop native games for all platforms.* *We will use this to develop our game.*
Photoshop	*This is a tool which will be used for making UI designs of our game.*
Android Studio	*This is a tool we use to build our game for android devices*
Local Storage/Cache	*We will use local storages of mobile/laptop devices to save our open levels, high scores and other stuff.*

4.Functional Requirements

Some functional requirements are as;

- ***StartGame();***

By using this function users start the game by pressing the play button. So, here, instead of starting automatically, we leave this option to the user.

- ***ResetGame();***

Here User can reset the game score; means setting the results to zero without starting the game again.

- ***Quit/ExitGame();***

By using this function users can easily leave the game, whenever they want.

- ***SoundController();***

Users can increase or decrease the volume of sound.

- ***HelpaboutGamecontrols();***

Users can get help with the controls of the game. Users can use keyboard arrows to move the player. Left and right button to move left and right, and move up button to jump, etc..

- ***BlockingAttack();***

Player/Enemy can block the attack from another entity to save themselves from any injuries, to maintain their health.

- ***LifeManagement();***

Life management of players will go through different hurdles like the number of attacks that hit the player.

- ***LevelsManagemen();***

Every level should be maintained conditionally. This means the player can only play the next level only after completing the previous one. In short, the levels will be dependent on each other.

- ***EnemiesLocation(); /Map();***

This provides a map in the game that can specify all the locations of enemies. To make players alert about their locations, in order to strongly face attacks.

- ***PlayerSuccess();***

The success of the player can be achieved after destroying all enemies.

- ***PlayerDeath();***

When the number of attacks on players from enemies is increasing continuously. Means the player can't defend itself, and at some point, his health will be zero. So, The player will die.

- ***LevelFailure();***

The level for a player will fail if the player dies before enemies.

- ***EnemyDeath();***

If the number of attacks on enemies from players is increasing continuously and they can not defend themselves. Then at some point its health will be zero. So, the Enemy will die.

- *.**ScoreManagement();***

The score can be handled through different conditions, like the player completing the mission before half of the given time.

- ***WeaponsManagement();***

Players can select a weapon of their choice from different weapons.

- ***GameManager();***

It will manage game flow and manage all game Objects.

- ***MiniMap();***

The enemy location will be displayed on the map. This will be useful for players to attack enemies and secure themselves.

5.Non-functional Requirements

5.1 Performance

- *The environment of our game is user friendly.*
- *Game will not be changed during its use.*
- *The graphics are good, making the game attractive.*

5.2 Safety

- *As our game is a single player game, there is no risk of data hacking.*
- *Simply means when user data will be on the internet so how it will be hacked.*

5.3 Security

- *Our game is offline.*
- *There is no risk of personal data sharing among different people on the internet.*
- *Because in online games sometimes players buy different weapons using their credit cards. In this situation the misuse of their personal information is high.*

User Documentation

User documentation include the followings:

- *Project Proposal*
- *Software Requirement Specification (SRS)*
- *Project Presentation and Final Documentation*
- *Executable Files*
- *CD*
- *User Manual*

.

DESIGN DOCUMENT

Objective of Design Document:

This design document can be used as a way to visualize our project before it takes place or as documentation for a project afterward. But the overall goal of this document is to allow us to visualize how a project is or will be working.

1.System Architecture Diagram

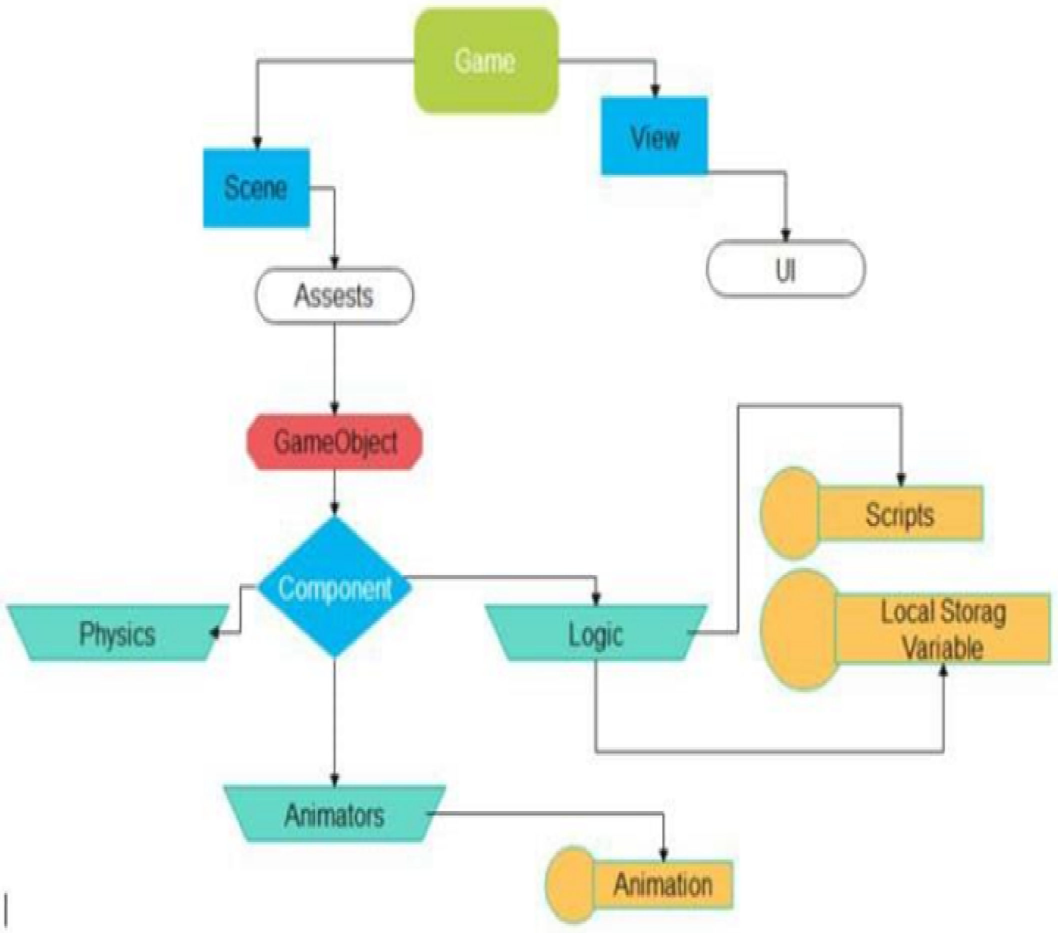


Figure 1: system architecture

Description:

This architectural diagram is a diagram of our system that is used to abstract the overall outline of the software system and the relationships, constraints, and boundaries between components. It is an important tool as it provides an overall view of the physical deployment of the software system like game consists of mainly two components: view and scene .Scene further contain objects like assets and physical and logical components that are related with each other .view contain user interfaces of our game.

2. Sequence diagrams

A sequence diagram shows, as parallel vertical lines (lifelines), different processes or objects that live simultaneously, and, as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner.In our project we use multiple sequence diagrams according to different scenarios.

2.1.Play Game

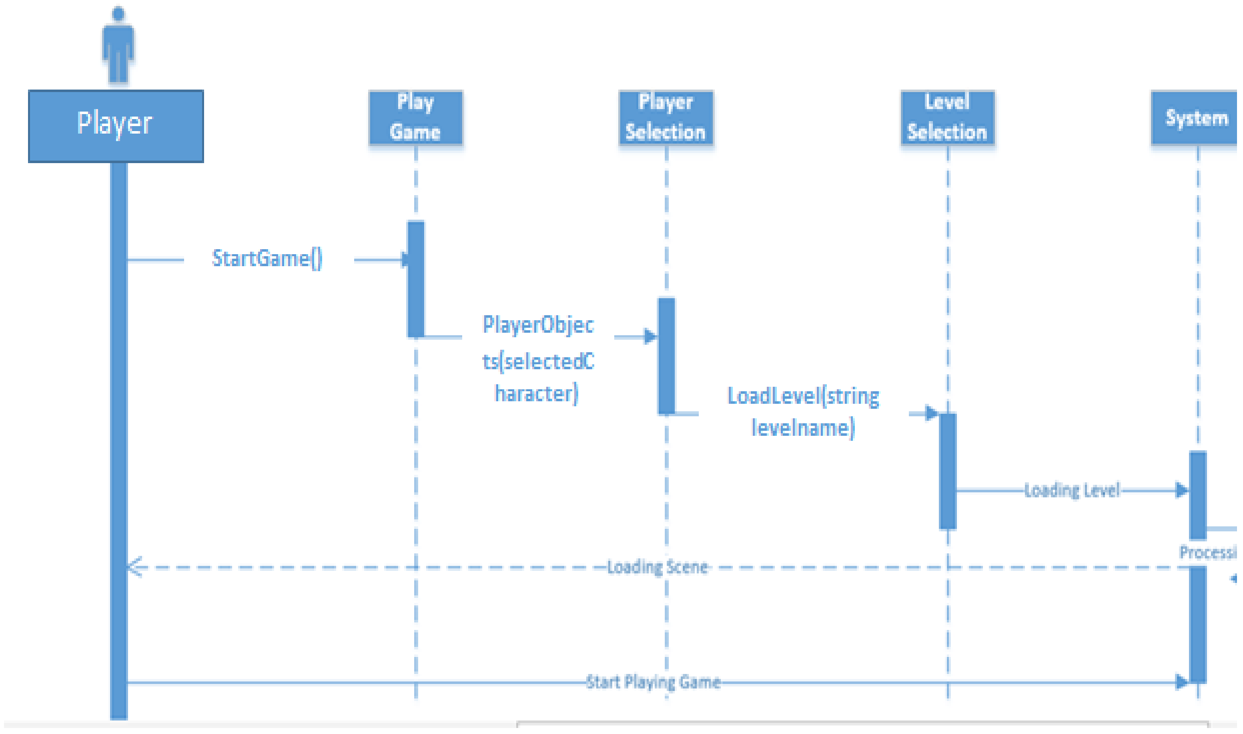


Figure 2Play Game

Description:

*In order to play the game, the player will first click on the play game option. Then the player will have to select a player (in game environment) and to select level. For play game the **StartGame ()** is used. For player selection **PlayerObjects (selected characters**) is being used. The system will load the level. For loading the level **LoadLevel (string levelname**) function will be used. After processing the system will send a reply of the loading scene and after this the player can start playing the game.*

2.2 .Resume Game

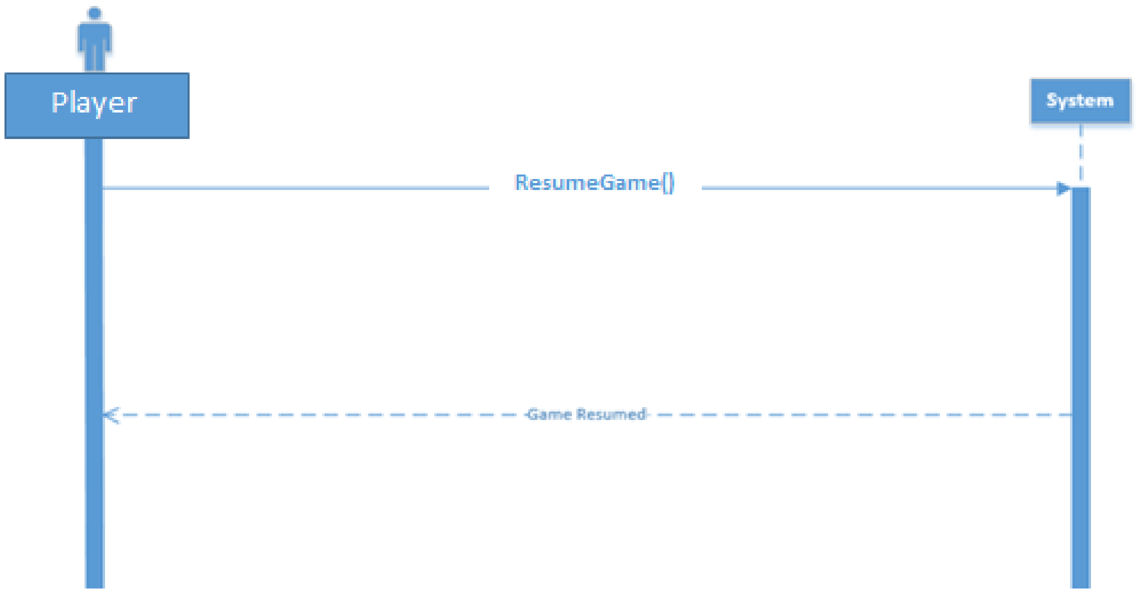
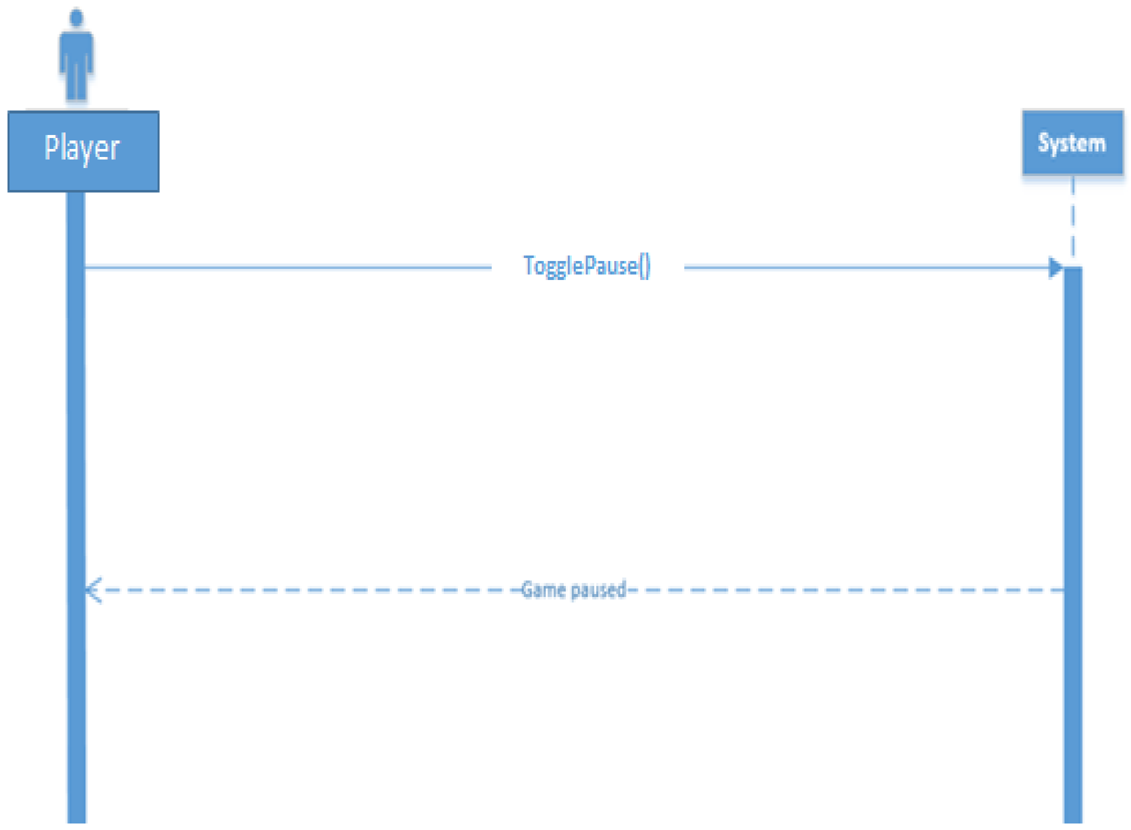


Figure 2 resume game

Description:

*When a player wants to resume the game after he has paused the game, the system will do this by using the ResumeGame**()** function. After resuming the game, the system will send a reply message if the game resumed.*

2.3.Game Paused



<u>Description:</u>
*When a player wants to pause the game, the system will do this by using the TogglePaused**()** function. After pausing the game, the system will send a reply message if the game paused.*

2.4.Level Failed

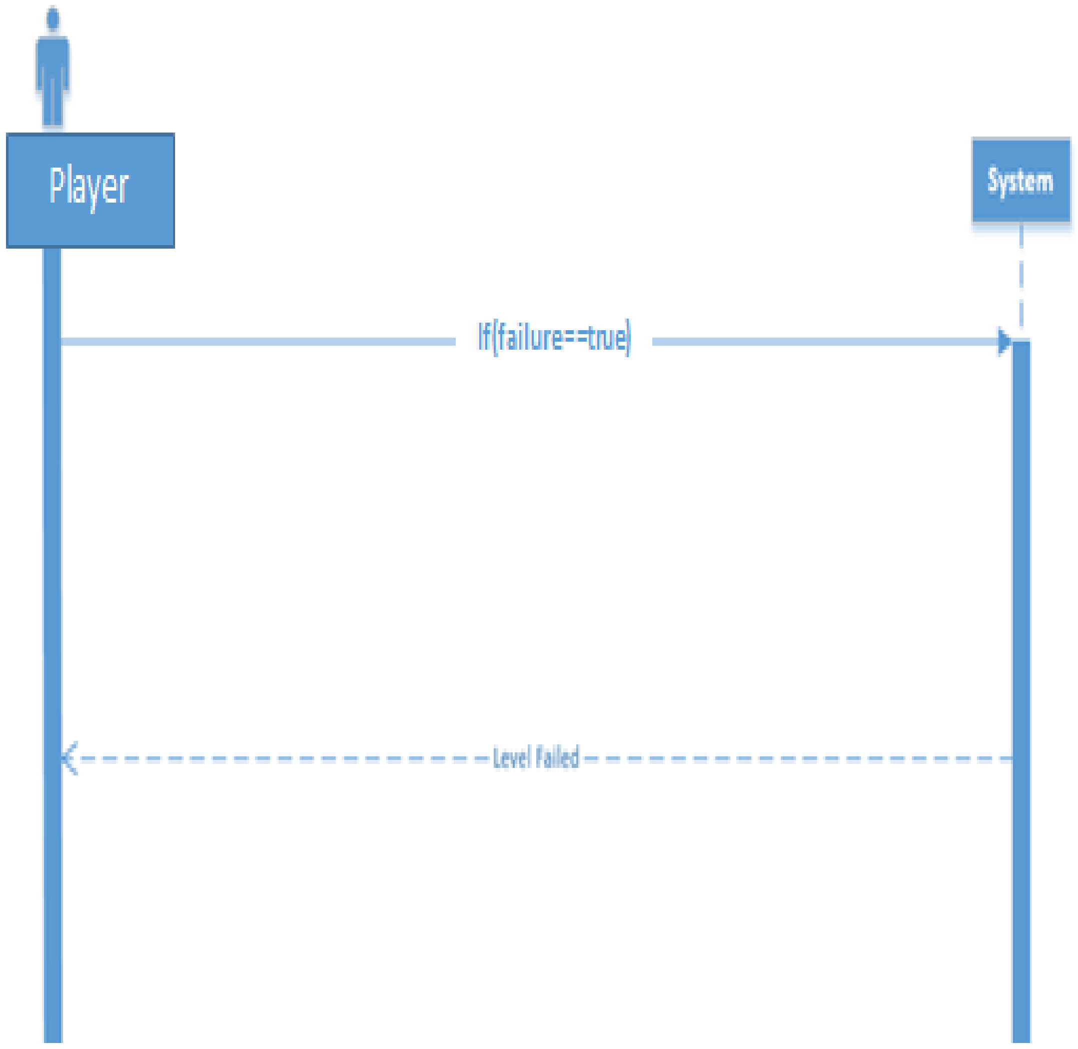


Figure 5; level failed

Description:

In the level failed sequence diagram , on synchronous messages the ***if(failure==true)*** *function will be used to check whether the level is failed or not. Then after checking , the system will send the reply of the level failed message.*

2.5.Game Score

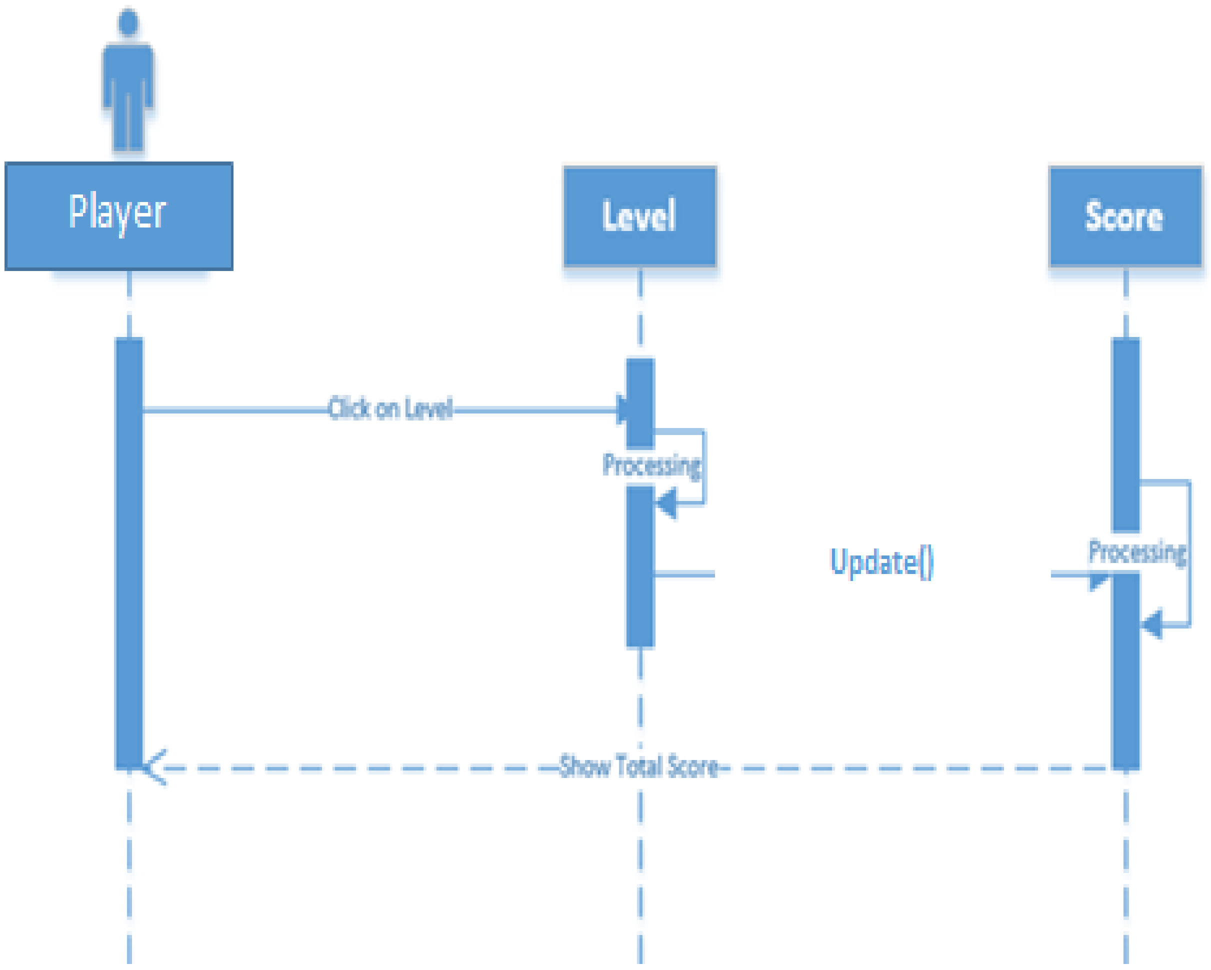


Figure 6 game score

Description:

For getting the score , the player will select a level . The level object will do processing (self message). For getting game scores ,the ***Update()*** *function will be used. Then the level will send feedback to show total score.*

2.6Total Score

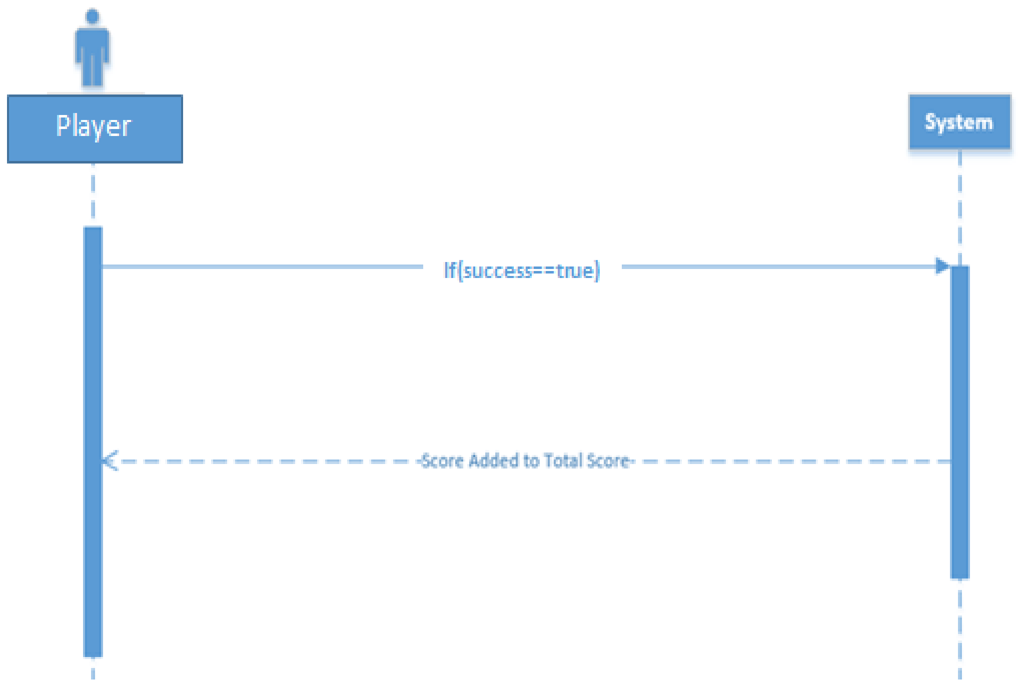


Figure 7 Total score

Description:

For getting TotalScore , on synchronous messages the if(success==true) function will be used to determine if a player has completed the level or not. If the level is completed means success , then the system will send the reply of Score added to the Total Score message.

2.7.Level complete

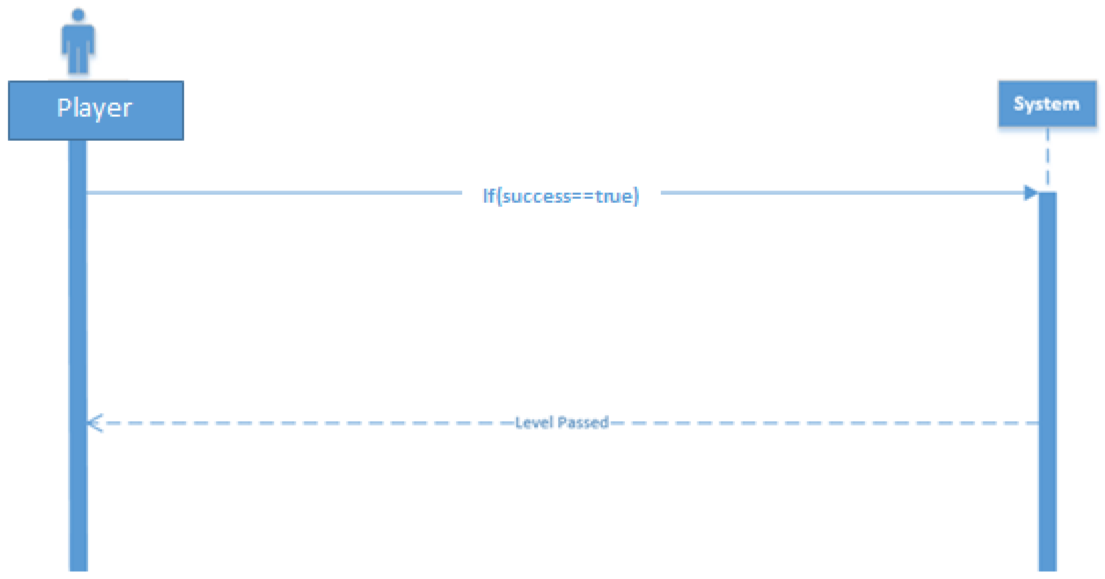


Figure 8 Complete level

Description:

Here in this level complete sequence diagram , on a synchronous message the ***if(success==true)*** *function will be used to check either the level is completed or not. Then after checking , the system will send the reply of level passed means level completed message.*

2.8 Reset Game

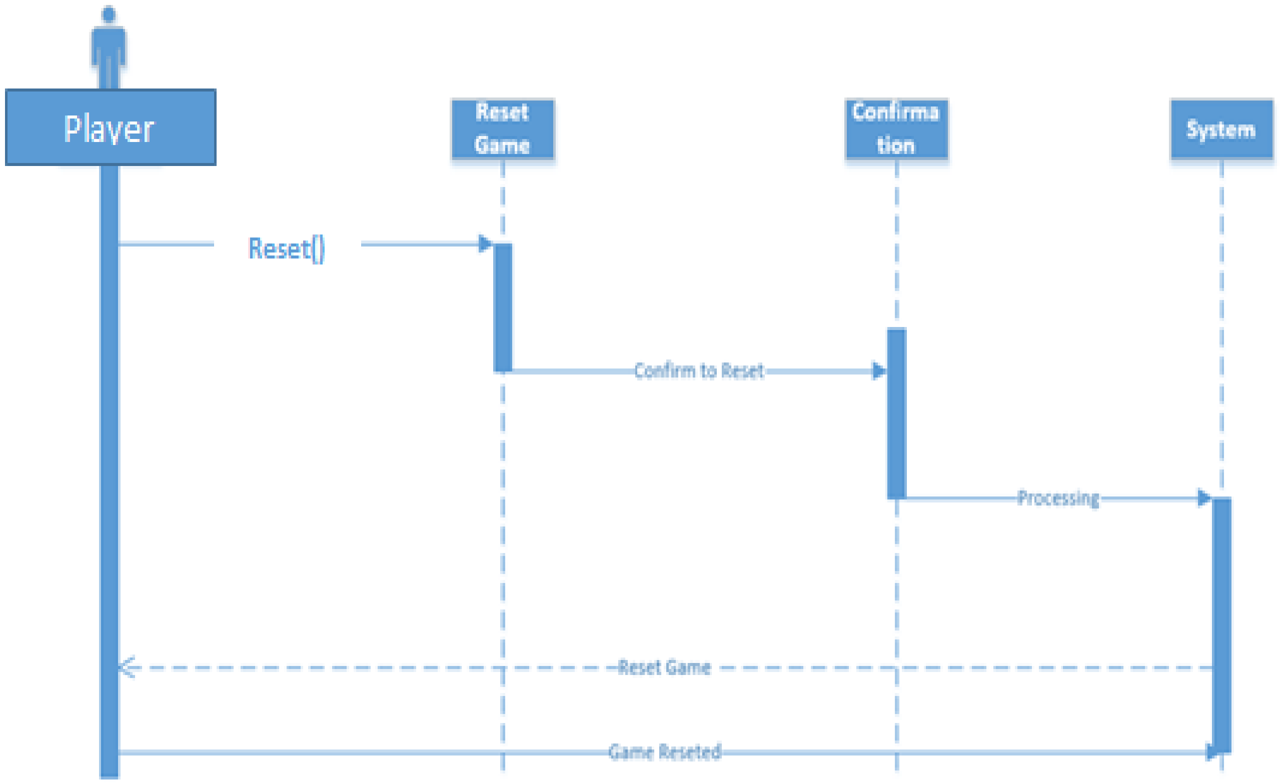


Figure 9 Reset Game

Description:

*In order to reset the game, the **Reset ()** function will be used by the Reset Game object. The confirmation object will do the confirmation of resetting the game. After confirmation, the system will do processing; send the reply of reset game. At last the game will be reset.*

2.10. Level Volume

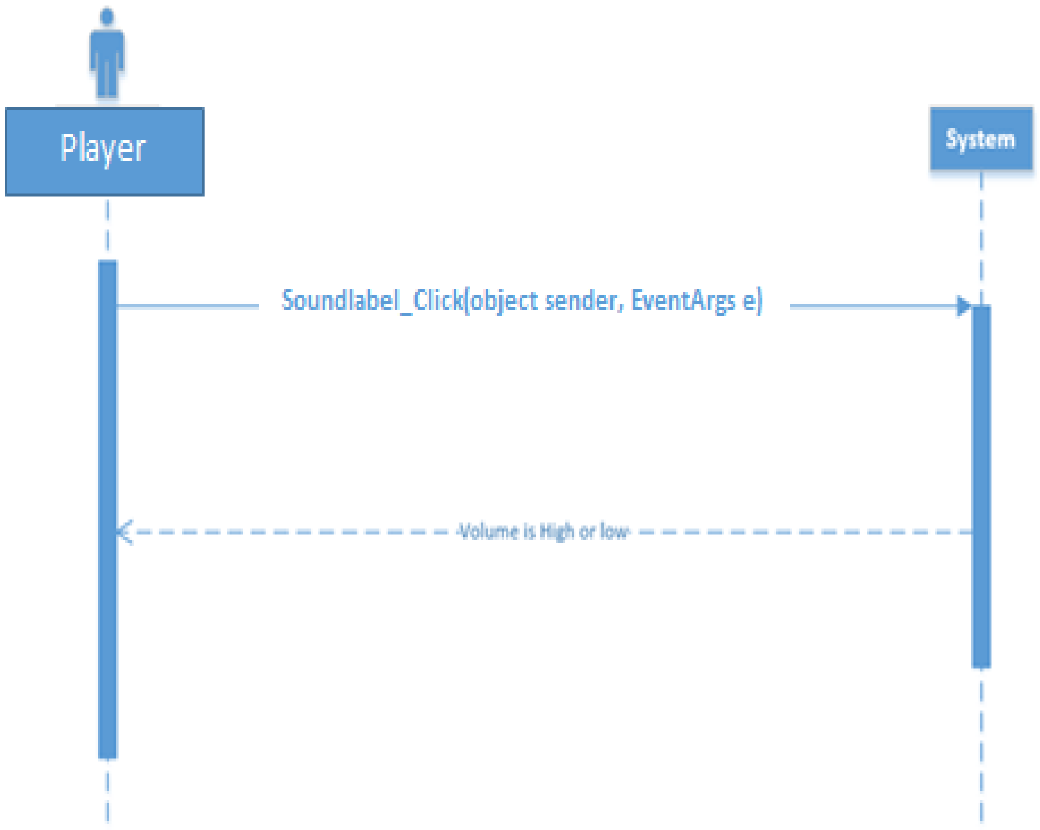


Figure 9 level volume

Description:

*When the player wants to high or low the volume, the **Soundlabel_Click(object sender, EventArgs e)** function will be used. Then the reply of volume high or low will be sent to the player by system.*

2.11.Game Exit

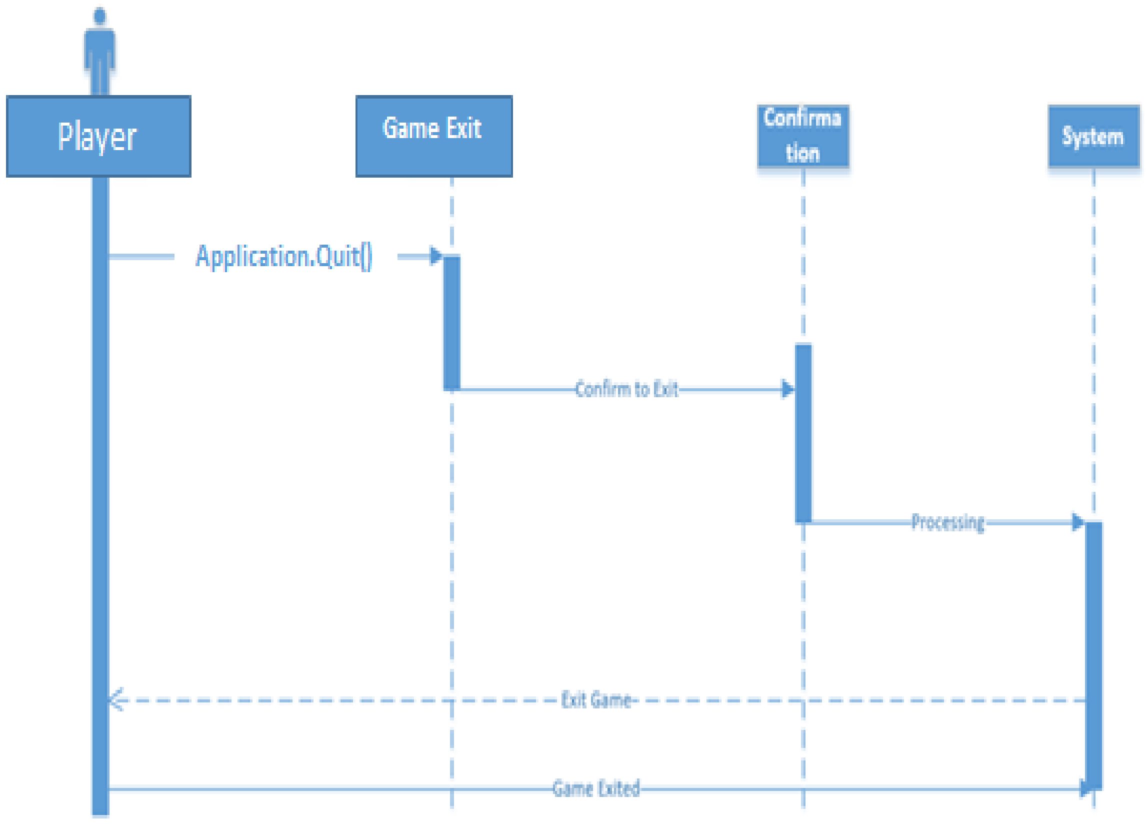


Figure 10 exit game

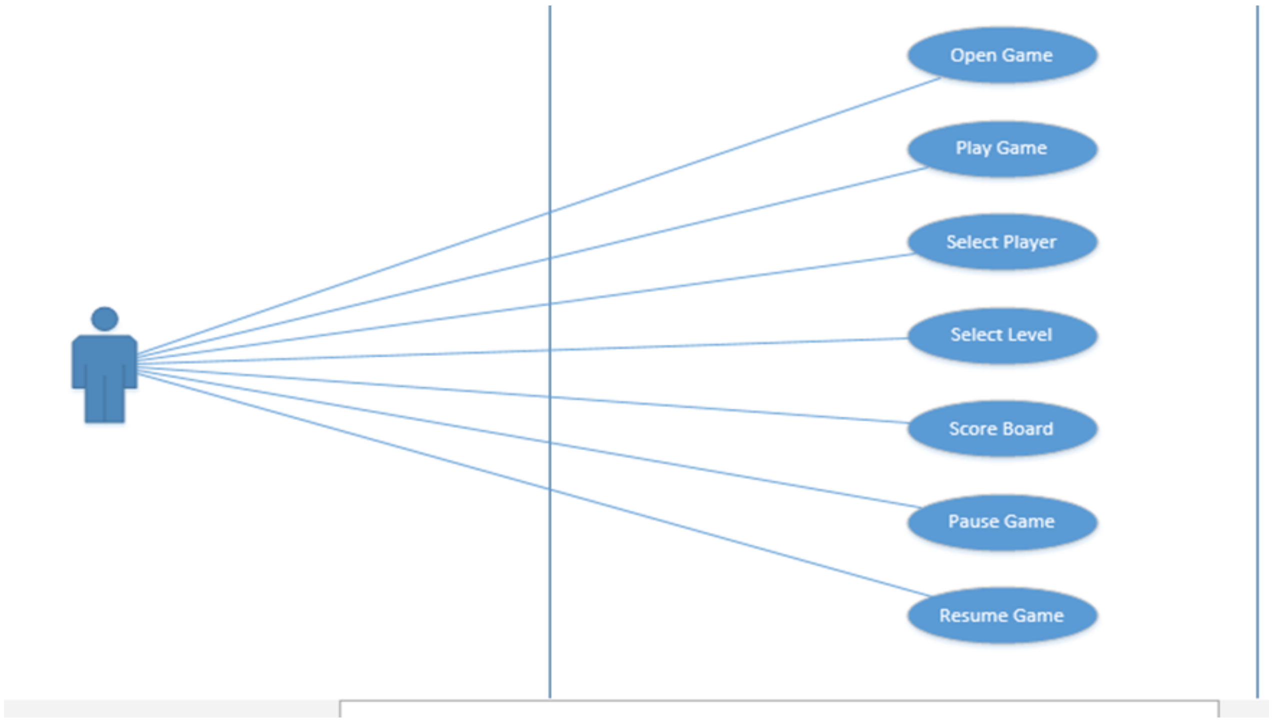
Description:

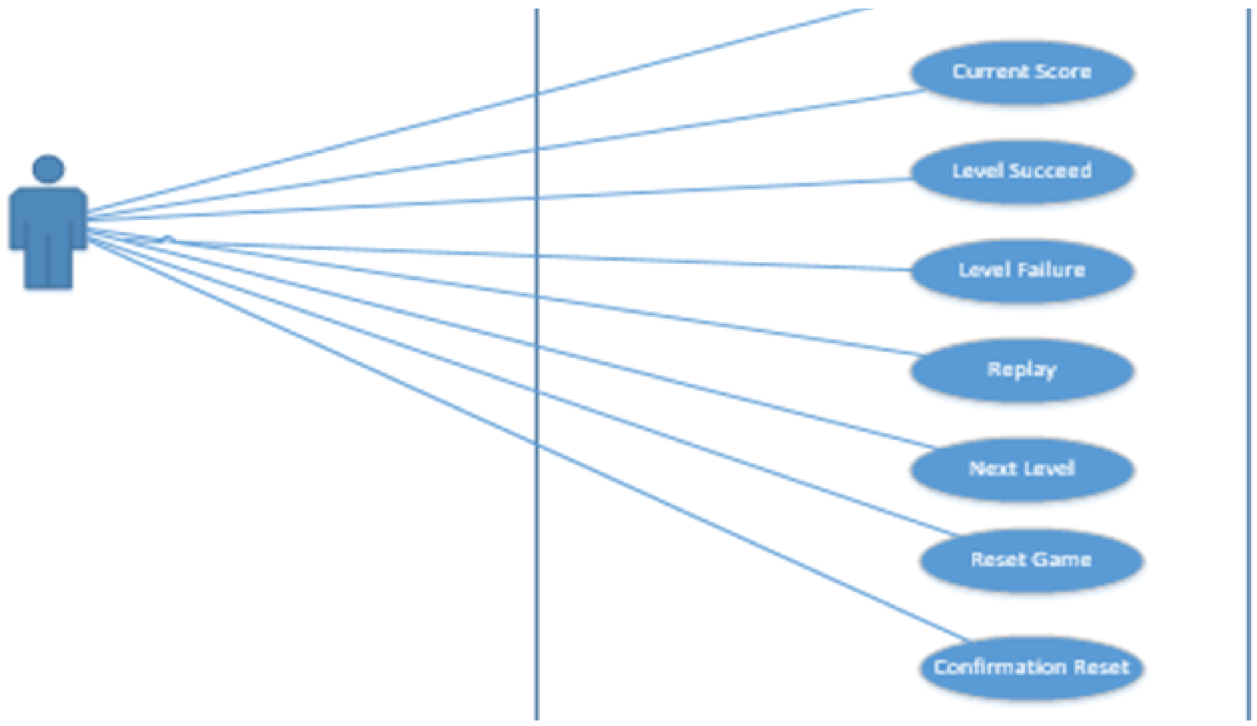
*In order to exit the game, the **Application.Quit()** function will be used by the GameExit object . The confirmation object will do the confirmation of the existing game. After confirmation , the system will do processing, send the reply of the exit game . At last the game will be exciting.*

3.Use case Diagrams

3.1.Usecase diagram for use:

This diagram shows the functionality of the user that a user can perform like playgame (), selectplayer (), pausegame () and so on.




Current Score
Level Succeed
Level Failure
Replay
Next Level
Reset Game
Confirmation Reset

3.2.Use Case Diagram for Start Game

Description:

The user can start the game and stereotype<<include>> represent that user must select the player after that user can select the level in which he wants to play. User can also do the sound setting, maybe he prefers the low sound or high.

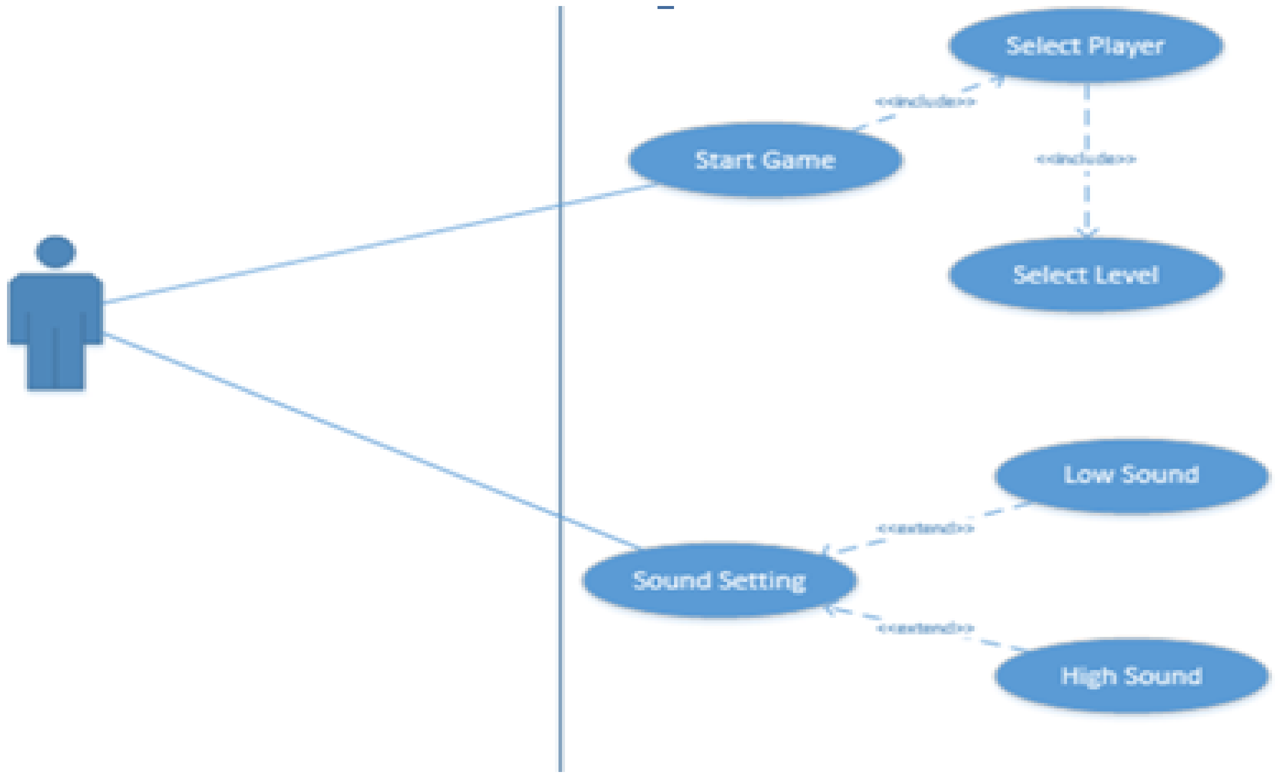


Figure Start Game

3.3.Use Case Diagram for Select Level:

User selects the level in which he wants to play. Here there are four levels, to reach at level 4 users must complete the remaining levels.

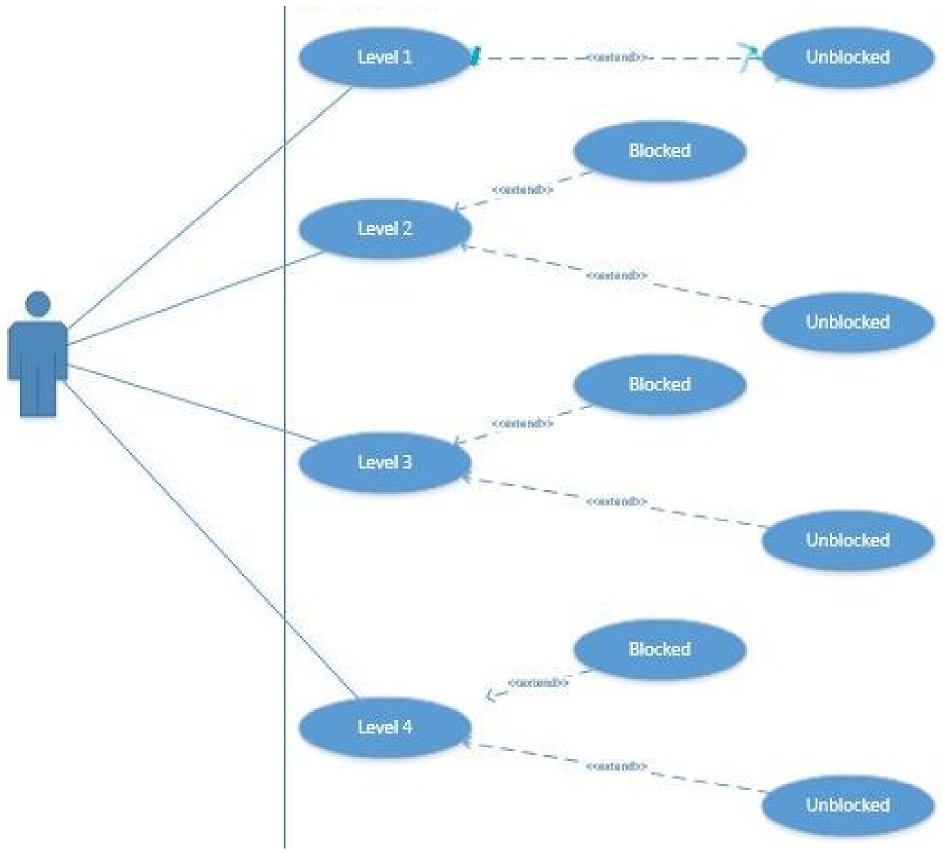


Figure 13 Select level

3.4.Use Case Diagram for Next Level

Description:

In which the user selects the next level if the previous levels are complete or may the next level be unblocked if the user gets the required score for the completion of a level.

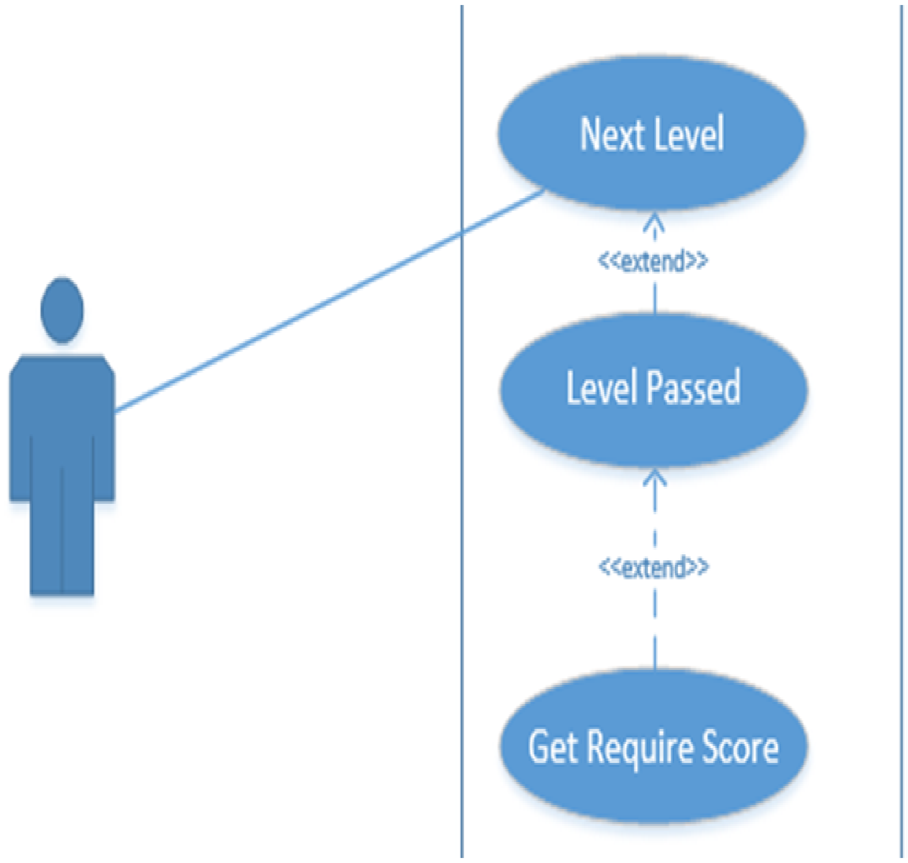
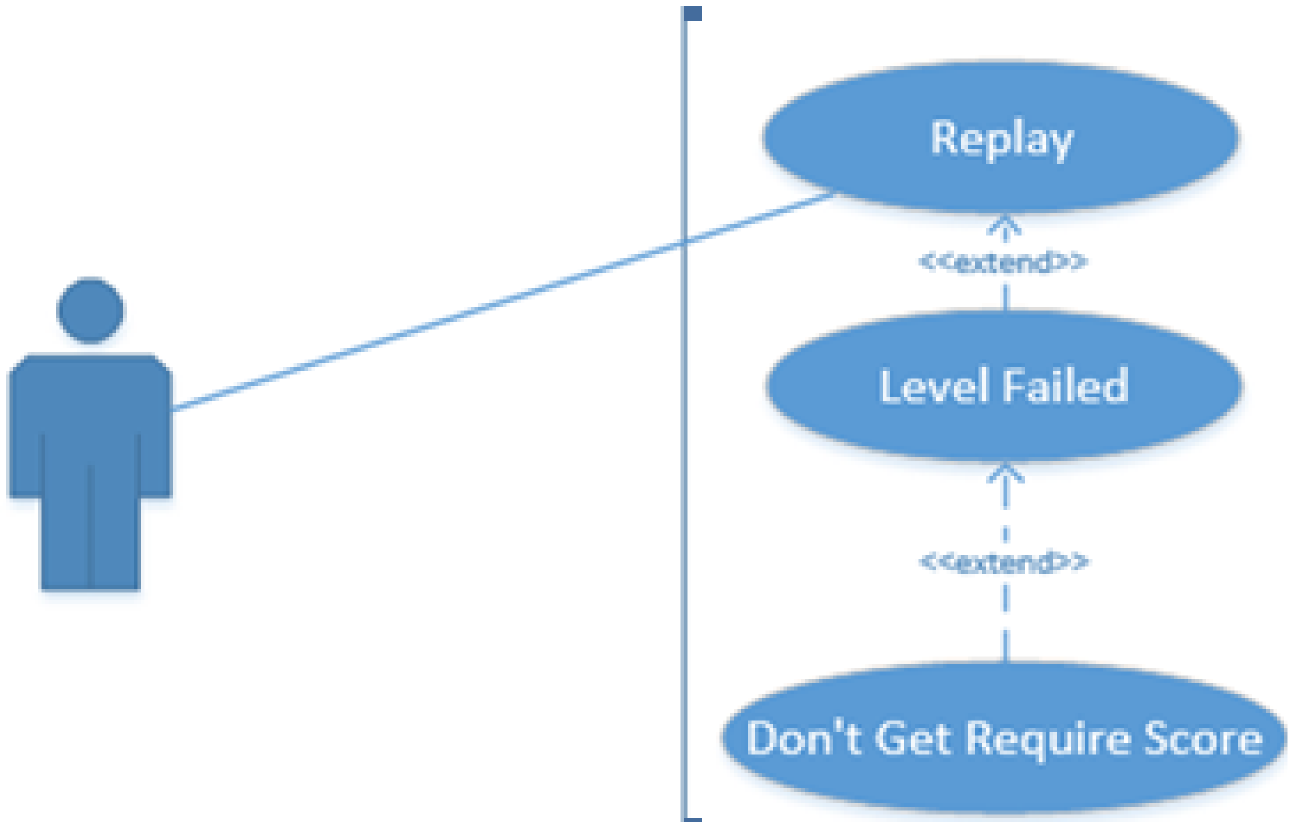


Figure Next level

3.5. Use Case diagram for user Replay

It shows that maybe the user wants to replay the game if the level is failed or if the user wants to improve their scores so he goes back and replay the game.



3.6.Use Case Diagram for Exit

The user wants to quit () the game a popup will appear for the confirmation whether a user really wants to quit () the game or continue the game.

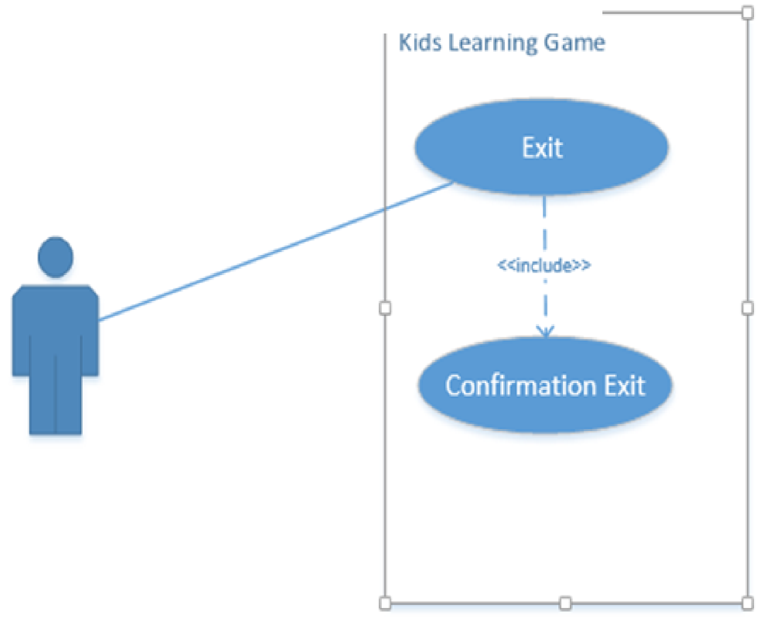


Figure 16 Exit

4 Use Case Tables

4.1Use Case for New Game

Goal:	*Play (start game)*
Primary Actor:	*User*
Level:	*User, System*
Precondition:	*Front screen (starting screen) appears*
Success End:	*Game start*
Failure End Condition:	*Game can't start (some errors)*
Main Success Scenario:	*1.User click on game icon* *2.The system display front screen with menu list* *3. Click on the play button.* *4. The new screen is the display of Player Selection.* *5. After player selection, a new screen opens with a display of Level Selection.* *6. System successfully starts the game after selecting the level that is already open.*
Extensions (Error Scenarios):	*3a. game hang* *4a. if available memory not enough to play the game* *4a.1 System returns error* *4a.3 The selected level may not open before.*

4.2 Use case for Exit/Quit Game

Goal:	*Exit*
Primary Actor:	*User*
Level:	*User, System*
Precondition:	*Front screen (starting screen) appears*
Success End:	*Exit from Game*
Failure End Condition:	*Game can't exit*
Main Success Scenario:	*1. The system display front screen with menu list* *2. Click on quit button* *3. A new screen comes in front of the user with a confirmation message.* *4. System successfully exit from game by clicking on Yes otherwise not successfully exited.*
Extensions (Error Scenarios):	*2a. If the quit button does not perform, the game cannot quit.* *2a. If the user selects "NO" from the confirmation message, then the system also doesn't quit.* *4a. System returns error* *4a. system don't respond on command*

4.3.Use Case for Game level selection

Goal:	*Levels*
Primary Actor:	*User*
Level:	*User, System*
Precondition:	*Play button appears, click on it, select the player and select levels.*
Success End:	*Complete the current level and move to next level*
Failure End Condition:	*Game can't start or unlock the next level after complete the previous level (some errors)*
Main Success Scenario:	*1. User click on play button* *2.The system display front screen with menu list* *3.Click on play button* *4. System show player selection screen and after player being chosen level selection screen shows.* *5. The user selects the level which he/she wants to play.* *6. The level may belong to character or car.*
Extensions (Error Scenarios):	*2a. If you select the next level before the previous level it cannot play before winning first level.* *3a. System returns error* *4a.1 User close game app and try game*

4.4 Use Case for Score Management

Goal:	*Score*
Primary Actor:	*User*
Level:	*User, System*
Precondition:	*User collect the coins means alphabets to increase the score*
Success End:	*Unlock the next level buy collect alphabets and achieve target score*
Failure End Condition:	*If you can't achieve the target score net level can't unlock*
Main Success Scenario:	*1. System successfully start game* *2.User collect coins (alphabets) to increase score*
Extensions (Error Scenarios):	*4a If a user cannot collect many alphabets to achieve the target score, the next level cannot unlock.* *4a System returns error* *2a.1 users must play preview level to unlock higher level.*

4.5 Use Case for Game Sound Management

Goal:	*Game Sound*
Primary Actor:	*User*
Level:	*User, System*
Precondition:	*Sound setting is available in the sound setting button on the screen where the menu list is present (First Screen).*
Success End:	*Set game sound high or low*
Failure End Condition:	*System may not respond user cannot set sound may error occurs*
Main Success Scenario:	*1. User click on game setting* *2. Click to set the sound high or low.*
Extensions (Error Scenarios):	*2a. If the system does not respond properly, you may not set the game sound.* *4a. System returns error* *4a.1 User back out or tries again.*

4.5 Use Case for Level Complete (Success)

Goal:	*Level complete*
Primary Actor:	*User*
Level:	*User, System*
Precondition:	*Play button appears click to play the game*
Success End:	*After playing, the level is completed successfully.*
Failure End Condition:	*Game level may not complete*
Main Success Scenario:	*1. User click on play button* *2. Select the player and level* *3. Play the current level.* *4. After play the game level is completed when he/she achieve the targeted goal.* *5. System successfully move the screen where user can* *select the next level.*
Extensions (Error Scenarios):	*1. The target score can't be achieved.* *2a. System may not show a success message after completion of level.* *4a. System returns error* *4a.1 User close game app and try again.*

4.6. Use Case for new game

Goal:	*Play (start game)*
Primary Actor:	*User*
Level:	*User, System*
Precondition:	*Front screen (starting screen) appears*
Success End:	*Game start*
Failure End Condition:	*Game can't start (some errors)*
Main Success Scenario:	*1.User click on game icon* *2.The system display front screen with menu list* *3.Click on the play button.* *4. The new screen is the display of Player Selection.* *5. After player selection, a new screen opens with a display of Level Selection.* *6.System successfully starts the game after selecting the level that is already open.*
Extensions (Error Scenarios):	*3a. game hang* *4a. if available memory not enough to play the game* *4a.1 System returns error* *4a.3 The selected level may not open before.*

Use Case for Score Management:

Goal:	*Score*
Primary Actor:	*User*
Level:	*User, System*
Precondition:	*User collect the coins means alphabets to increase the score*
Success End:	*Unlock the next level buy collect alphabets and achieve target score*
Failure End Condition:	*If you can't achieve the target score net level can't unlock*
Main Success Scenario:	*1. System successfully start game* *2.User collect coins (alphabets) to increase score*
Extensions (Error Scenarios):	*4a If a user cannot collect many alphabets to achieve the target score, the next level cannot unlock.* *4a System returns error* *2a.1 users must play preview level to unlock higher level.*

Use Case for Game Sound Management
Use Case for Game level failure

Goal:	*Level failure*
Primary Actor:	*User*
Level:	*User, System*
Precondition:	*Play button appears click to play the game*
Success End:	*After playing, the level may not complete successfully.*
Failure End Condition:	*Game level may not complete*
Main Success Scenario:	*1. Play the current level.* *2. After play, the game level is not completed.* *3. The time is over.* *4. System not move the user to the next level.*
Extensions (Error Scenarios):	*2a. system may hang when level fails.* *3a. System returns error* *4a.1 game app close and try again.*

Goal:	*Game Sound*
Primary Actor:	*User*
Level:	*User, System*
Precondition:	*Sound setting is available in the sound setting button on the screen where the menu list is present (First Screen).*
Success End:	*Set game sound high or low*
Failure End Condition:	*System may not respond user cannot set sound may error occurs*
Main Success Scenario:	*1. User click on game setting* *2. Click to set the sound high or low.*
Extensions (Error Scenarios):	*2a. If the system does not respond properly, you may not set the game sound.* *4a. System returns error* *4a.1 User back out or tries again.*

Use Case for reset/restore game

Goal:	*Reset*
Primary Actor:	*User*
Level:	*User, System*
Precondition:	*Front screen (starting screen) appears*
Success End:	*Reset Game*
Failure End Condition:	*Game can't reset*
Main Success Scenario:	*1. User click on game icon* *2. The system display front screen with menu list* *3. Click on reset button* *4. A new screen comes in front of the user with a confirmation message.* *5. System successfully reset from game by clicking on Yes otherwise not successfully reset.*
Extensions (Error Scenarios):	*2a. If the reset button does not perform, the game cannot quit.* *If the user selects "NO" from the confirmation message then the system also doesn't reset.* *4a.1 System returns error* *4a.2 system don't respond on command*

<u>Class Diagram of VSA. RPG</u>

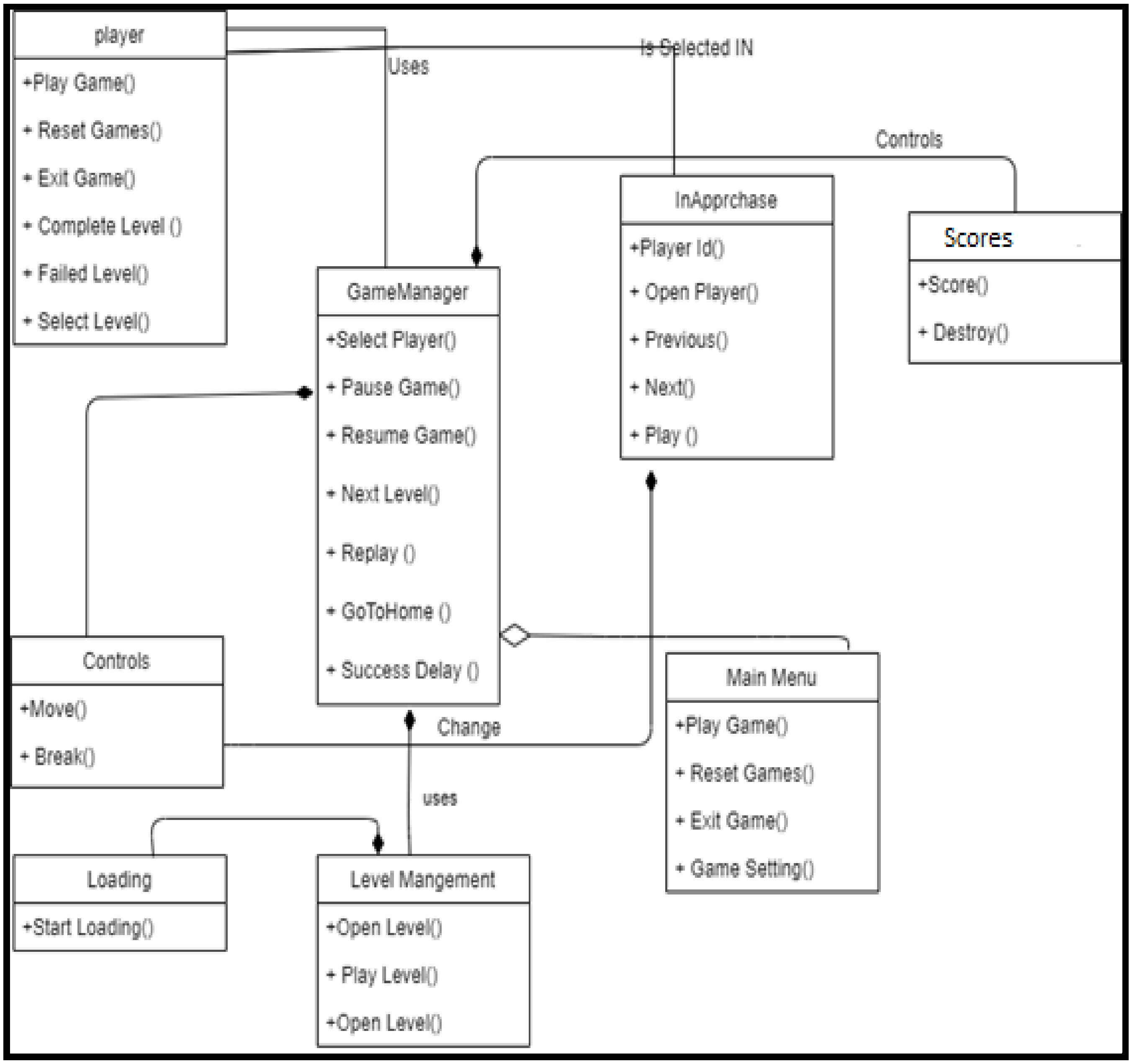


Figure 17 class diagram

Description of class diagram:

Player:

This class contain the major functionalities like,

Playgame (): *User will play the game.*
RestGame (): *By pressing this button the user will reset the game.*
ExitGame () *If the user wants to leave the game, then click the exit button.*
CompleteLevel (): *When the level is complete and the user gets the required scores for completion of level then this function is called.*
FailedLevel (): *When the user remains unsuccessful to complete the level then this function is called.*
SelectLevel (): *According to the desire user can select the level and enjoy the game.*

Game Manager:

Game manager will manage the whole game functionality like,

SelectionPlayer (): *To select the player.*
PauseGame (): *To pause or stop the game for a while.*
ResumeGame (): *Again, continue the game.*
Replay (): *To restart the game or a level.*
GotoHome (): *if the user wants to go back to the home, then this function is called.*

InAppPurchase:

It contains the following functions like,

PlayerId (): *Will store the id of player to uniquely identify the player.*
previous (): *To go to the previous state.*
Next (): *To go forward.*
Play ()*: To start the game.*

Scores:

This class controls by the game manager class that

Scores ()*: To store the record.*
Destroying ()*: And also release the memory by destroying the record of the user.*

Controls:

This class contain the two major functions:

Move (): *To move the player left and right up and down by pressing the keys of the keyboard.*
Break (): *To stop the player.*

Loading:

StartLoading (): *This class will load the game by calling* ***StartLoading ()*** *function.*

Level Management:

OpenLevel (): *To unblock or open the level.*
PlayLevel (): *To play a level.*

MainMenue:

This class is handled by the MainMenue functions like ***PlayGame (), RestGame (), ExitGame ()*** *and* ***SettingGame ().***

Data flow diagram of village survival action RPG

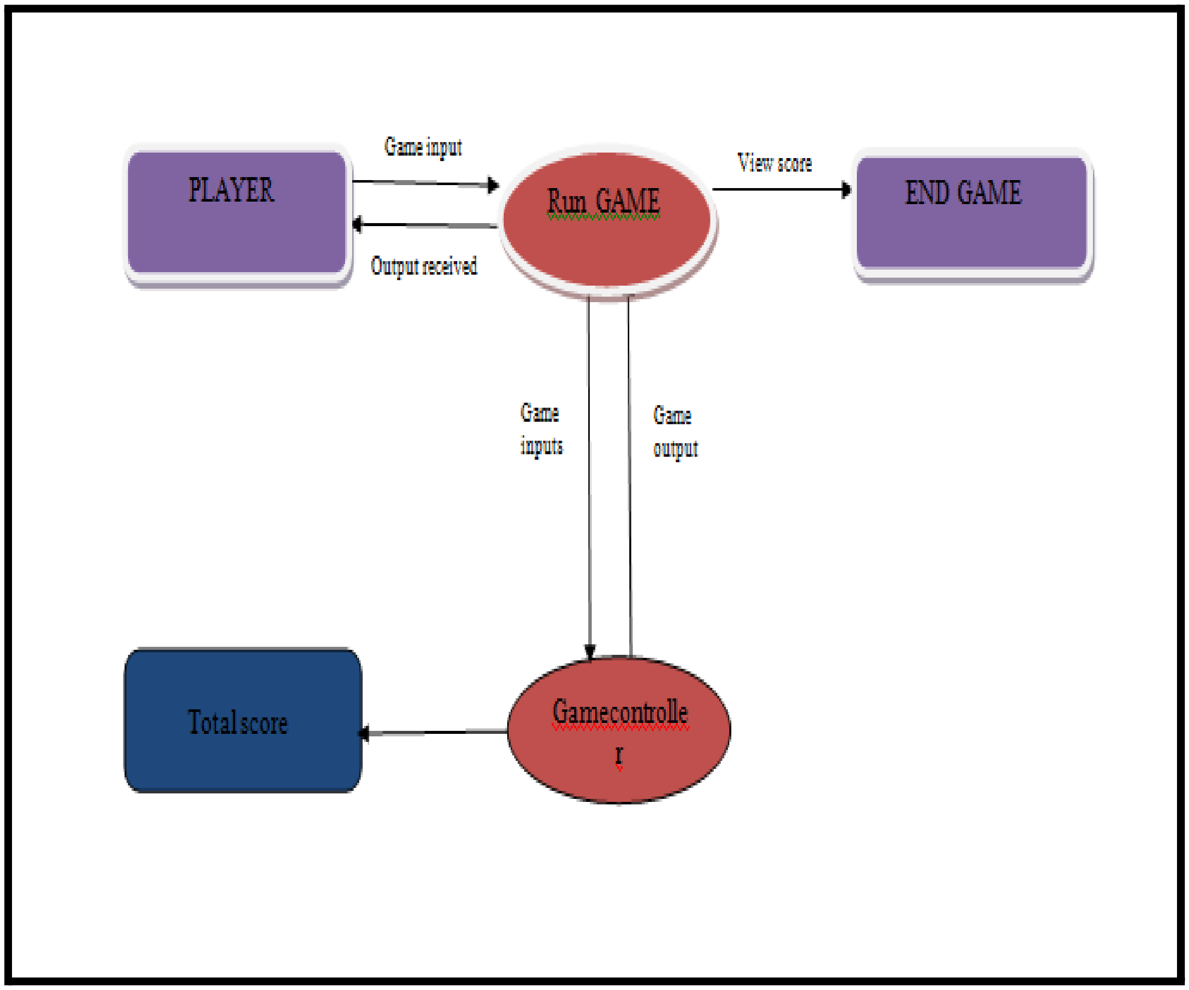


Figure 18 Data Flow Diagram

Description:

Players start the game by giving an input from the keyboard if the user wants to view the score it presses the score button and also sees the history.Game controller manages the scores and total scores after playing the game. the user will end the game.

CONSTRUCTIVE COST MODEL (COCOMO)

Introduction:

Video games or mobile games is a big concern which has been the most important topic till date. Video games can teach important skills or address serious issues, organizations such as games for change promote the use of games for education and social action, and often involve young people in the creation of games they can use to express themselves on important issues and current events.

Abstract:

This paper provides a real sort of applying COCOMO as an estimation technique for the required software development effort in a game development project. The main goal and contribution of the case study is to support the research on software effort estimation and to provide software practitioners with useful data based on a real project. The alternatives obtained from results will be used for future work in order to increase the effort estimation accuracy in safety-critical software projects.

Cocomo class:

We create a game:

This game has objectives

They can provide a fun and social form of entertainment.

Increase children’s self-confidence and self-esteem as they master games.

Develop skills in reading, math, technology and problem-solving.

Provide points of common interest and opportunities for socialization.

This is a user-friendly application that can be accessed by anyone who has installed it in their smartphones. Our intention is to provide you with the best and attractive graphics so that everyone can enjoy it. Our proposed game application lies under ORGANIC class of COCOMO.As the team size required for our proposed system is adequately small, the problem is well understood and has been solved in the past and also the team members have a nominal experience regarding the problem.

Stages of cocomo:

There are three levels of the model:

Basic
Intermediate
Detailed

Basic cocomo level 1:

The first level, Basic COCOMO can be used for quick and slightly rough calculations of Software Costs. Its accuracy is somewhat restricted due to the absence of sufficient factor considerations.

Software Projects	**Constant values**			
	a	b	*c*	d
Organic	2.4	1.05	2.5	0.38
Semi-detached	3.0	1.12	2.5	0.35
Embedded	3.6	1.20	2.5	0.32

Fig.1 Basic Level Estimation

<u>Basic coccomo level 1:</u>

<u>Input:</u>

project mode: Organic

Project size: Small

<u>Output:</u>

Effort: 2740028.08309

Duration: 7075760.12034

Average staffing: 0.3872415170

Effort estimation: a_b (KLOC) $^{b}{}_{b}$

Duration: c_b (EFFORT) $^{d}{}_{b}$

Average staffing: E/D

LOC: lines of codes.

Our proposed system LOC approximation is 4000 LOC.

Effort estimation	***= a_b (KLOC)^{b_b}***
	= 2.4 (400K)$^{1.05}$
	=2740028.08309

Fig.2 Basic Level effort Estimation

Duration	***= c_b (EFFORT)^{d_b}***
	= 2.5 (2740028.08309)$^{0.38}$
	=7075760.12034

Fig. 3Basic Level Duration Estimation

Average staffing	***= EFFORT/DURATION***
	= 2740028.08309/7075760.12034
	=0.3872415170

Fig. 4 Basic Level Duration Estimation

Intermediate cocomo level:

The Intermediate COCOMO model computes software development effort as a function of program size and a set of "cost drivers" that include subjective assessments of product, hardware, personnel and project attributes.The Intermediate COCOMO model computes software development effort as a function of program size and a set of "cost drivers" that include subjective assessments of product, hardware, personnel and project attributes.

Software Projects	Constant values			
	a	*b*	*c*	*d*
Organic	3.2	1.05	2.5	0.38
Semi-detached	3.0	1.12	2.5	0.35
Embedded	2.8	1.20	2.5	0.32

fig:5 intermediate level estimation value 1

EFFORT ESTIMATION	$=a_i \text{ (KLOC) }^{b_i}$
	$= 3.2\ (400\text{K})^{1.05}$
	=2740028.08309

fig:6 intermediate effort estimation value 1

<u>Intermediate Cocomo model Level 1</u>

<u>INPUT:</u>

PROJECT MODE: Organic

PROJECT SIZE: Small (4000 KLOC)

<u>OUTPUT:</u>

EFFORT: 2740028.08309

ESTIMATE: Obtained by BASIC COCOMO

It returns estimates by using 15 set cost drivers.

Effort estimation

=ai (KLOC) bi

Our proposed system LOC approximation is 400 LOC.

LOC: lines of codes.

GUI's for VSA RPG

GUI means Graphical User Interface. It is the common user Interface that includes Graphical representation like buttons and icons and communication can be performed by interacting with these icons rather than the usual text-based or command-based communication.

GUI for Splash Screen

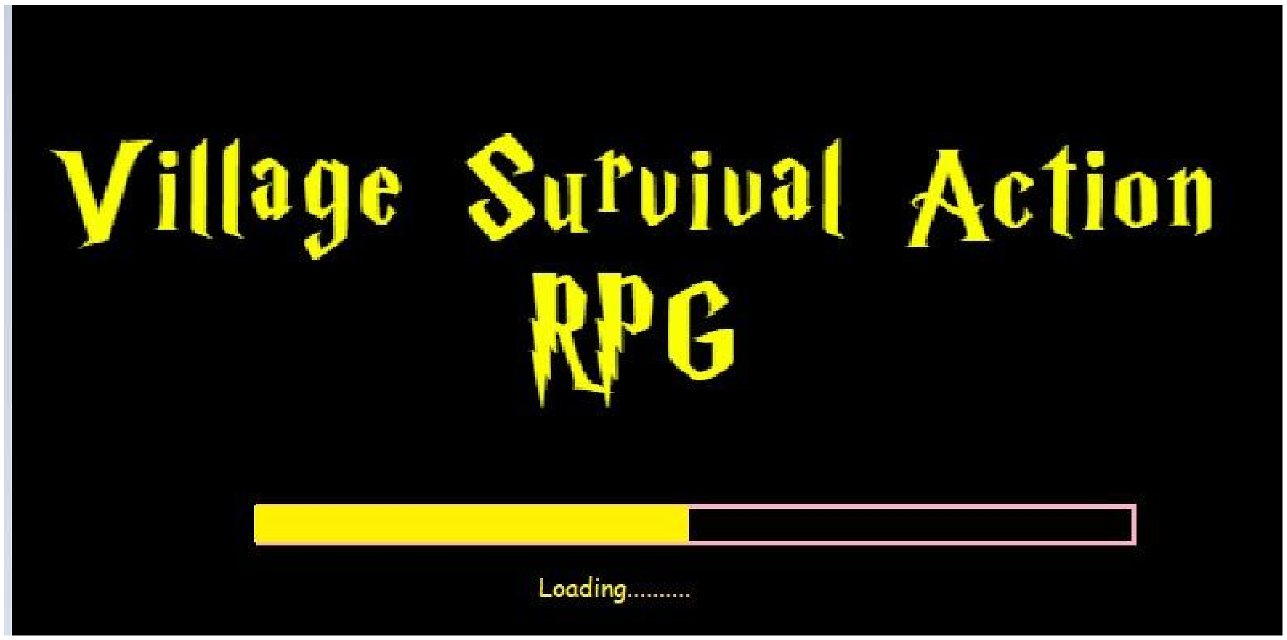


Fig.19 splash screen

Description

The above GUI is for the splash screen of a VSA RPG game. splash screen is a screen which displays first to the user when he/she opens the application/game. It includes the game name and progress bar for showing the loading state of the game.

GUI for Main Menu

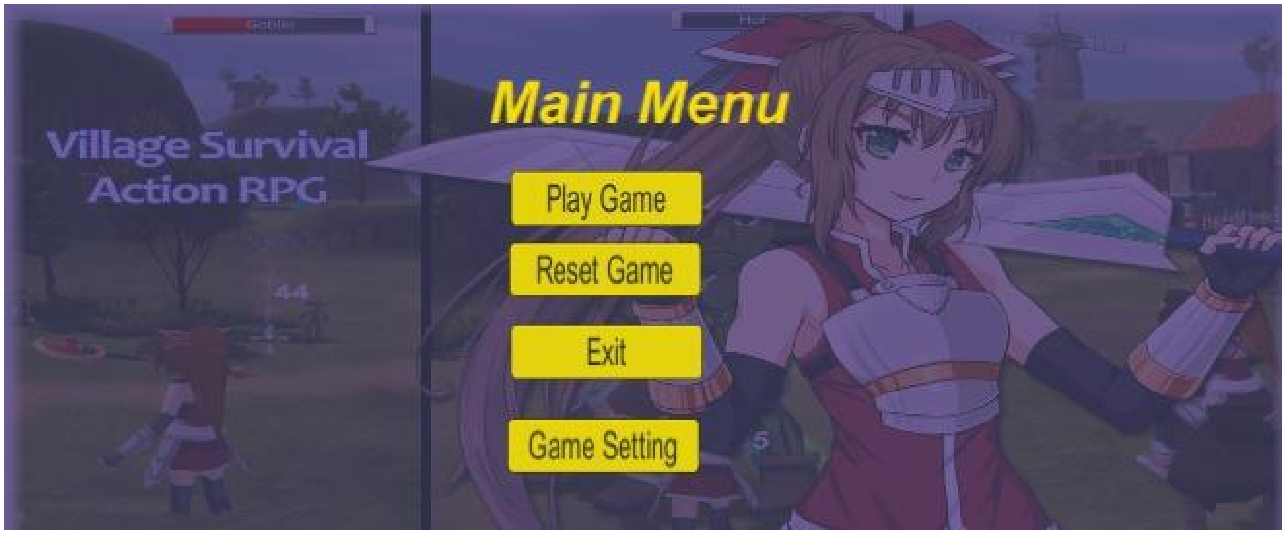


Fig.20 Main Menu

Description

The above GUI is for the main menu of a VSA RPG game. It includes the Play Game, Reset game, Exit and Game Setting options , for the player to select.

GUI for Game Setting

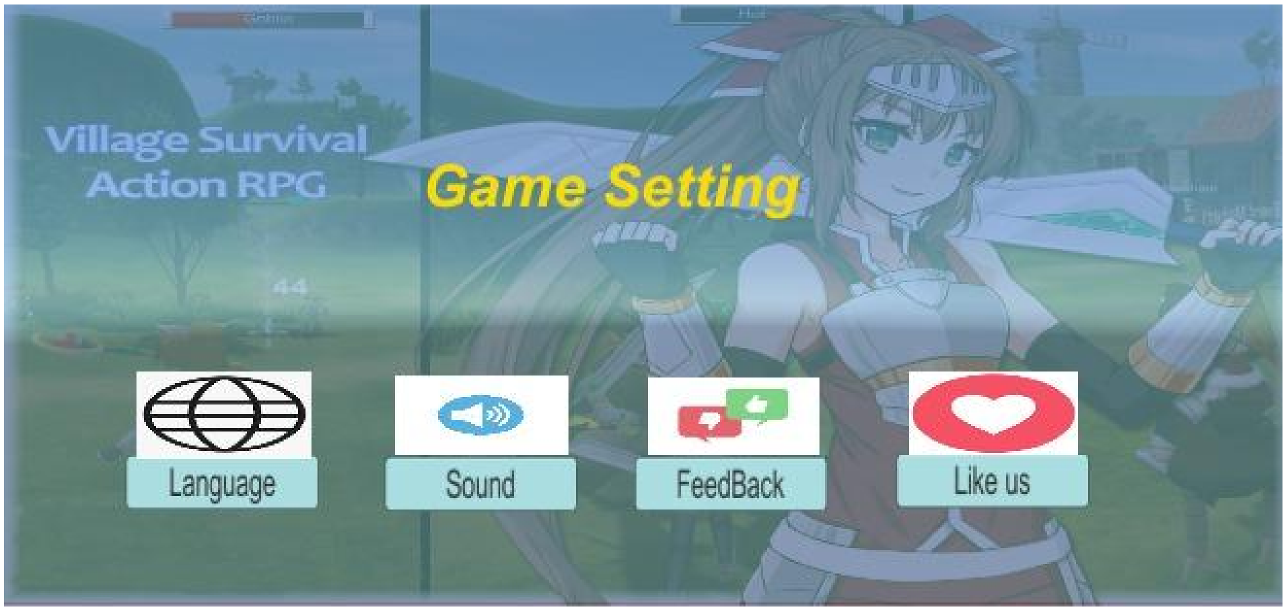


Fig.21 Game setting

Description

This GUI is for the game setting of a VSA RPG game. It includes the Language, Sound, FeedBack and Like us options, for the player to select.

GUI for Level Completed

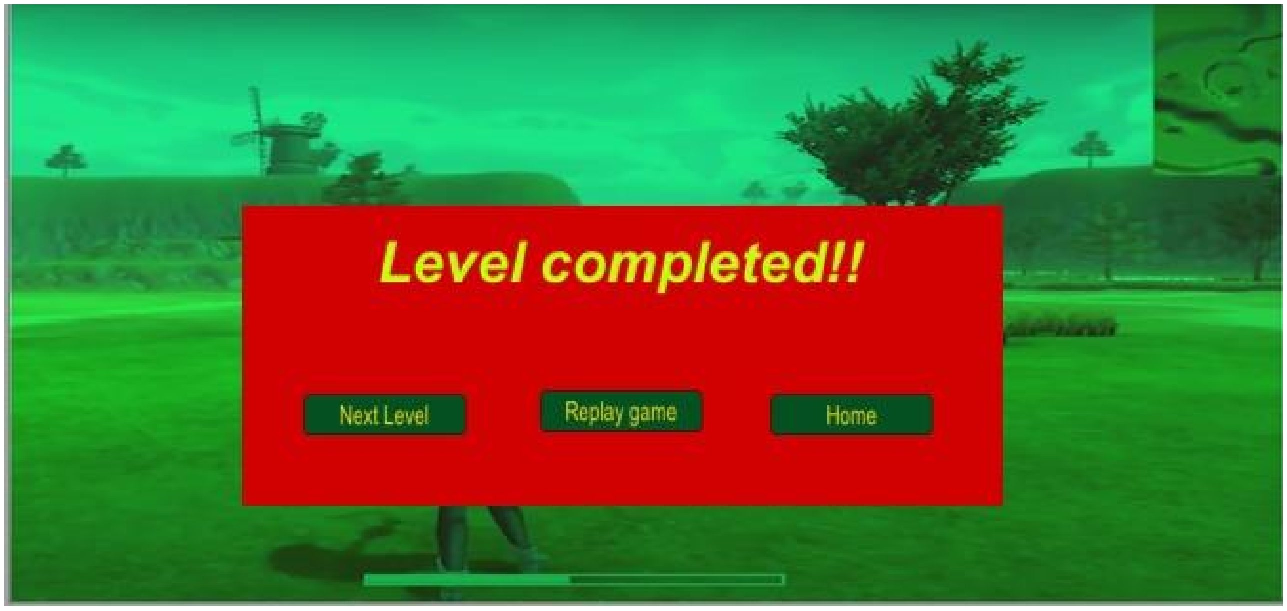


Fig.22 Level completed

Description

This GUI is for showing level completion of a VSA RPG game. It includes the Next Level, Replay game and Home options , for the player to select.

GUI for Level Failed

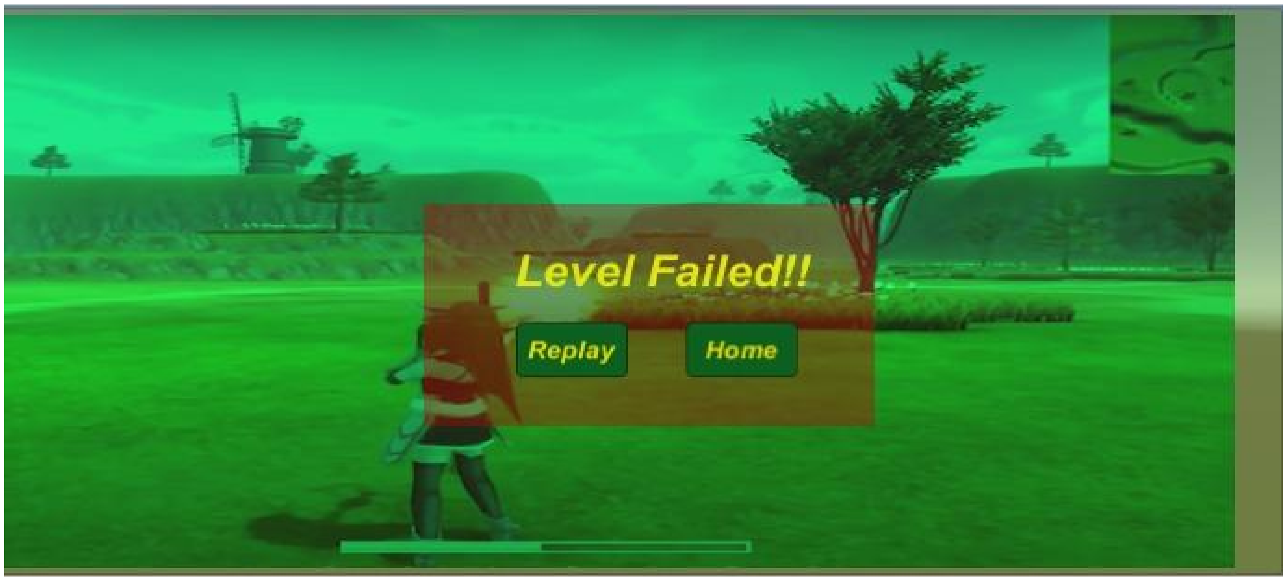


Fig.23 Level failed

Description

The above GUI is for the display of level failure of a VSA RPG game. It includes the Replay and Home options , for the player to select.

GUI for Game Pause

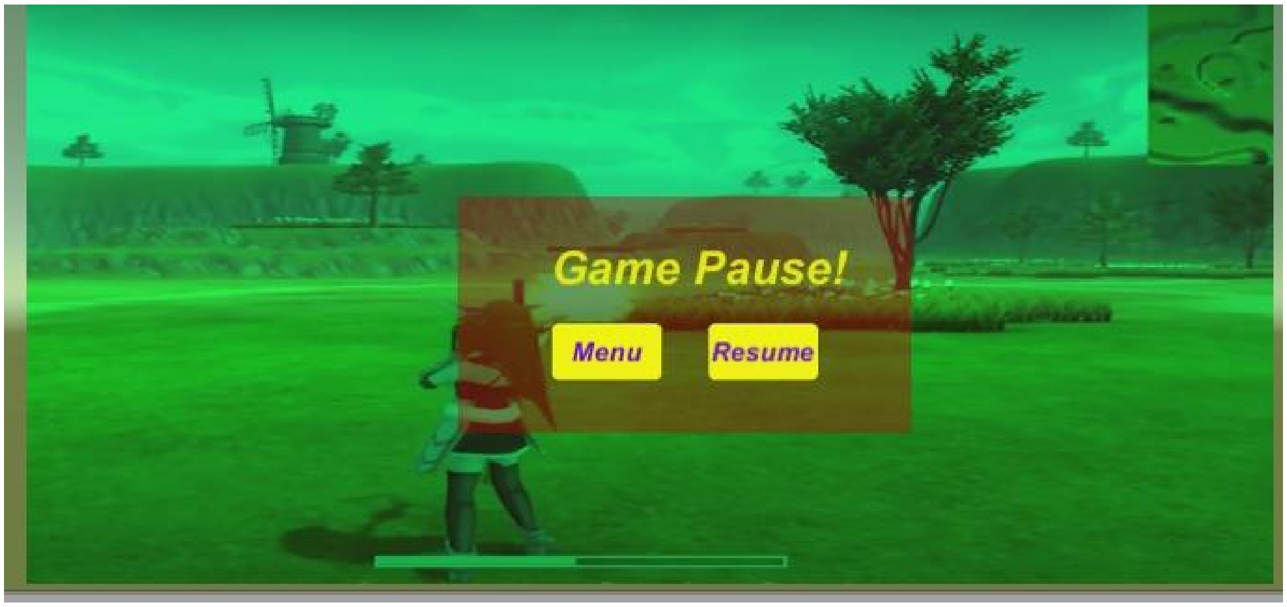


Fig.24 Game pause

Description

The above GUI is for the display of game pause of a VSA RPG. It includes the Menu and Resume options , for the player to select.

GUI forExit Game

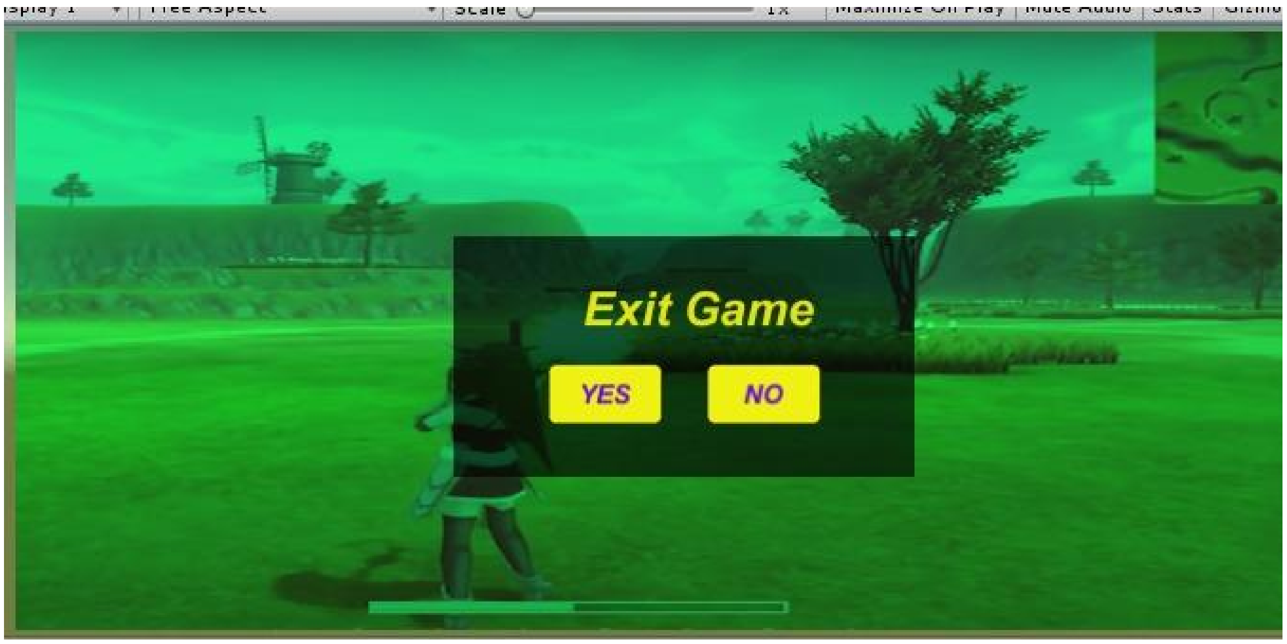


Fig.Exit game

Description

The above GUI is for the exit game of VSA RPG. It includes the Yes and No options , for the confirmation ,whether the player wants to exit the game or not.

Level Selection:

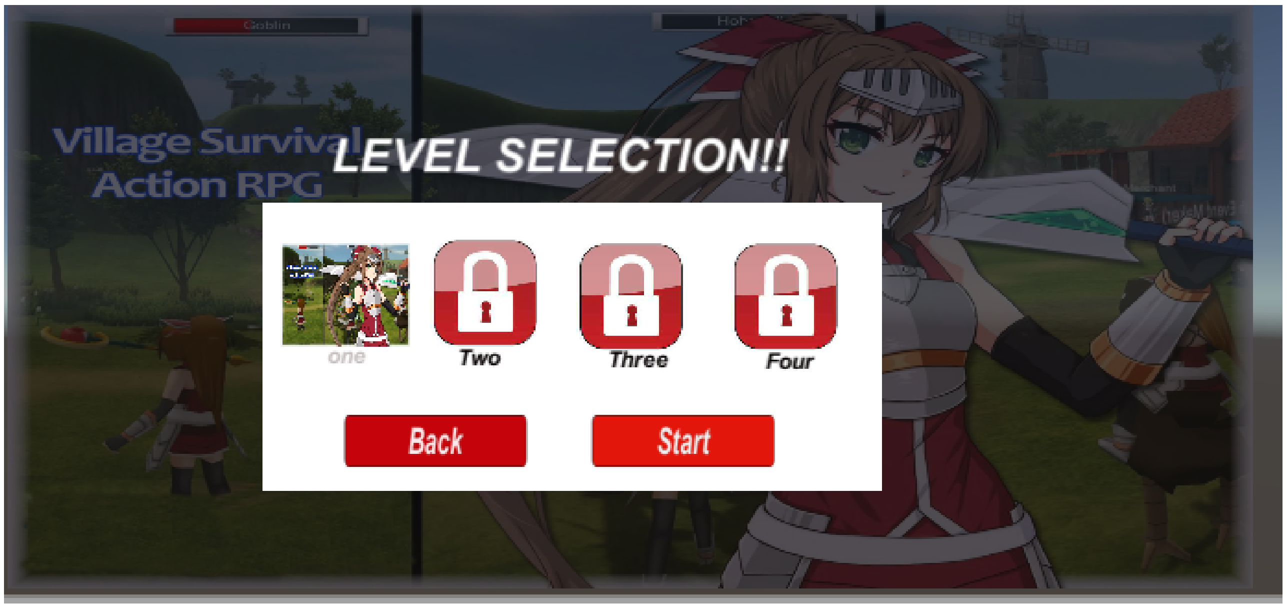


Fig.26 Level selection

Description:

The above GUI is for level selection of VSA RPG. It includes unlock and locked levels. The first level is unlocked and the later ones are locked. To unlock other levels the player has to win the previous level. At the last there are two options for the player whether to go back or

play the level which he/she has already selected above.

GUI For Reset Game:

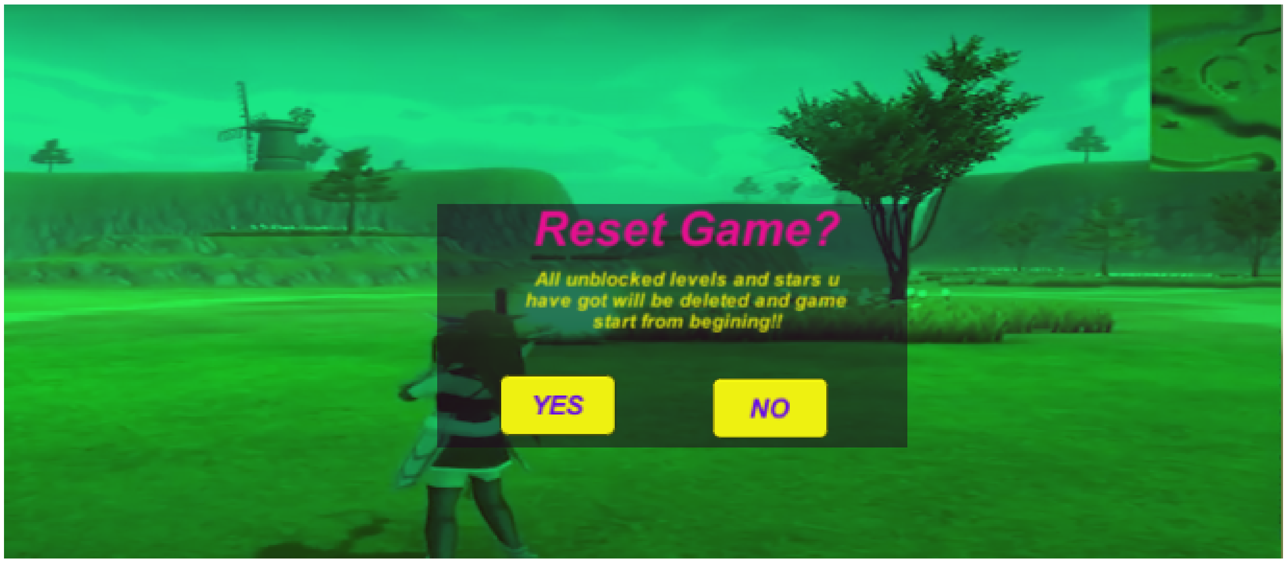

Reset Game?
All unblocked levels and stars u have got will be deleted and game start from begining!!
YES
NO

fig :27 Reset Game

Description:

The above GUI is for reset game. It includes a message for the player,telling him/her that if she/he chooses to reset the game. All the previous work including unlock levels as well as earned stars will be deleted. And the player has to start the game from the beginning. There is confirmation about whether the player wants to reset GE or not by Yes and No options.

References

Ref. No.	***Document Title***	***Date of Release/ Publication***	***Document Source***
1	*Project Proposal*	*12/11/2021*	*https://docs.google.com/document/d/1CjGAoy1X6pakMCEc1p50Uo4LfQ9ZK_j-/edit?usp=sharing&ouid=100876147004553426689&rtpof=true&sd=true*
2	*Software Requirements Specifications*	*24/11/2021*	*https://docs.google.com/document/d/1CjGAoy1X6pakMCEc1p50Uo4LfQ9ZK_j-/edit?usp=sharing&ouid=100876147004553426689&rtpof=true&sd=true*
3	*Software Design Specifications*	*19/1/2022*	*https://docs.google.com/document/d/1CjG*

			Aoy1X6pakMCEc1p50Uo4LfQ9ZK_j-/edit?usp=sharing&ouid=100876147004553426689&rtpof=true&sd=true