Bilkent University
CS 319
Object-Oriented Software Engineering
Fall 2014

Dungeon Escape

ANALYSIS REPORT
October 30th, 2014

GROUP #10
Ahmet Emre Danışman
Ateş Balcı
Buket Depren
Batuhan Berk Yaşar
Ayşe İrem Güner

Contents

1 Introduction

As our group project, we decided to develop a side scrolling-puzzle platformer game called 'Dungeon Escape'. There are many side scrolling and puzzle platformer games in the market with variety of user interfaces and features. Some of these games, such as Super Mario Bros, Donkey Kong and Little Big Planet became legends and are the most successful games of this kind. The aim of these games is to go to a finish point from a starting point in a map while you are dealing with numerous enemies and puzzles. The number and strength of enemies, the tools that player use and difficulty of puzzles vary from game to game depending on games' topic and developers' decision.

2 Proposed System

1.1 Overview

In Dungeon Escape our main char will try to escape from a sci-fi themed prison filled with puzzles and enemies by using some weapons and tools. Puzzles will be solved by pulling some levers, pushing some buttons, jumping on platforms or just simply breaking obstacles to pieces. As you reach the finishing point of a level, you will advance to the next level. If you die on your way to the finishing point, you will restart that level.

1.1.1 Gameplay

The main character will be controlled by both the keyboard and the mouse. The user will move the character, select tools and weapons to use and interact with objects on the map via the keyboard. Mouse will be used to use tools, shift the camera view and use the menu.

1.1.2 Content

The game will consist of a number of different levels that includes many platforms that main char will go through. Design and atmosphere of the levels will be related to the story. While main char is advancing through these levels, main char will be using ranged weapons and melee weapons to damage enemies and obstacles as well as using ropes, hooks and teleportation devices to solve puzzles and pass platforms. While main char is doing these, main char also will be fighting enemies that are trying to stop him.

1.2 Functional Requirements

- Register/Login: Progression for users are separated in order to enable multiple players to play the game from the same machine without save file overlaps.

- Play: Opens a screen with a list of levels to choose from.

- Level Select: Shows a list of different levels which loads a stored list of GameElements to the World.

- In-game menu: A menu which pauses the ongoing game when opened. It contains a resume button to resume the game and an exit button to return to main menu.

- Options: Opens a screen where the user can set display and sound (example: screen resolution).

- Exit: Exits the game.

1.3 Non-functional Requirements

- Performance: Game should be running with a high and stable frame rate in order to give the user a better real time platformer experience. 60 FPS is a good target frame rate in order to provide this experience.

- Accuracy: Although the game is not exactly a physics simulation, physics should still be moderately accurate in order to give the user a better experience. A more accurate physics algorithm would definitely increase the player's satisfaction during gameplay.

1.4   System Models

1.1.1   Scenarios

Login

Use Case Name: Login

Primary Actor: Player

Stakeholders and Interests:

User wants to get access to the game main menu to play.

- System checks whether the user has valid username & password.
- If the user has valid username & password combination, system allows user to play the game.
- If user doesn't have valid username & password, system displays an error message and gives advice to go to register section and create and account.

Pre-condition: User should have a valid username & password combination.

Post-condition: If the process is successful the access to the main menu is provided.

Entry Condition: User starts the program.

Exit Condition: Player selects "Exit" to close the game.

Success Scenario Event Flow

1. User starts the game.
2. Enters valid username & password if s/he has.
3. Gets access to main menu.

Alternative Flows

If the user doesn't have valid username & password register process will be required.

Use Case Name: Play Game

Primary Actor: Player

Stakeholders and Interests:

Player aims to complete all levels.

System allows user to save the current situation of the user.

Pre-condition: At the beginning the game settings are set as default. If the user wants to use different game items, background etc., s/he may change them from the change settings option. After changing the settings, the adjusted settings will be applied to the game.

Entry Condition: Player selects "Play Game" button from Main Menu.

Exit Condition: Player selects "Return to Main Menu" from Pause Menu or closes the game window.

Success Scenario Event Flow

1. Game is started after the Play Game option has been chosen.
2. Player starts form the first level.
3. Player plays until the finish point of the first level is reached.
4. If the user completes the level, system allows gives access to next level.
5. User can collect different items through the game. (Pistol, sword etc.)
6. System keeps the record of collected items.

7. MainChar in the game has 100 health at the beginning and may lose some of them during the fight with enemy characters.

8. User plays until the last level is done or the health of the mainChar reaches zero.

Use Case Name: Choose Weapon

Primary Actor: Player

Stakeholders and Interests:

- System provides a weapon at the beginning of the game.
- User can change the weapon before s/he starts to play.

Pre-condition: The weapon set as default item. (Rope)

Post-condition: Game settings are updated.

Entry Condition: User selects the choose weapon option from the main menu.

Exit Condition: Player selects "exit" to go back to main menu.

Use Case Name: Change Settings

Primary Actor: Player

Stakeholders and Interests:

-User wants to change game settings: changing the background, changing the sound options etc.

-The game will be updated once the user is done.

Pre-condition: The game settings are set as default.

Post-condition: Game settings are updated.

Entry Condition: User choses "Change Settings" from main menu.

Exit Condition: User choses **Exit** to return to main menu.

Success Scenario Event Flow:

1. User choses the Change settings option from the main menu.

1. Game settings will be displayed to user on the screen.

2. User may or may not change the displayed settings.

3. Any changes will be applied to the game when the user is done.

Help

Use Case Name: Help

Primary Actor: Player

Stakeholders and Interests:

Player is willing to learn the instructions of the game, level flow, items in the game.

Player wants to learn the basic information & FAQs about the game.

Pre-conditions: Player should be in Main Menu or Pause Menu.

Entry Condition: Player clicks "Help" button from Main Menu or Pause Menu.

Exit Condition: Player selects "Exit" to return previous menu.

Success Scenario Event Flow:

- User clicks the Help button in the main / pause menu.
- System displays the general purpose and the instructions of the game.

View Credits

Use Case Name: View Credits

Primary Actor: Player

Stakeholders and Interests:

Player wants to see the general situation of his/her mainChar.

Player wants to see the weapons that s/he has.

Pre-conditions: Player should be in Main Menu.

**Entry Condition:** Player should have a played game.

Player clicks "View Credits" button from Main Menu.

Exit Condition: Player selects "Exit" to return previous menu.

Success Scenario Event Flow:

- User clicks the View Credits button in the main / pause menu.
- System displays the weapons & health of the mainChar.
- System displays the level results of the player.

1.1.2 Use-Case Model
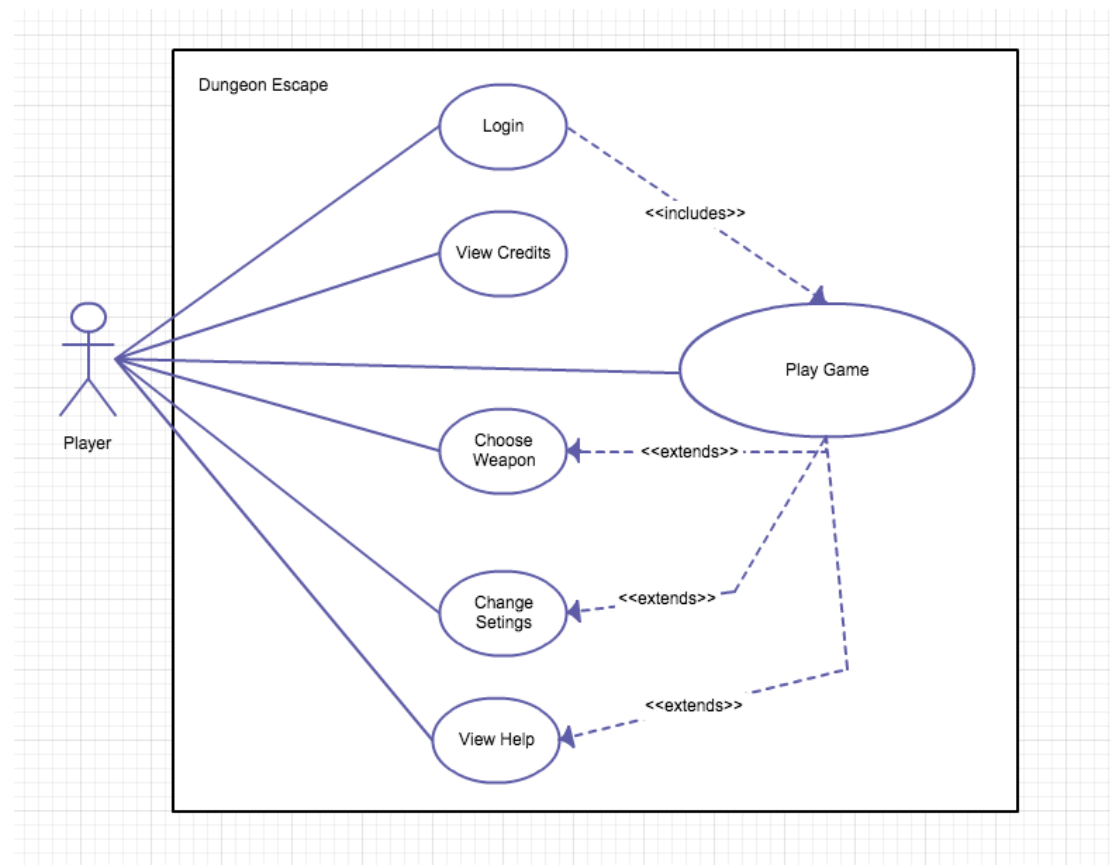
Figure: A use case diagram of the UI

1.1.3 Object and Class Model

GameElement

<<Property>> #width : int
<<Property>> #height : int
<<Property>> #x : double
<<Property>> #y : double
<<Property>> #verticalSpeed : double
<<Property>> #horizontalSpeed : double
<<Property>> #weight : double
<<Property>> #elasticity : double
<<Property>> #ground : boolean
<<Property>> #flying : boolean
<<Property>> #smooth : boolean
<<Property>> #active : boolean
<<Property>> #jumping : boolean
<<Property>> #fricted : boolean
#left : boolean
#right : boolean
#jump : boolean
<<Property>> #fixed : boolean
+GameElement(x : double, y : double)
+getRectangle() : Rectangle2D
+intersects(e : GameElement) : boolean
+contact(direction : String, e : GameElement) : void
+action(d : double) : void
+right(b : boolean) : void
+left(b : boolean) : void
+jump(b : boolean) : void
+moveX(xChange : double) : void
+moveY(yChange : double) : void
+draw(g : Graphics, camera : Point) : void
+setVerticalSpeed(verticalSpeed : int) : void
+setJumpHeight(jumpHeight : int) : void

StaticElement

+StaticElement()
+obstruction(direction : String, e : GameElement) : void

MoveableObject

+MoveableObject(x : int, y : int)
+draw(g : Graphics, camera : Point) : void
+obstruction(direction : String, e : GameElement) : void

Player

+Player(x : double, y : double)
+action(d : double) : void
+left(b : boolean) : void
+right(b : boolean) : void
+draw(g : Graphics, camera : Point) : void
+getCenter() : Point

Obstacle

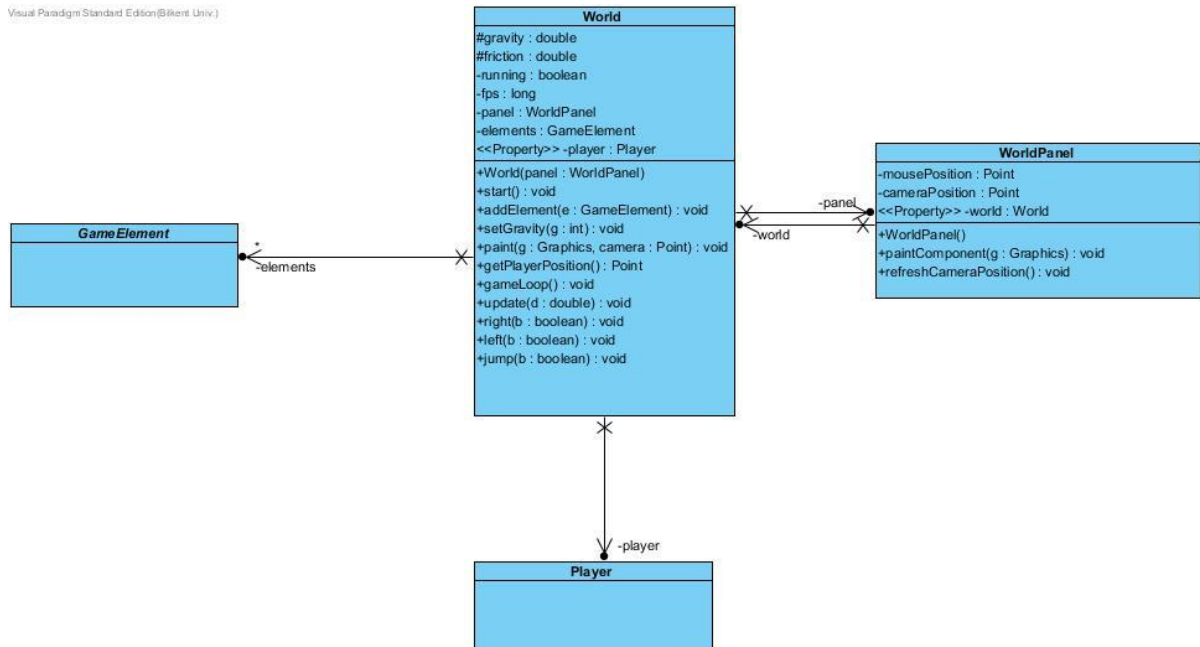| |
|---|
| +Obstacle(x : int, y : int, w : int, h : int) |
| +draw(g : Graphics, camera : Point) : void |

Figure: Class Diagram of GameElement class

Figure: Class Diagram of the UI
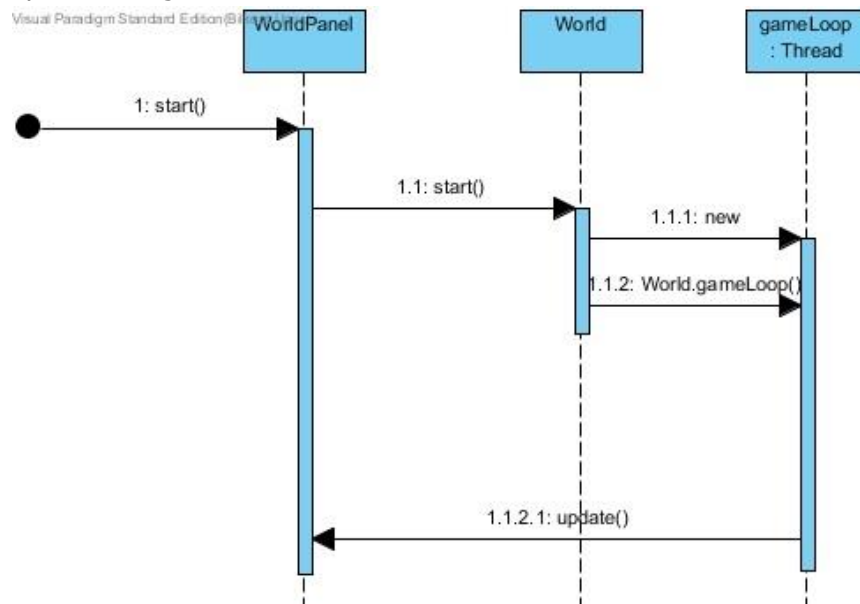
1.1.4 Sequence Diagram

Figure: Sequence Diagram of startup of the game

Note: gameLoop Thread is looped and "update" method of WorldPanel is called until the game or level is over (update method is inherited from JPanel to refresh the paint component).
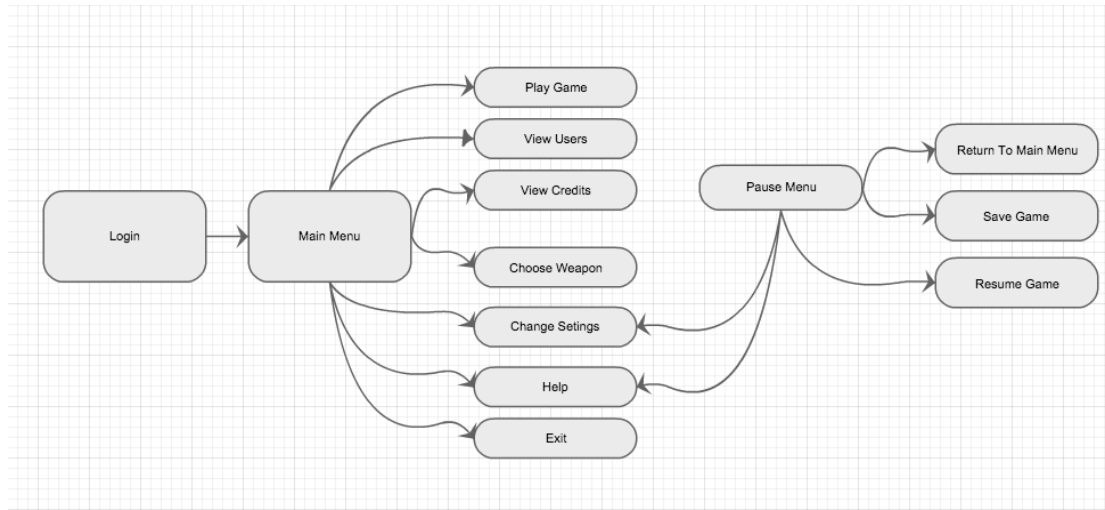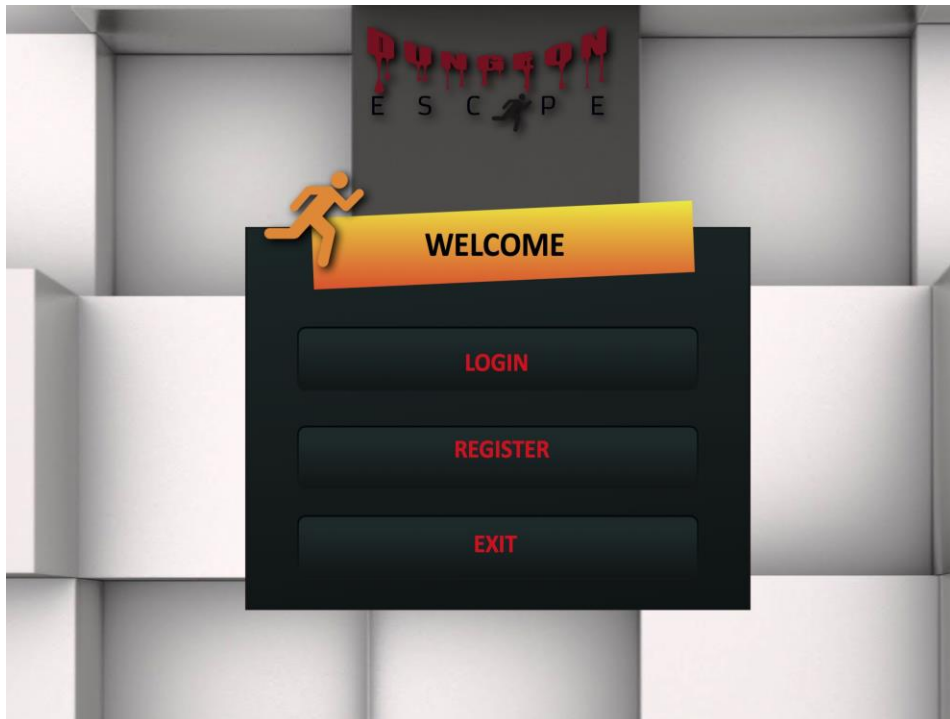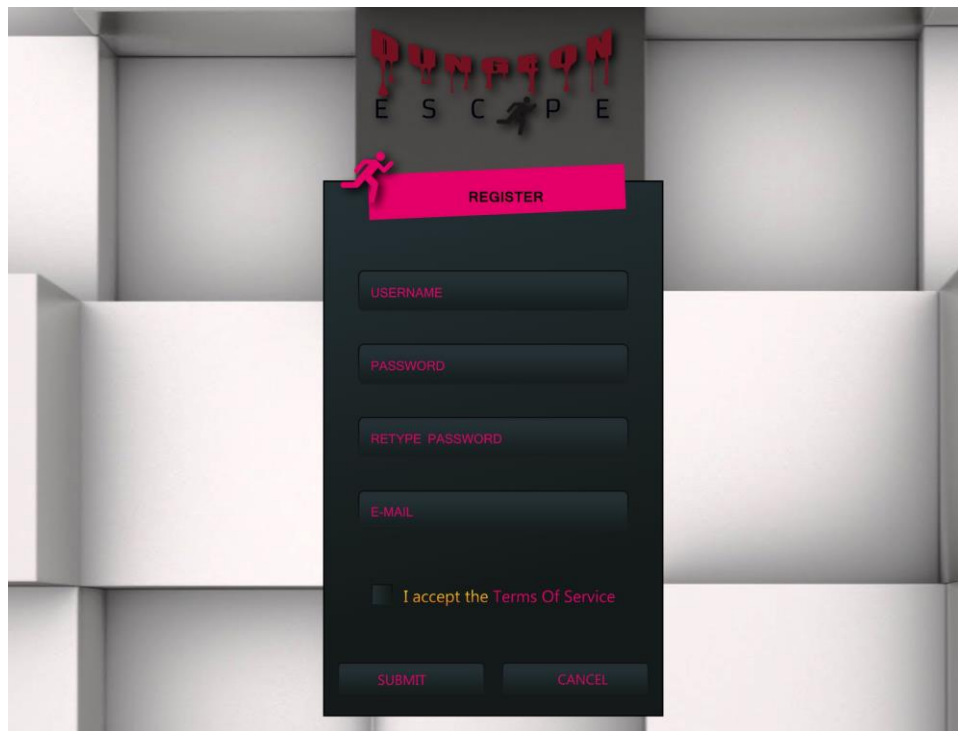
1.1.5 User Interface

Figure: Navigational Path

Screen Mock-ups:

Welcome Menu:

When the player starts the game, s/he will see the screen that is shown below. Users should login in order to get access to the game. Users either can login if they have an account, or they can create an account by using register section.
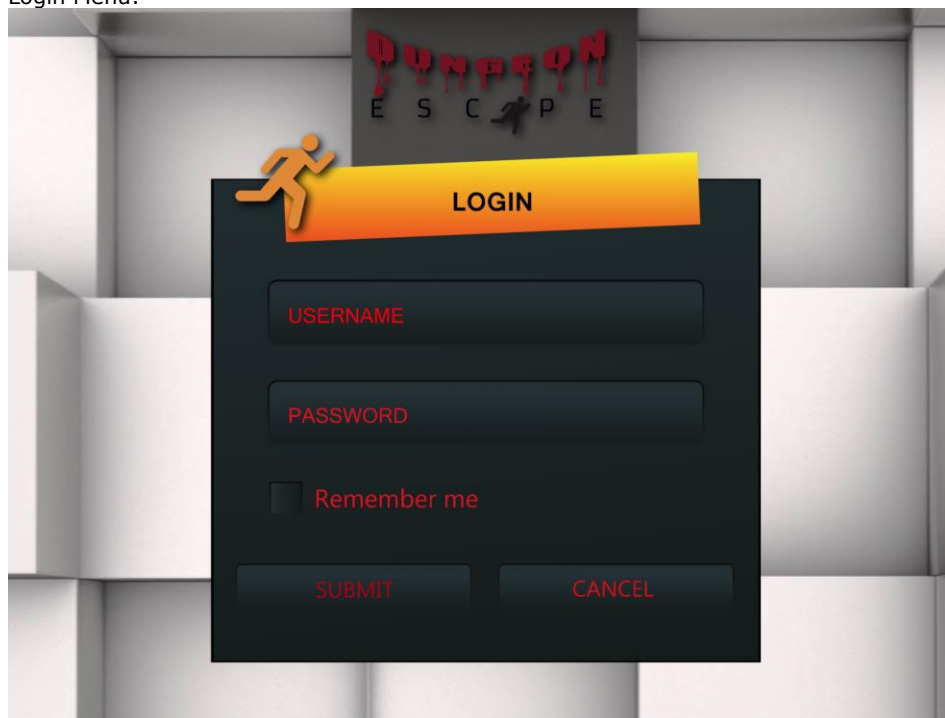
Register Menu:

If the users don't have an existing account in the system, they should create an account by using register menu. Registration operations require Username, Password, Re-Password and e-mail to create an account.

Login Menu:

If the user has an account, s/he should login to have an access to game. The login screen will be appearing when the login section is chosen. Users should type their valid username & password to login to system.

Main Menu:

Main Menu contains the all the features of the game. Users can;

Start playing game by the "Play Game" option. The system will direct user to the saved games panel to ask if they want to continue their saved game if there are any.
Click View Users button which allows them to see each other current situation in the system, their weapons etc.
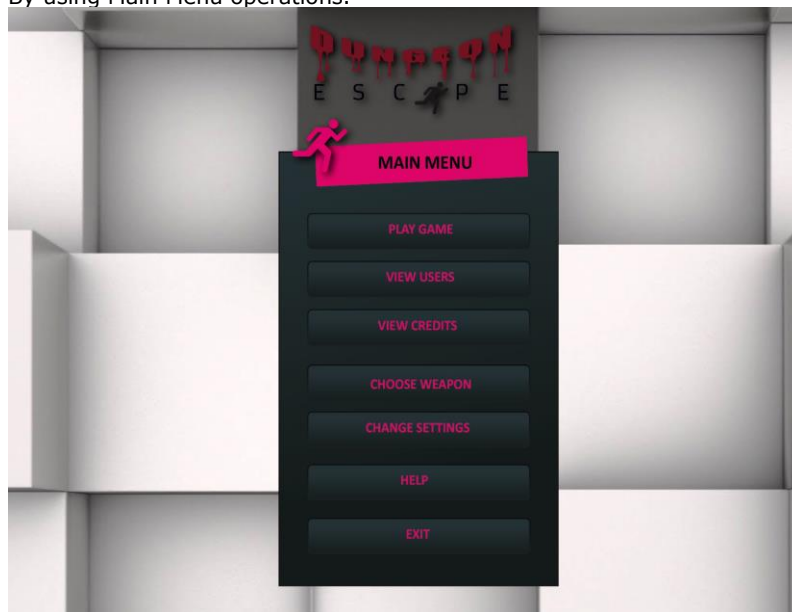Click View Credits button to see the information about his/her mainChar's situation & results.
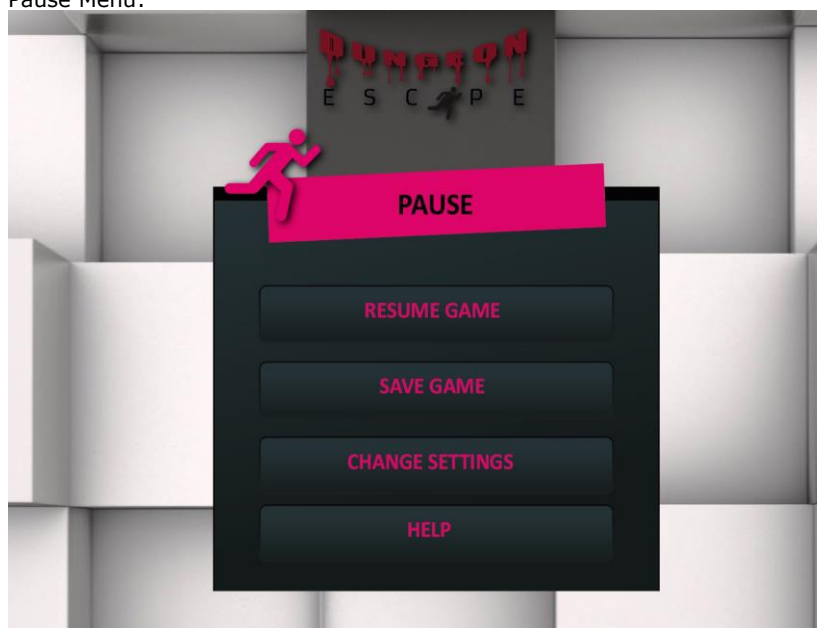Change setting from the main menu.
Get help instructions.
Exit from the game

By using Main Menu operations.

Pause Menu:

Pause menu is a menu that allows user to change the current settings of the game. Additionally, users can save the current situation of the game by using this pause menu. If the user wants to learn

something about the game; the help option, which contains everything about the game, is also provided in the pause menu.

Items

 Rope

The rope is a weapon that is provided to the user at the beginning of the game. It is the default weapon of the mainChar.

 Sword

Through the game there will be prices in different parts. Sword is one of the item that user can get.

 Pistol

Pistol is a powerful weapon in the game, which can be found in the upper levels.

 Health Pack

Health pack is a power up which increases the mainChar's health.

3 Glossary

- FPS: Frames per Second is a term used to show how many times the screen is refreshed in a game in order to describe its performance.

4 References

- "OMG UML Resource Page - Unified Modeling Language (UML)." Web. 30 Oct. 2014.