**CHAPTER 5**

**THE CODETEST - A WEB BASED RPG**

**System Overview**

The CodeTest - A Web Based RPG is an online role playing game which has the same mechanics and capabilities of Pokemon games. The game serves as a learning tool and at the same time, making gamers enjoy themselves fully to the game’s capabilities as such the game also includes Pokemon games popular combat system in which there will be NPC’s scattered all over the region or simply in each levels of the said region. For instance if the player is found by an NPC, both will enter the combat system. The player will be forced to complete the NPC’s task. The task consists of making the player fulfill the task given by the NPC in which he/she has to solve the problem of the NPC in a way of coding. The game has its embedded programming, in which the player can enter lines of codes so that the player can get pass through the NPC, or simply win the given task. Given that if the player failed to complete the task or shall we say, he/she does not have any idea how to code the given task, hints will be there but there will be a decrease of the total score after completing the task.

The game itself of course has a story so that the player won’t get bored by simply battling NPC’s, fulfilling their tasks. Of course every story has its antagonists or villains, or evil organizations that are hell bent on taking over the world.

**System Objectives**

The study aims to develop a web based game that will help the students or users learn programming while enjoying.

It specifically aims to:

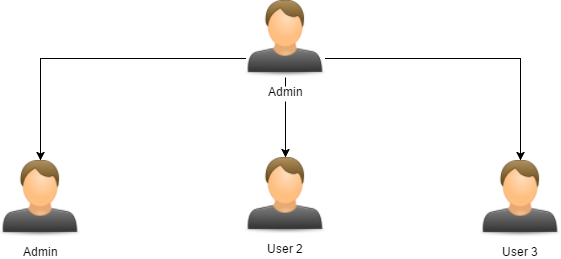
1. develop a role playing game that will test a student’s knowledge in coding.
2. introduce coding challenges to students in an interesting approach.
3. allow students to apply their knowledge in coding and gain more from the game.
4. design a user-friendly and interesting interface for the students.
5. develop a role playing game that will show students their game records, errors made in coding and how they can improve it.

**System Scope and Limitations**

The system focuses on providing edutainment to College of Computer Studies students of LSU-Ozamiz or even netizens. The system will mainly focus on teaching C#, as C# is one of the hardest and time-consuming to learn for a programming language. Another is being able to save their progress of the game, not delete their progress except in creating new game as it will overwrite their old progress. The system will not cover other concerns except the things stated above.

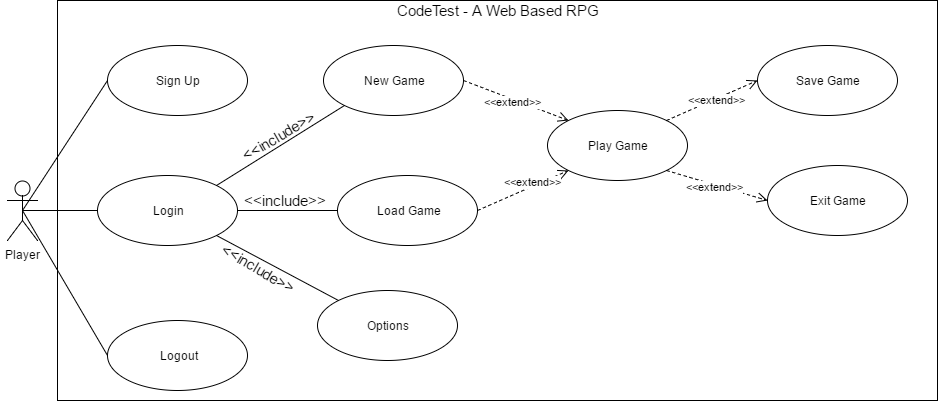
**Architectural Design**

**User Hierarchy**



**Figure 1: User Hierarchy Diagram**

**Use Cases**



**Figure 2: Use Case Diagram**

**Player Use Case List**

**Login**: The player must login his/her account before playing the game

**Sign Up**: The player will need to register if he/she does not have an account yet or want to create another account

**Logout**: The player can log out of their respective accounts

**New Game**: The player will have play a new game if they do not have progress or saved game yet

**Load Game**: The player can load their previous save game in their respective accounts

**Options**: Allow players to adjust graphics, sound or resolution of the game

**Play Game**: The player can also, of course, play the game, whether a new one or their previous saved game

**Save Game**: Whenever they are in a rush or taking measures, they can save the game to prevent losing progress

**End Game**: Allows players to end their game, thus leading to the main menu of the game

**Player Use Case Scenarios**

**Table 1**

**LOGIN**

|  |  |
| --- | --- |
| U1: Login | |
| Pre-Condition: Sign Up | |
| Actor’s Action | System Response |
| 1. Player enters username and password | 1. The system checks if username and password exist |
| Post-Condition: Main Menu appears | |
| Abnormal Path: If the username and password is invalid, the system will ask to re-enter both | |

**Table 2**

**SIGN UP**

|  |  |
| --- | --- |
| U2: Sign Up | |
| Pre-Condition: None | |
| Actor’s Action | System Response |
| 1. The user enters needed information on each field | 1. The system checks if the entered information already existed on the database   3. “Successfully registered. Please click the button to proceed” is displayed |
| Post-Condition: Login scene | |
| Abnormal Path: If the username and password already exists, the system will ask to enter a new one | |

**Table 3**

**LOGOUT**

|  |  |
| --- | --- |
| U3: Logout | |
| Pre-Condition: Login | |
| Actor’s Action | System Response |
| 1. The user clicks “Logout” button if he/she wishes to log out of his/her account | 1. The system logs out the user |
| Post-Condition: Login scene | |
| Abnormal Path: If connection is lost, the player can’t do a thing if he/she keeps on clicking the logout button | |

**Table 4**

**NEW GAME**

|  |  |
| --- | --- |
| U4: New Game | |
| Pre-Condition: Login | |
| Actor’s Action | System Response |
| 1. The user clicks “New Game” button if he/she is new to the game or wishing to create a new one to replace the old one 2. The player chooses gender for his/her character for it to be sent to the game world | 1. Prologue scene will then proceed |
| Post-Condition: Character selection screen | |
| Abnormal Path: If connection is lost, since it automatically saves the very first minute of the game, the player can’t do a thing until the connection comes back | |

**Table 5**

**LOAD GAME**

|  |  |
| --- | --- |
| U5: Load Game | |
| Pre-Condition: Login | |
| Actor’s Action | System Response |
| 1. The user clicks “Load Game” button if he/she wishes to continue the game | 1. The system acquires the current account’s saved file and loads it 2. Save file’s progress is successfully loaded |
| Post-Condition: Saved game loads, displays the current coordinates of the player’s character or simply where he/she last saved | |
| Abnormal Path: The saved file’s corrupted as a result from losing connection while the game is saving | |

**Table 6**

**PLAY GAME**

|  |  |
| --- | --- |
| U6: Play Game | |
| Pre-Condition: New Game or Load Game | |
| Actor’s Action | System Response |
| 1. The user roams around the level map until he/she encounters an NPC 2. The user enters the correct code, according to what task the NPC gave to him/her | 1. NPC automatically interacts to the player if found 2. Combat is initiated, the player will have to do what the NPC gave 3. An embedded in-game compiler is displayed for the code of the user to be entered 4. The compiler checks if it displays the same result as what the NPC gave |
| Post-Condition: None | |
| Abnormal Path: None | |

**Table 7**

**OPTIONS**

|  |  |
| --- | --- |
| U7: Options | |
| Pre-Condition: New Game or Load Game | |
| Actor’s Action | System Response |
| 1. Clicks the “gear” button on the side of the screen 2. Clicks “Option” button 3. Adjusts to what he/she wants to have a better gameplay | 1. System responds to the user’s option changes till he/she resets it back |
| Post-Condition: | |
| Abnormal Path: None | |

**Table 8**

**SAVE GAME**

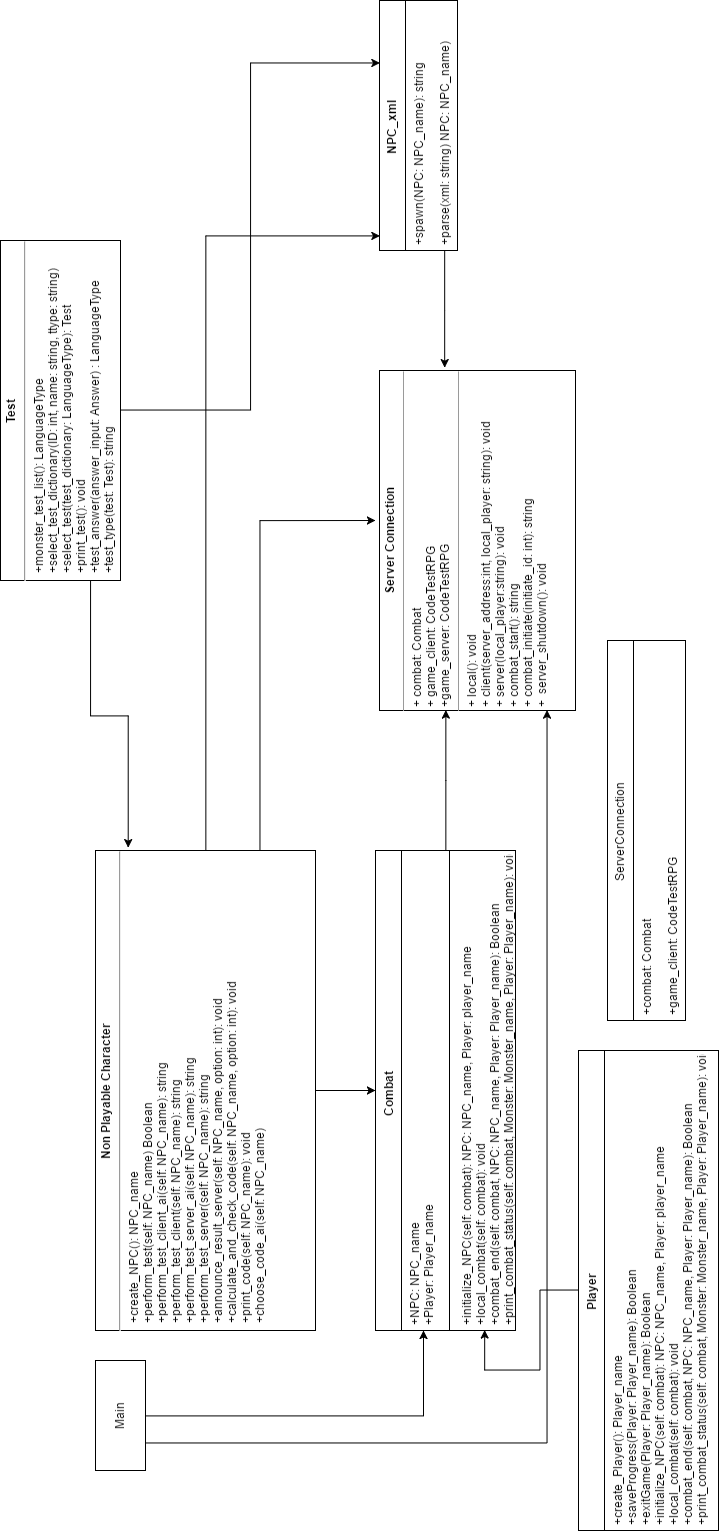
|  |  |
| --- | --- |
| U8: Save Game | |
| Pre-Condition: New Game or Load Game | |
| Actor’s Action | System Response |
| 1. Clicks the “gear” button on the side of the screen 2. Clicks “Save Game” button | 1. NPC automatically interacts to the player if found 2. System saves the user’s progress 3. Stores save file into the database 4. System displays “Successfully saved” |
| Post-Condition: | |
| Abnormal Path: The saved file have a chance of being corrupted as a result from losing connection while the game is saving | |

**Table 8**

**EXIT**

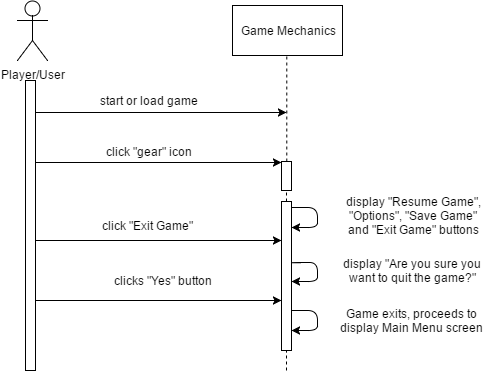
|  |  |
| --- | --- |
| U9: EXIT | |
| Pre-Condition: New Game or Load Game | |
| Actor’s Action | System Response |
| 1. Clicks the “gear” button on the side of the screen 2. Clicks “Exit” button | 1. System displays “Make sure you have saved quitting to avoid loss of progress” over “Are you sure you want to quit the game?” 2. System proceeds to Main Menu |
| Post-Condition: | |
| Abnormal Path: None | |

**Class Diagram**

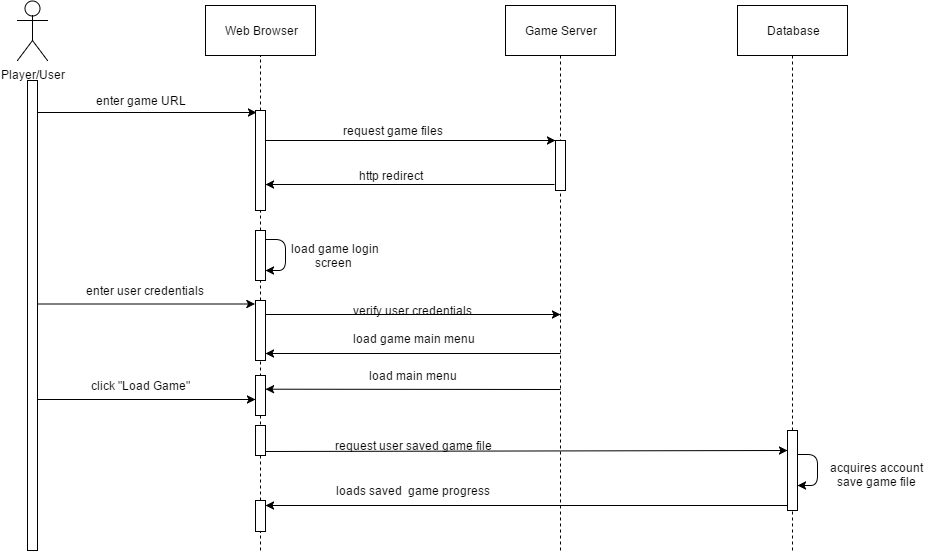
****

**Figure 1: Class Diagram**

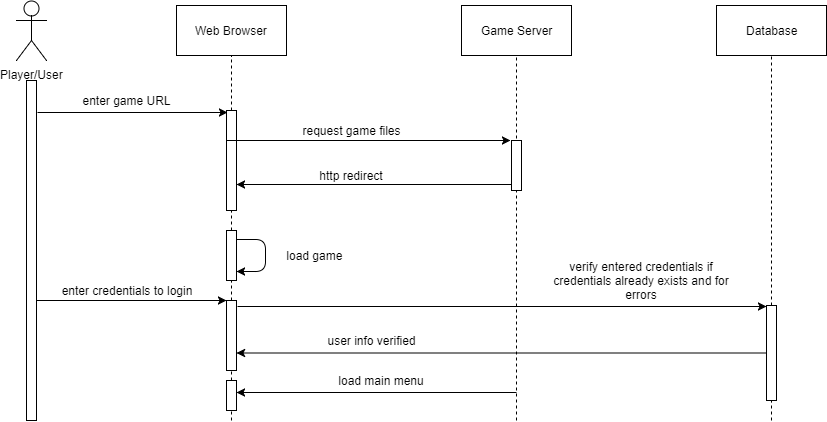
**Sequence Diagrams**



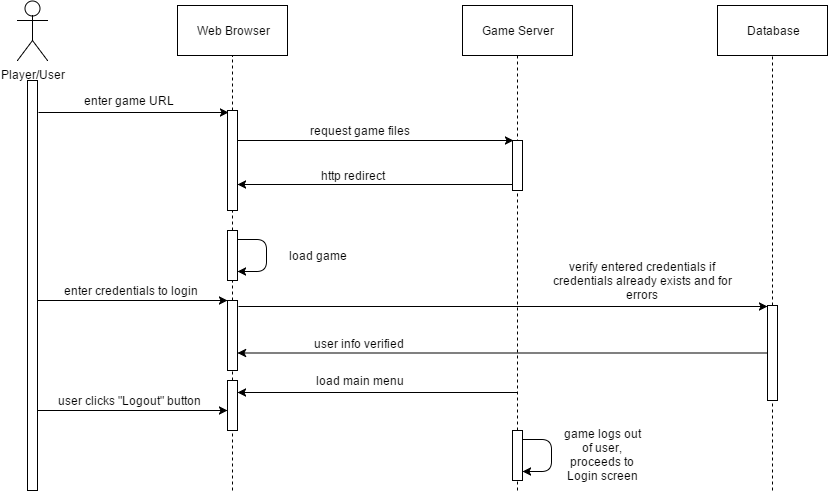
**Figure 1: Exit Game Sequence Diagram**

****

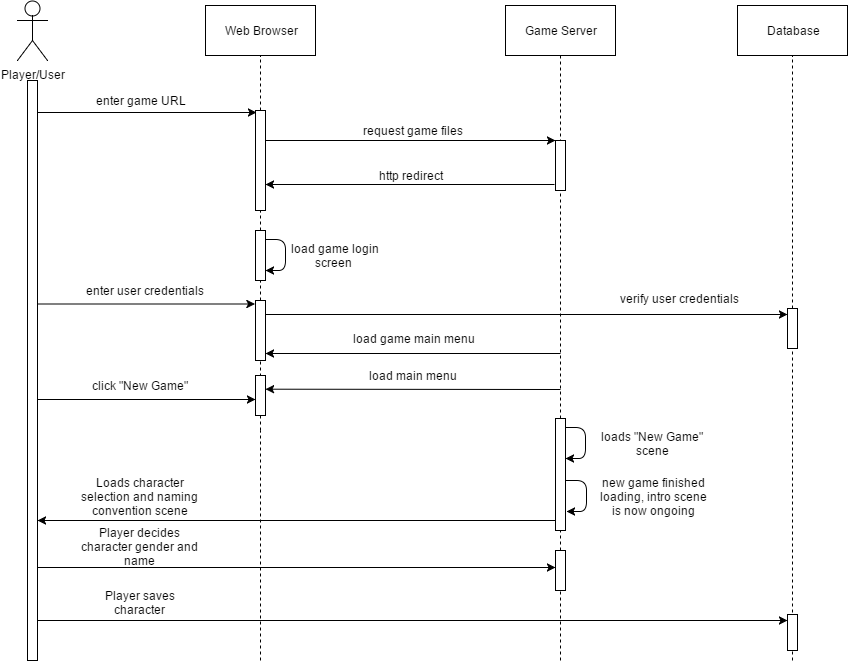
**Figure 2: Load Game Sequence Diagram**



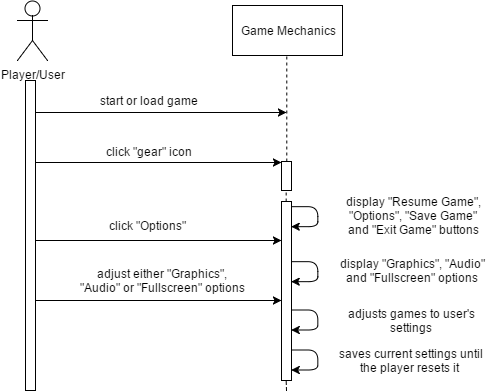
**Figure 3: Login Sequence Diagram**

****

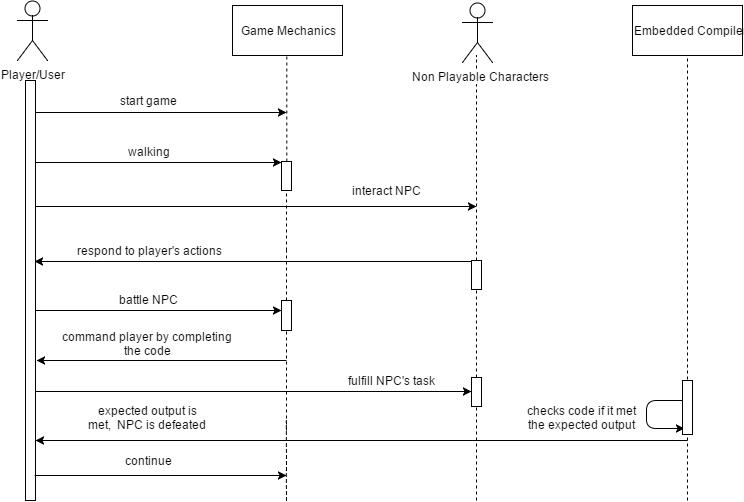
**Figure 4: Logout Sequence Diagram**



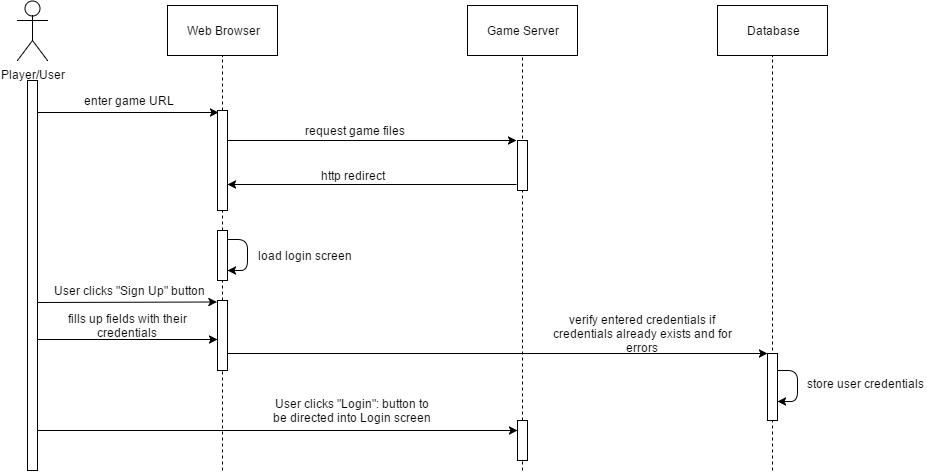
**Figure 5: New Game Sequence Diagram**

****

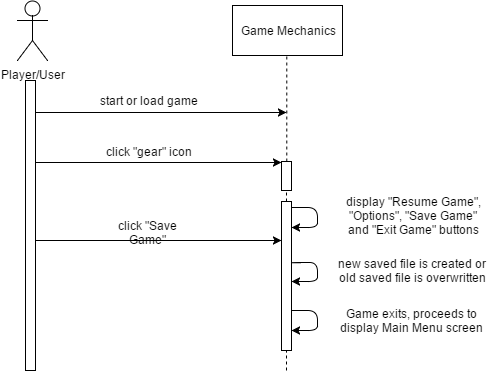
**Figure 6: Options Sequence Diagram**

****

**Figure 7: Play Game Sequence Diagram**

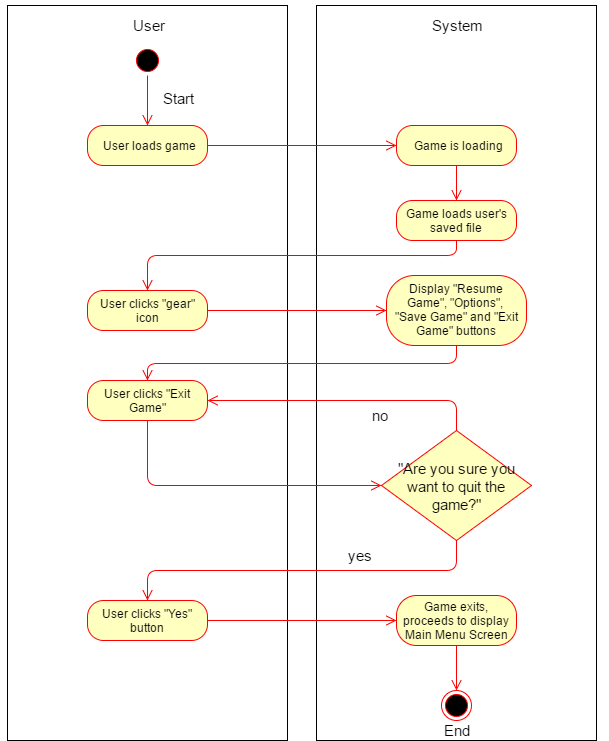
****

**Figure 8: Register Sequence Diagram**

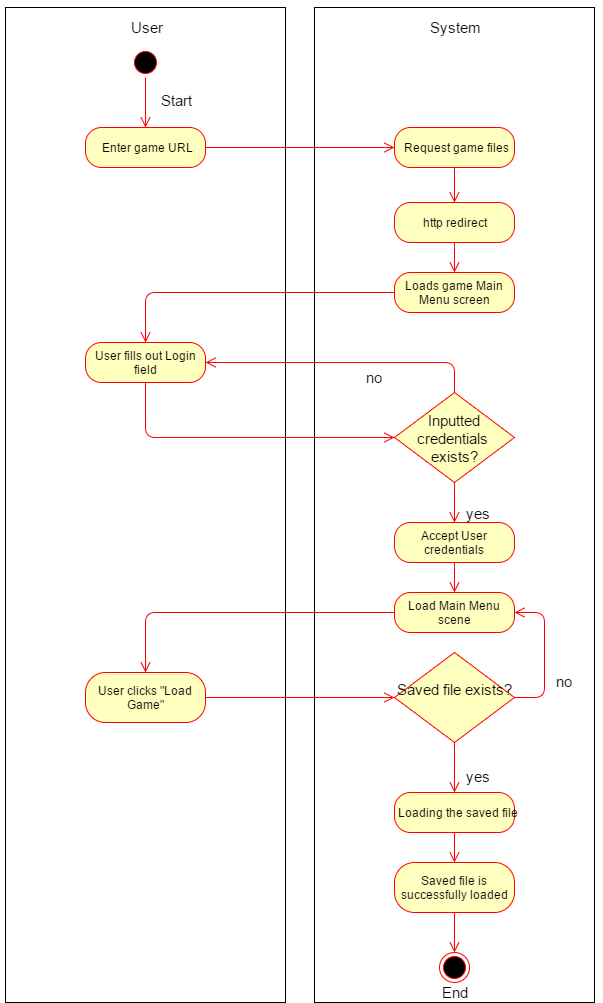
****

**Figure 9: Save Game Sequence Diagram**

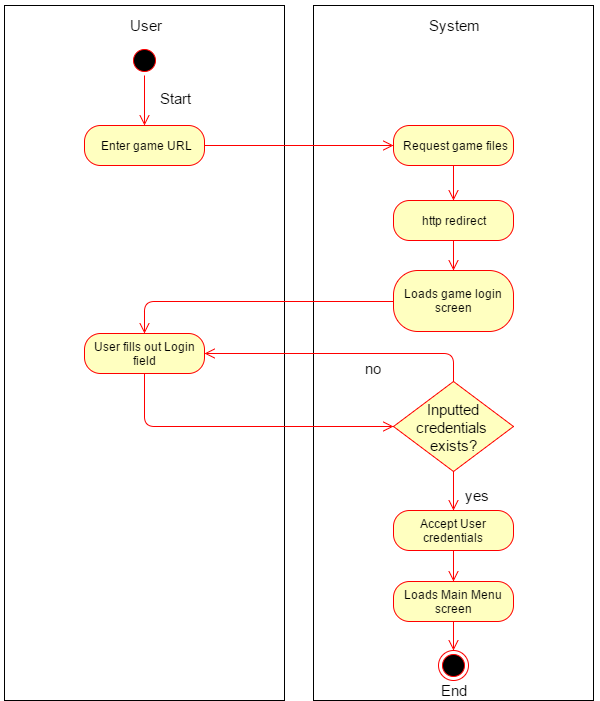
**Activity Diagrams**

****

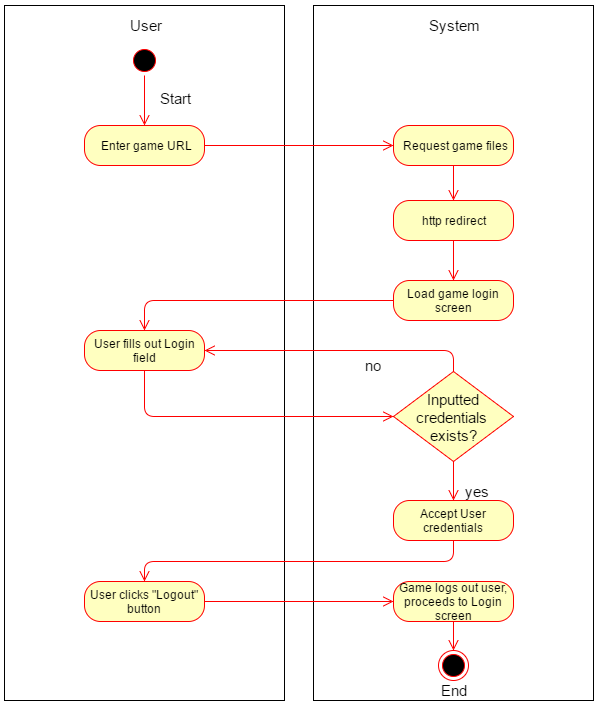
**Figure 1: Exit Game Activity Diagram**

****

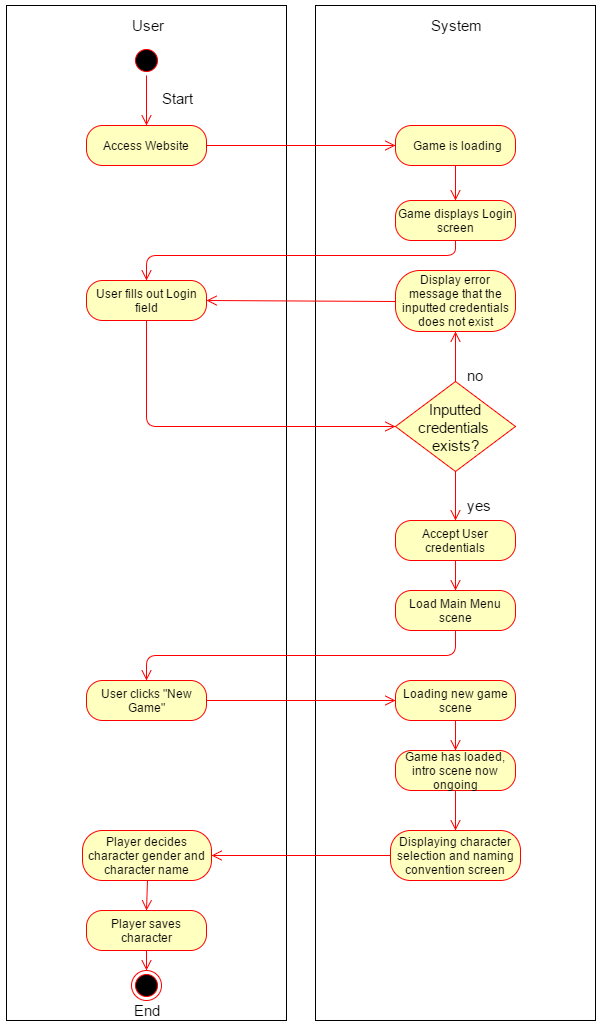
**Figure 2: Load Game Activity Diagram**

****

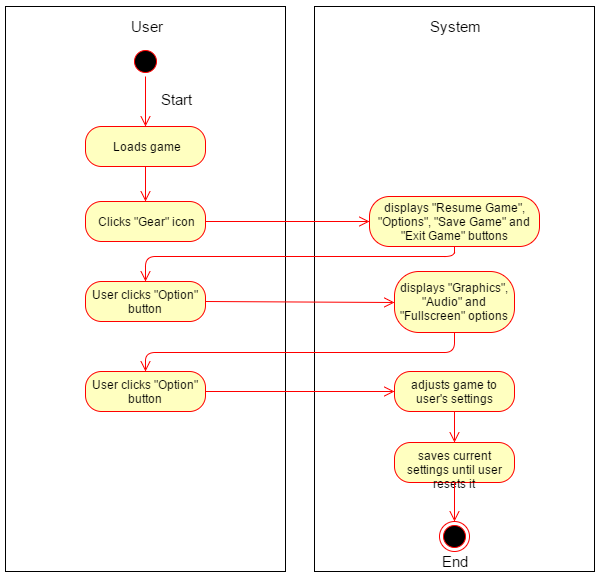
**Figure 3: Login Activity Diagram**

****

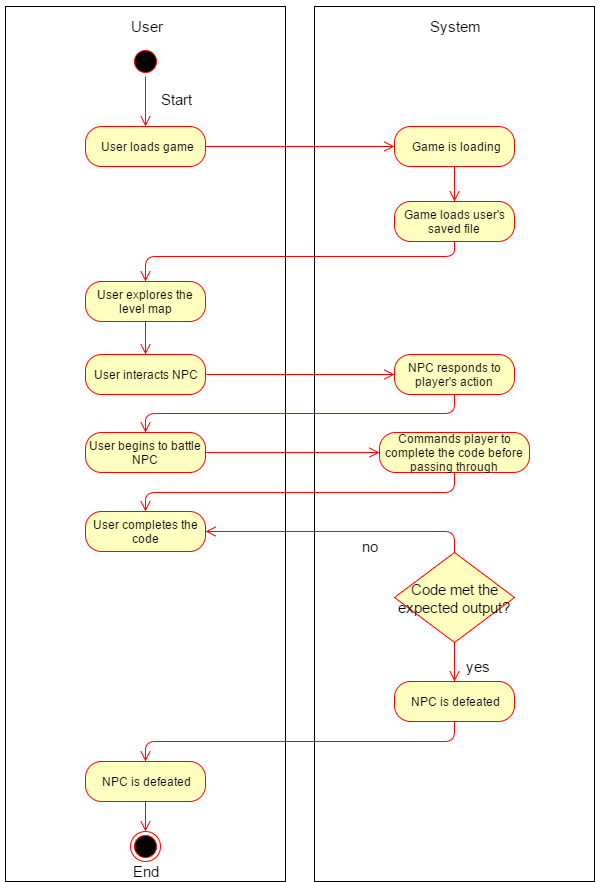
**Figure 4: Logout Activity Diagram**

****

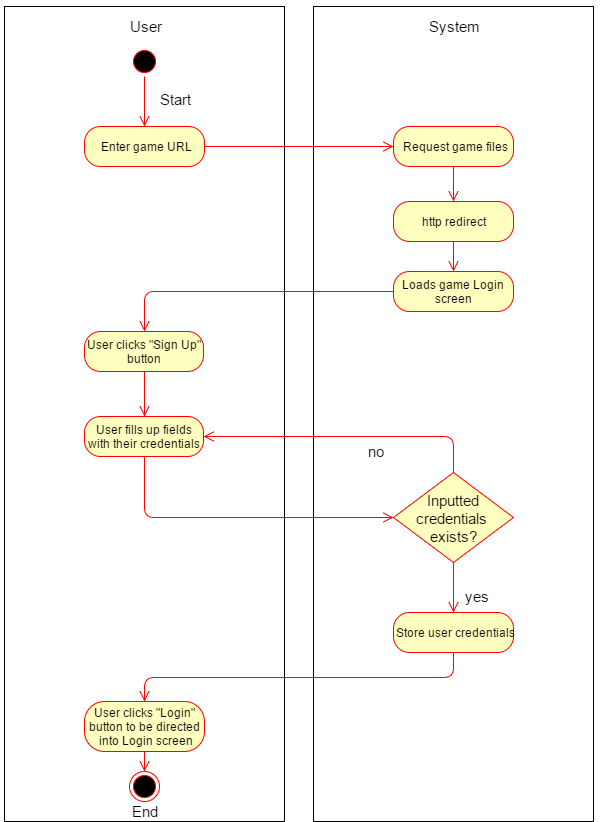
**Figure 5: New Game Activity Diagram**

****

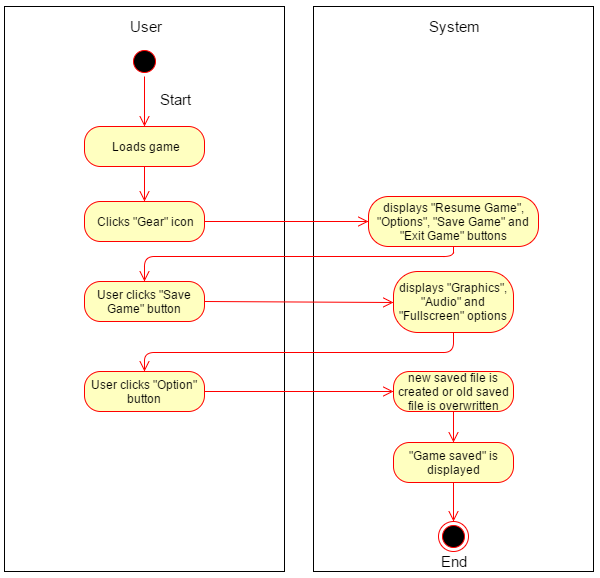
**Figure 6: Options Activity Diagram**



**Figure 7: Play Game Activity Diagram**



**Figure 8: Register Activity Diagram**

****

**Figure 9: Save Game Activity Diagram**

**System Function**

List of functions that must be performed by the system:

**Login**

This function is one of the prerequisite in order for the game to be played. A player must have an account. If ever the player does not have an account, there will be a button below the login fields.

**Sign Up**

This function is also a prerequisite before playing the game, as such the player must log on in using their accounts first before creating a new game or loading their saved game. The field consists of basic information which are username and password. Email will not be included as such skills of ours are not that broad to implement a tighter security. Account managing like these one is also prone for attacks, rendering the system’s security useless.

**Logout**

Logout is a basic function where the user logs out of their respective accounts. After logging out, they will be directed into the log in screen of the web-based game.

**New Game**

The new game functions serves as more like of a “new life” or “rebirth” for the players. If the player has already a save game stored in the database, and wishes to create a new game, the old save game file will be replace with the new one as soon the player saves using the Save Game button.

**Load Game**

Load Game function basically continues the progress of the player if he/she has saved the game. If no save file is found, the Load Game will be unavailable in the menu screen, or simply the user cannot click on the Load Game button.

**Options**

Options function simply tells the user that they can adjust the settings to their heart’s content or in what adjustments they are comfortable with, as such it wont affect the gameplay itself. Graphics, Audio and Game Resolution are the main options in which the users can adjust. There will be Low, Medium and High for Graphics, as they will experience lag if the game’s graphics is not adjusted for a low-end PC. As for the Sound, they can adjust the game’s volume if they are annoyed with the background music or they just don’t like it. Lastly, Game Resolution is consisted of an only option, which is Fullscreen but in a checkbox form. Fullscreen makes the browser not visible to the eyes, but instead focuses on the game.

**Play Game**

Playing the game, is the main purpose why the game exists. Players explores the region, going through each levels, battling NPC’s by making the fulfill their tasks. In order to get through the encountered, the player must code according to what the NPC wants. The player does not need to worry about debugging, as it will be included in the embedded compiler inside the game. They can check the code if it has bugs, before proceeding to run and match the code to the NPC’s task. If it is wrong, the player will have to do it again, same task but the total score is deducted if the second or third try is right. Buildings, puddles of water, rivers, landmarks will be in the game to avoid simplicity of the game, making the maps more appealing to the eyes of the players and playable.

**Save Game**

Save Game is one of the most important function on the game. It saves the game every time the player feels like they are making progress and want to continue the game later, by just clicking a button. There are instances the player forgets to save the game, even though they have progressed so much in the game, as it will be useless if they exited the game, coming back to play the game and finding out that the progress they worked so hard, doesn’t exist.

**Exit Game**

Exit Game is a function in which the player exits the game in which can be accessed through a gear icon. If the player does not have the feeling to play the game, maybe tired or simply boring for them, Exit Game button exists. But the players should save the game first before leaving the game, as such exiting the game wont automatically save the game like what other video games do.

**Physical Environment and Resources**

These are the system specifications used by the proponents in developing the research:

**Developer Side System/Application**

**Hardware**

**Platforms to be used:**

**RAM**: 4.00 GB

**Hard Disk Drive**: 1000 GB

**Processor**: Intel Core i5 4th Gen and up processor chips

**Graphics Card**: Graphics card with DX9 (shader model 3.0) or DX11 with feature level 9.3 capabilities.

**Software**

**Platforms to be used:**

**OS**: Windows 7 SP1+, 8, 10, 64-bit versions only

**Platform**: Unity, MySQL Workbench, Tiled, Tiled2Unity

**Language**: C#, Lua, SQL

**Deployment Specification**

**Recommended System Requirements**

**Desktop Computer:**

**Computer OS**: Windows 7 or later

**RAM**: 2.00 GB

**Browser**: Any recent desktop version of Firefox, Chrome, Edge or Safari.

**Calendar of Activities**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ACTIVITY** | **AUG** | **SEP** | **OCT** | **NOV** |
| **Data Gathering** | **\*\*\*\*\*** | **\*\*** |  |  |
| **Software Requirements Analysis** |  | **\*\*\*\*** | **\*\*\*\*** | **\*\*\*\*** |
| **Initial Architectural Design** |  |  |  | **\*\*\*** |