(Associate Professor)

ASAD BASHIR
01-134131-012
AYESHA ASGHAR
01-134131-104

A Game Base Learning Environment for Programming Logic Building

Bachelor of Science in Computer Science

Supervisor: Mrs. Saima Jawad

Department of Computer Science
Bahria University, Islamabad

October 2016

Certificate

We accept the work contained in the report titled "A Game Base Learning Environment for Programming Concept Building ", written by Asad Bashir and Ayesha Asghar as a confirmation to the required standard for the partial fulfillment of the degree of Bachelor of Science in Computer Science.

Approved by . . . :

Supervisor: Mrs. Saima Jawad (Associate Professor)

_____

Internal Examiner: Name of the Internal Examiner (Title)

_____

External Examiner: Name of the External Examiner (Title)

_____

Project Coordinator: Dr. Arif ur Rahman (Assistant Professor)

_____

Head of the Department: Dr. Faisal Bashir (Associate Professor)

_____

September 31st, 2016

Abstract

Students can learn a variety of skills through game-based learning, which appears to enhance problem solving, critical thinking, and creativity for users. Games are changing in order to help students learn more easily. The project is basically a game based learning environment for logic building. Most of the students learn the concepts very easily but learning logic is the problem. Effective learning comes through activities. If the same logic is built using puzzle games, the student is both at the delivering and receiving end making learning much more effective. Efficient understanding leads to efficient learning but effective learning comes via activities involving games. That knowledge lasts longer and gives us a better understanding of things. The puzzle game 'Treasure Hunt' applies greedy technique and helps students understand logic of the "Greedy" algorithm. The second puzzle game developed called 'Toads and Frogs' applies brute force technique and helps students understand logic of "Brute Force" algorithm. The puzzles in this application are taken from Anany Levitin's book "Algorithmic Puzzles". So it will not only be a source of entertainment and fun but it will also help the students learn the algorithmic techniques in a simpler way.

Acknowledgments

We would like to thank almighty Allah who enabled us pursue our ideas. Then we would like to thank our supervisor Mrs. Saima Jawad for believing in us, being a constant helping hand whenever we came across new challenges, motivating us whenever needed and most of all being a constant source of guidance and appreciation.
We would like to pay our gratitude to Bahria University Islamabad for allowing us to proceed with the system and putting our plans into action.

ASAD BASHIR
AYESHA ASGHAR
Islamabad, Pakistan

September 2015

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Game-based learning (GBL) begun in the early 1980s and has steadily grown and evolved[1] . Learning comes as a mandatory part of progress in a game. As we go further in the game out mind is enjoying the pleasure of struggling with the new game and coming to understand it. This holds true for both entertaining and serious mind boggling games.

1.1 Project Background

GBL is a broad terminology which refers to the use of games to uplift learning and teaching. GBL has invariably progressed since its origin in the early 1980s. Students can learn multiple techniques through GBL which can improve their ability of problem solving and critical thinking and can make their brain function in the best possible way. Creativity is another gift of GBL. Games are evolving in ways that improve skills of children rather than just amusing them. If the power of well developed games be rightly utilized to hit learning targets, it will result in efficient learners who will utilize their knowledge from the game in specified areas and practice problem solving techniques that they learned. The project is basically a GBL environment for logic building. Most of the students learn the concepts very easily but learning logic is the problem. Effective learning comes through activities. If the same logic is built using puzzle games, the student is both at the delivering and receiving end and the learning is effective. Within the learning environment of the game there is a lot of work being done in order to reach the goal, making choices and facing the consequences of the choice in an artificial environment. Through experimentation, logic is inculcated in the players mind. These games ignites the though process needed for logic building for programming problems. This knowledge can easily be ported to real life programming problems from the stimulated environment. Efficient understanding leads to efficient learning but effective learning comes via activities involving games.

That knowledge lasts longer and gives us a better understanding of things. The puzzles are taken from Anany Levitins book Algorithmic Puzzles. Solving algorithmic riddles is the most profitable and definitely the most pleasant approach to create and reinforce one's algorithmic thinking skills. So it will not only be a source of entertainment and fun but it will also help the students to learn the algorithmic techniques in a simpler way. Learning doesn't mean mere memorization. It implies obtaining the abilities and points of view expected to react suitably under pressure in all kinds of circumstances. Over the previous decade, the quantities of individuals who play electronic games have increased dramatically. Gaming itself has thrived as an industry. For the most part individuals play games to kill time and for amusement, yet imagine a scenario in which games were utilized to educate the users about various ideas. Student's motivation determines what they do to learn. The digital generation that makes up a substantial part of today's workforce is unaffected by conventional, lecture and tutorial based online learning approaches. Then again, they are exceptionally comfortable with videogames and amusement based learning. As indicated by specialists, learners have a tendency to learn faster on basis of feedback from the game, for example, scores and assessments.

1.2 Objectives

Most games demand the students to think fast and use their mind actively. This methodology helps them develop their own logic to scenarios and strengthen their ability to think of new ideas and techniques by themselves.

Students gather knowledge, analyze and combine it with what they already know and apply on the scenario they are placed in. This learning methodology is more interactive. Student centered learning is applicable by many strategies, few of which are:

- Activity-based learning,
- Problem-based learning,
- Case-based learning,
- Model-based learning, and
- Game-based learning.[2]

The idea of interactive, highly engaging training and learning is very old. The gap between old fashioned, passive learning methods and multimedia, user controlled learning methodologies continues to increase. GBL tools bridge that gap and results in more productive and engaged students and workers. These students and workers embrace learning rather than viewing it as a disruptive burden. To develop command in logic building, students must acquire component skills and practice integrating them. Logic

building is a process that happens slow and in bite sized chunks, each learner working at an alternating pace. There are training programs to help students learn to follow this process, but mostly the training is conducted on group basis. This means that students who learn slow will suffer and those who learn fast will wait for them to catch up. The focus on these training is to necessarily be on learning facts and rules, with limited opportunities to apply them therefore there's a need to develop an application that takes all students at the same pace and provides a platform to apply those learning on. Factual knowledge doesn't lead to building logic, to build logic, even in a group basis; all students can play games side by side and acquire logic in the meantime.

1.3 Problem Description

In this project we will mainly focus on GBL. It intends to build logic of students on algorithmic thinking. The project however will not cover almost entirely all puzzles needed for efficient logic building. The puzzles are categorized in three levels. Games will be designed in software 'Unity 3D'. A character will be designed in iClone 6 which helps move around in the environment. With the help of arrow keys, the character will be directed to the desired level of difficulty. The puzzle game will start once the character collides with the building. The score update will be displayed during the game and the aggregate score towards the end.

Iterative model of development will be used throughout the duration of project. Games will be designed in an incremental manner with each game being ready to play after completion. Once a game is ready, it will be tested for bugs and errors shortly after which it will be handed over to the user for reviews. Updates will be made keeping in consideration the user's view point. Lastly all the games will be put together in an environment in the final step. Software Development Life-cycle used for this project will be the spiral model. As the project consists of series of individual games that will be integrated into a single environment, prototyping is seen as best fit for this purpose.

1.4 Project Scope

The project will have puzzle games:

- With increasing level of difficulty.
- The player will be given privileges to pause and resume the game as per convenience.
- Hints will be given to keep the user from losing interest.
- Puzzles that allow multiplayer will be out of the scope of this project.

- The total score of the user will be displayed towards the end of the game
- The game will be designed for window based systems.

Note: The project however will not cover almost entirely all puzzles needed for efficient logic building. The puzzles are categorized in different level of difficulty.

Chapter 2

Literature Review

2.1

E-gaming has caught an eye of a lot of researchers over the past decade as they believe games have an immense effect on the users and therefore if games are developed for educational purposes, users would learn concepts more effectively. Game based learning is the field related to the study of such sorts[3]. Solving algorithmic puzzles is the most productive and definitely most enjoyable way to develop and strengthen one's algorithmic thinking skills[4] .
GBL encourages students playing computer games, which is fairly radical in numerous learning situations for something besides entertainment. There is apparently a distinction between what students realize while playing games (e.g critical thinking, visual-spatial thinking, coordinated effort, asset administration), and what teachers are working to achieve. The former provides a good source of learning with little effort and high degree of understanding.
To date, many games have been made on GBL, only a few of which will be discussed and analyzed critically in correspondence with the application that is intended to be made.

Code Combat targets a younger audience therefore the focus mainly stays on gamification with fantasy cartoon graphics. It is a source of amusement too as well as learning. The player sees a screen split in two halves, the left half has the game environment and the right half has the code editor. The game environment shows an avatar controlled via input commands in the text editor such as avatar.moveUp(), avatar.moveDown, avatar.attack() etc. Incorrect commands with logical programming errors will cause the avatar to run against the wall and hit dead ends, eventually causing it to die.
The avatar has certain tasks assigned for each level. The tasks include collecting gems, defeating evil powers or merely to reach the exit passing all the hurdles. As the level proceeds, the player is required to use advanced programming concepts like loops and

classes. Shuffling between levels, the player can also make use of the points from the previous level to get better armor, weapon or programming commands that help solve greater programming task in higher levels.

The learning curve of Code Combat is best suited to entertain students with no programming experience. It starts with less or no code acquiring tasks and gradually with the progress in levels. The curve grows steeper as programming tasks are included in the game environment. The levels itself become difficult as newer challenges are introduced in game environments such as surprise attacks, dead ends, fire traps, enemies with advanced weapons and much more. The levels have beautiful and colorful graphics and the complex riddles in the later stages meet the demands of gamers and coders alike.

Critically analyzing the game, Code Combat is a pleasant experience. It attracts younger ones to play and elder ones to help younger ones play. The best parts are the code comments which have hidden hints and instructions for student's assistance. The game itself is easy to understand without much help. The colorful graphics works well amongst younger students and gamification makes it famous amongst the elder ones. Students have a good time pondering where to invest their hard earned diamonds and what weapon to choose best in limited resources. However, the instructions and hints in the form of code comments were in no other language except English, making it hard for students with weak English skills. Few of Code Combat commands are for use only in the game. Those commands don't hold true in real programming problems. Instead of the structural command, the right Python command should've been used making the commands work in both gaming and real world.

Often children sneak past classes, reading material and homework assignments, without learning a lesson amongst the most essential math lessons of all which is 'Math is Fun'. "Math Doodle" application presents four Challenges, Sums Stacker, Connect Sums, Unknown Square and Splat GoRound. Every Task gives a lot of numerical practice, inside a recreational math setting. The numerical riddles permit clients to play, explore, and explore different ideas regarding mathematical calculations, while creating and reinforcing their technique and problem solving skills. Math Doodle challenges are intended to take into account numerous solutions. There can be more than one correct answer.

Math Doodle presents math ideas in an overwhelming way. The main screen is a perfect case of the unique approach this iPad application takes in demonstrating kids that math is enjoyable. A large number of the doodles on this screen are intuitive. One doodle that got my attention is of experts and individuals who do math; not only mad scientists, but wildlife biologists, game designers and nuclear engineers. Kids today still ask why they have to take in this stuff, since they are NEVER going to utilize the learning of math. Everybody thinks of that.

- Challenge number 1, Sums Stacker; is a decent test for beginners. The client drags the items from stack to stack until the aggregate of the articles in every stack equals the objective total appeared at the base of the stack. To expand further; values are divided into three stacks. Every stack is arbitrarily given a sun which it needs to reach. Move values with the help of your finger from stack to stack until the sum of every stack meets the objective aggregates shown at the base of the screen. The qualities can be represented in 22 distinctive ways that shift in their level of reflection: preschoolers may begin with the fingers, dice or curls. Grown ups have much more customizable options.

- Challenge number 2, Connect Sums; has the student touch the objects displayed to equal the target sum. During the Hard mode, the student must be able to connect the objects. In an array of 4x4 student tries to clear series of random values, this is very helpful in developing their number sense. The values that are selected are cleared if their aggregate equals the target sum.

- Challenge number 3, Unknown Square; requires the student to determine the missing value of the letters in the grid. In an array of 3x3 students begin to explore algebraic thinking by developing strategies to solve for unknowns in the array. Aggregate of all values is displayed in straight lines, row or columns.

- Challenge number 4, Splat GoRound; has the student choosing the correct hour, minutes, angle or fraction to move the fly swatter hand on the clock in order to splat the fly. This helps student understand the concepts of rotations.

Challenge Modes are for choosing how long the student wants to play the Challenge. Solve Mode requires the student to solve five sets in the least amount of moves. In the Infinity Mode the student plays until they choose to stop. This is the only mode currently available for Splat GoRound. Race Mode is for those that want to race against the timer; multiple options are available.
Math Doodles is super easy for all students to navigate. It is not one of those apps that you find yourself clicking through menu after menu to choose the options for play. In Math Doodles simply choose the challenge then choose the objects to represent values, the challenge mode and the difficulty all on the same screen. The manner in which positive feedback and acknowledgment of when the set has been completed correctly is offered will encourage the child to continue playing. The app is named Math Doodles for a reason. All of the artwork looks as though it has been taken from notebooks, napkins and all sorts of scrap paper. The doodles are extraordinary and add to grabbing student's attention. Due to the app not providing narration of the instructions some students may need assistance. Math Doodles is completely student-safe. It does not contain in-app purchases, ads, social media buttons or external links. Critically analyzing these games, the concept is so unique

and challenging. The hand-drawn graphics are so much fun. To increase the level of challenge and to make the game user friendly there are many customizable features. The games are cleverly designed. The lower level of games would entertain the kindergarteners. Some levels would be fun and amusing for high school children and adults. The best part about the game is customizable features. Different symbols and items can replace the numerals to make it more fun and catchy. These games can serve as an engaging way to help children catch up with students in advancing class.
Montessori Crossword is based on learning methodology for Montessori students. Montessori Crosswords helps students develop their reading, writing, and spelling skills.
Montessori Crosswords helps students learn and understand two fundamental concepts:

- Students understand that words are made up of sounds which are also referred to as phonemic awareness. For each word, students can touch the empty rectangle where a letter is supposed to be placed, and hear the sound the corresponding letter produces.
- Students memorize the sound associated with letters by providing a phonics-enabled alphabet where students can touch each letter and hear the associated sound. Montessori Crosswords allows you to select categories depending upon the sound of the word and the difficulty of the word:
- Level 1 displays a word comprising of 3 letters with no difficulty for beginning readers.
- Levels 2 and 3 offer more complex words that contains more complex sounds such as long vowels sounds. It also allows the student to make multi-word crosswords in numerous different combinations.
- Alternately, you can choose words that contain a specific sound.

To create an enjoyable experience and to motivate the student for better, interactive visual effects appear once the word is completed. The sound and animations are also well designed.
For parents trying to find an educational app to give their children a head start or teachers looking for a phonemic awareness/phonics activity for their students, this is the perfect app. The letter sounds are clear and the options available for voice are not irritating for children. The settings allow you to choose upper or lower case letters, abc order or qwerty order, and varied fonts. It follows the best teaching practices for helping young children learn the sounds and functions of the letters. The letter sounds are accurate and the animations are fun to play with.
Lightbot is a simple puzzle game designed for students to learn basic designing concepts. The initial set of basic levels in the game focus on basic programming commands. The

student learns the basic and the series of basic commands to guide the avatar in the environment. The student also learns a simple way of writing a short computer program.
In Lightbot, a robotic avatar is appears on a grid of square tiles. The goal to achieve in each level is to have the robot move in all directions and light up all the blue tiles in the grid. The Lightbot understands basic set of instructions to move with the help of arrow keys and a Lightbot icon is used to tell the Lightbot to light the tile its standing on. Lightbot is controlled via instructions also called a program. On press of the play button, the program will run and all instructions in the program will be executed. The Lightbot will perform all the tasks as instructed in the program. The Lightbot will be instructed by the program to go the start in case of a mistake. Then on change the program to give correct instructions for Lightbot to make no mistakes. All the instructions the Lightbot knows won't be known to you all at once. More instructions will be revealed as the game proceeds in levels.
The aim is to get the student to learn basic programming concepts. This can be some in four simple steps. The student thinks of instructions to move the avatar around the puzzle board. The student then writes them down and has the Lightbot run through them. The student now checks for all blue tiles, whether they're lit or not. Lastly a simple Debugging technique is applied if the results weren't as desired. Try not to be demoralized if your system is not right on your first attempt. Basically read each instruction you gave Lightbot closely and envision how he would and ought to act at every progression. This exceptionally difficult puzzler does a remarkable job with regards to introducing children to programming ideas but critically analyzing, it might mostly connect with children who might be attracted to programming already, since the bar or level of difficulty raises dramatically on consecutive levels. There are no user accounts, so there can't be multiple students playing at one time, however children can go back and begin once again for another player. The level of test inclines up rapidly, so younger children may appreciate easier games like My Robot Friend or Kodable Pro. This game targets students who know ground level programming. It's normal to get stuck for some time on a few levels but as a drawback no clues or pieces of information are given in comments or hints. Kids get as many lives as need be, however, and they'll gain from every failure. The game stops once the student is successful.
This chapter covers few games that are built to promote learning of Math, Letters/Alphabets and counting at a Montessori level. Nothing like logic building has been inculcated so far in the world of game based learning. With the advent of GBL, gaming environments have been extended to facilitate all age groups and all existing fields thereby taking graphics, ideas, concepts and color schemes to a whole new advanced level.
GBL environment for logic building draw us into practical virtual environments that students find relatable with real life problems. Students can relate the knowledge achieved with the real-life work really easily. Having a good game design and mature graphics intends to attract audience of a greater age group. Apps like Math Doodle and Montessori Crossword can get away with glitches of repetitive questions but it doesn't go unnoticed

in an Application designed for older audience. Such malfunction draws the interest of audience away and the purpose of learning is also sidelined. The room for mistakes is therefore expected to be minimum in the application. The learner applies prior knowledge and learns knowledge which is nothing like previous applications where inculcation of new knowledge and learning is the basic aim.

In contrast, previous Applications drill on certain narrow procedures, and then evaluate on the memory of bookish knowledge. Even when the child successfully retains the lesson's facts and procedures, the logic remains untested. In addition, even the most comprehensive of these games will not cover procedures for every complex or tricky question the child may encounter. In game based environment for logic building, the most important underlying hows and whys are taught to the child. This understanding of deeper and more abstract situations prepares the student to perform consistently and effectively, even in new and unexpected situations. One of the most well-known web game site that helps players in programming is

[5] GBL is designed to achieve defined learning outcomes. The outcome is expected to target audience of defined age groups and mind sets. What attracts one may or may not attract others. So in order to cater interest of a larger audience, a generic game is developed. GBL environment for logic building is targeting beginner level programming students. Hence the outlook, colors and graphics used are in consideration with the mindset that the student is expected to have.

When education or training feels dull, we are not being engaged and motivated. The learning process slows down or stops. "Learning" doesn't mean memorizing things unthinkingly. It means learning the art of thinking in the right mind and acquiring a thought process needed to respond appropriately under pressure, in a variety of situations. Logic building is an even greater accomplishment that follows learning.

Chapter 3

Requirement Specifications

3.1 Existing System

This project is intended to make games on GBL for logic building. Logic cannot be taught, it can only be built. . Mostly people play games to pass their time and for entertainment, but what if this platform was used to teach the users about different concepts [6]. The existing systems that are intending to build logic are doing so in a completely different manner. Teaching logic building techniques and methodologies are confined to classrooms and books only. The knowledge gained off lecture slides and tuition tutorials is not long lasting and was mostly teacher centered learning. The teacher is likely to end up with a better understanding of the subject than the student. In classrooms not all attention is given to lectures, not every word is carefully heard and absorbed. Students lose track of snippets of lecture and find hourly lectures boring. Due to conventional ways of teaching logic, student looses interest and is absent minded during lectures. The knowledge taught is not always made complete use of in real world scenarios as student finds it difficult to relate and apply bookish knowledge with real time situations. Logic takes time to develop and logic building is a time consuming thought process. The brain functions best in practical scenarios where it is being made use of the most.

3.2 Proposed System

GBL utilizes challenging activities, either setting the students against each other or inspiring them to test themselves so as to inspire them to learn better. Games frequently have a fantasy component that draws in players in a learning movement through a storyline. Over the time convenient ways of learning are being replaced by newer and efficient ways. GBL is a technique which has been worked on over the past few years and has phenomenal feedback. Students find it interesting and spend hours playing games which is an indirect way of their learning. They do not realize they are building logic on the backend which

can be easily applied when exposed to real world scenarios. The slow student does not lag behind as it can be in lecture theaters, everyone learns and adapts to the game in more or less at a constant pace. The system that is intended to be made is supposed to be installed in laboratory systems as part of student's lab session activity. Later on the game may also be ported into their mobile phones as home exercise.

3.3 Requirement Specification

Learning was never so much fun. Hints are provided as comments to keep student from losing interest. Both puzzle games are classical examples of GBL hence learning is a mandatory part. Logic is being built as each puzzle targets a specific knowledge area and the game is built accordingly. The background sound will be an added feature. The games are divided into three distinctive levels which make the student to aim for higher level once done with the lower one. To create a truly educational game based on student's interest, a motivational dialogue appears in case of win. The students are now motivated to learn outside classroom on their own as game play is fun and knowledge is easy to learn. The lab sessions will now liven up and will be a whole new fun experience. The students immerse themselves into the learning experience and contribute one hundred percent of their bit as instructors contribute theirs. The students learn from their mistakes and gradually do much better at home after practicing each level of the game numerous times.

3.4 Use Cases

UC1: Login

Description: User chooses to sign up or sign in. **Purpose:** To get users choice. **Pre-Condition:** Application must be opened. **Primary Actors:** The user of the Application.

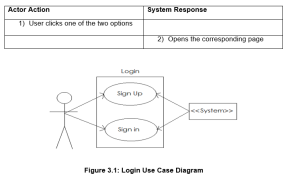

Figure 3.1: Login Usecase Diagram

UC2: CreateNewAccount Description: User creates a new account. **Purpose:** To get user registered. **Pre-Condition:** User must choose sign up option. **Primary Actors:** The user of the Application

UC3: EnterUsernameAndPassword

Actor Action	System Response
1) 1) The user enter username	
2) The user enter password	
3) Retype Password	
4) The user selects to signup	
	5) The system will save the username
	6) The system will save the password
	7) Application will open login page

Figure 3.2: Signup Usecase Diagram

Description: User Signs in. **Purpose:** To get user started. **Pre-Condition:** User must choose sign in option. **Primary Actors:** The user of the Application

Actor Action	System Response
1) The user enter username	
2) The user enter password	
3) The user selects to login	
	4) The system will verify the username
	5) The system will verify the password
	6) Application will display all the games

Figure 3.3: Signin Usecase Diagram

UC4: SelectGame Description: Select Game Use case is used to select game. **Purpose:** To select any game. **Pre-Condition:**

- The user must have an account.
- The user must be logged in.

Primary Actors: The user of the Application.

Actor Action	System Response
1) The user selects any game from the interface.	
	2) The system will verify the game
	3) The system will open the categories of that particular game.

Figure 3.4: Select game Usecase Diagram

UC5: SelectCategory Description: Select Category Use case is used to select category from the interface. **Purpose:** To select category for the selected game. **Pre-Condition:** The game must be selected. **Primary Actors:** The user of the Application

UC6: GamePlay Description: Interactive game environment Use case is used to show human computer interaction during the game. **Purpose:** To show interactions during the game. **Pre-Condition:** The game category must be selected from the interface. **Primary Actors:** The user of the Application

Actor Action	System Response
1) The user selects any category (easy, medium, hard) from interface.	
	2) The system will verify the category
	3) The system will load the game of that category

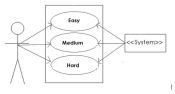

Figure 3.5: Select Category Use Case Diagram

Figure 3.5: Select Category Usecase Diagram

Actor Action	System Response
	1) Start up
	2) Set up game
3) Play	
	4) Result displayed
	5) The system will compare and display high score in a pop-up window

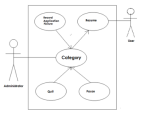

Figure 3.6: Game Play Usecase Diagram

UC7: ExitOrPlay Description: Quit or play Use case is used to select between exit or play further. **Purpose:** To show choices for the user. **Pre-Condition:** The game level must be completed. **Primary Actors:** The user of the Application

Actor Action	System Response
	1) Shows user two discrete choices of Exit or play.
2) User selects a choice.	
	3) If user selects Exit, Shut down Else returns to homepage.

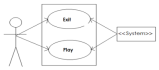

Figure 3.7: Exit or play Use Case Diagram

Figure 3.7: Exit or Play Usecase Diagram

Chapter 4

Design

4.1 System Architecture

The system being developed will comprise of an interactive Graphical User Interface that will be displayed as the start up scene. The environment will give user an option to select a game of his choice as shown in figure 4.1. The character moves in the environment with the help of arrow keys and goes through all options of games and their levels which will be graphically displayed in the form of buildings. Each building will be representing a distinctive game. Once the user selects the game and the character collides in the desired building, the game is built and loaded. Upon the completion of the game, the score is compiled; the aggregate score is displayed alongside "Game Over" animation. The user is then asked to select an option between quit or replay. The control is transferred to main GUI in case of quit. Relay redirects the game to the startup scene of the same game.

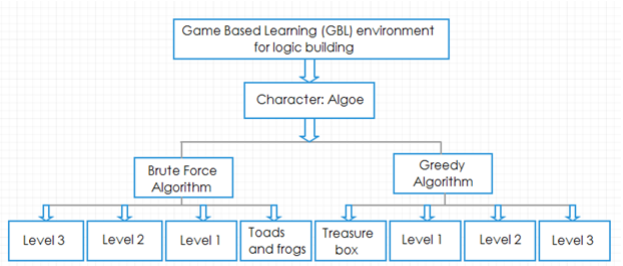

Figure 4.1: System Architecture

4.2 Design Constraints

The application does not have multiplayer feature and therefore players cannot compete against each other. The game must be installed in a system for the user to play, it is not available online. The game does not entirely cover all algorithmic techniques for logic

building and hence is a future enhancement. Customizable features are an optional scope. A future enhancement of the game would be adding more customizable features to it hence making it user friendly.

4.3 Design Methodology

Iterative model of development will be used throughout the duration of project. Games will be designed in an incremental manner with each game being ready to play after completion. Once a game is ready, it will be tested for bugs and errors shortly after which it will be handed over to the user for reviews. Updates will be made keeping in consideration the user's view point. Lastly all the games will be put together in an environment in the final step. Software Development Life-cycle used for this project will be the spiral model and is shown in figure 4.2. As the project consists of series of individual games that will be integrated into a single environment, prototyping is seen as best fit for this purpose.

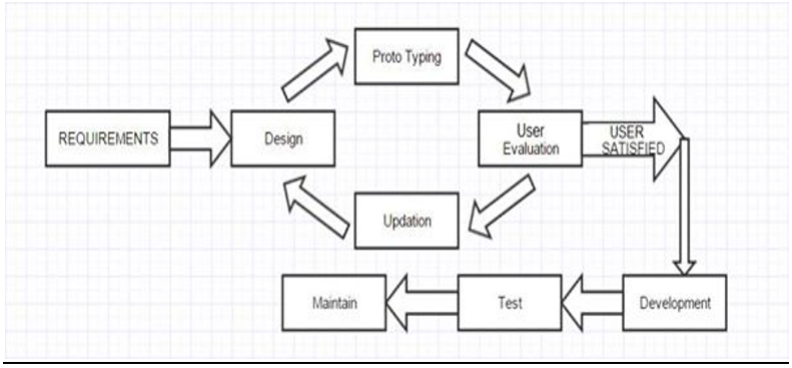

Figure 4.2: Design Methodology

4.4 High Level Design

This section provides the abstract view of system and its outlook. High level design contains the view of the application to the user and not the internal details.

4.4.1 Conceptual Design

Figure 4.3 shows the conceptual and logical view.

4.4.2 Process Flow

Figure 4.4 describes the sequence of actions that take place during the course of the application in other words it explains the working of application and the interaction of processes at runtime.

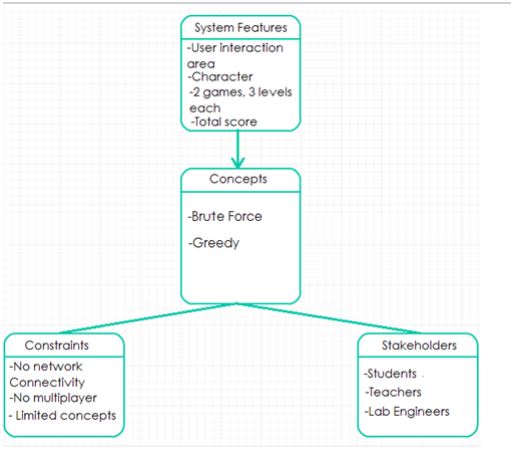

Figure 4.3: High Level Design

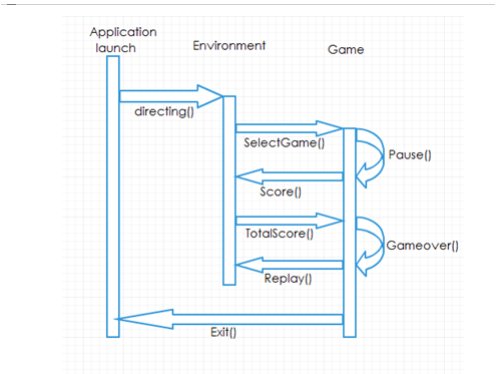

Figure 4.4: Process Flow

4.4.3 Modular Diagram

This section shows the modules of the system and their interaction with each other.

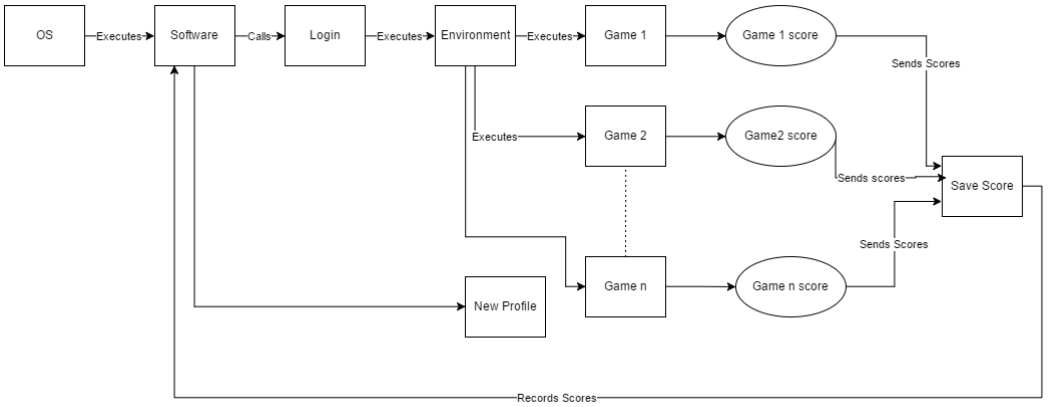

Figure 4.5: Modular Diagram

4.5 Low Level Design

This section explains the detail working of the component of the application in such a way that the diagram is self explanatory.

4.5.1 Treasure hunt

Figure 4.5 shows working of Game 1, level 1, level 2 and level 3.

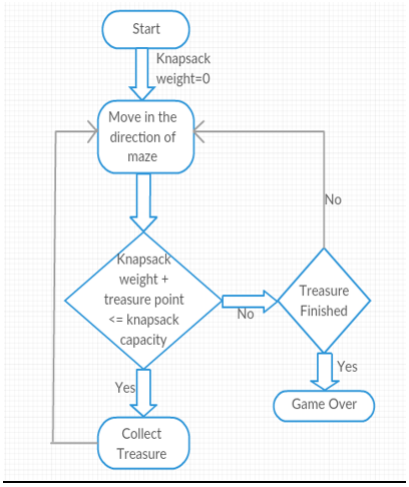

Figure 4.6: Game 1 Flowchart

4.5.2 Toads and frogs

Figure 4.6 shows working of Game 2, level 1, level 2 and level 3.

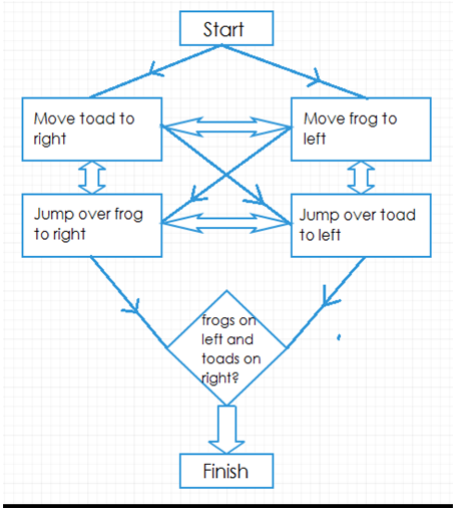

Figure 4.7: Game 2 Flowchart

4.6 Graphical User Interface Design

Figure 4.7 shows the rough outlook of the homepage. This is the interactive interface that the user will come across at star of the game.

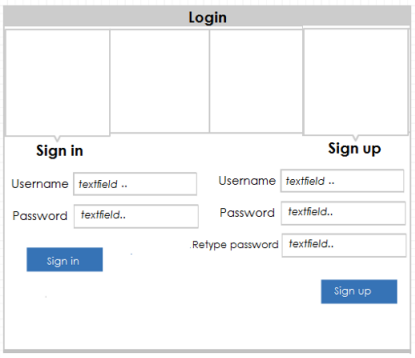

Figure 4.8: GUI Design

Chapter 5

System Implementation

Implementation is putting a plan into effect hence executing the program. It is the practical demonstration of a mental model or execution of a set of ideas. In computer science, an implementation is a realization of a technical specification or algorithm as a program, software component, or other computer system through computer programming and deployment.

5.1 System Architecture

The project comprised of a total of six iterations. Each iteration resulted in a deliverable i.e. prototype. By the end of every iteration, the product was submitted for user evaluation and possible updates were made accordingly.

5.1.1 First Iteration

The first iteration resulted in a prototype of level 1 of the first game that catered the logic of "Greedy" algorithm. It is a demonstration of the classical "Knapsack Problem". The students should be able to apply logic of greedy algorithm in practical life scenarios once they have played the game. The iteration comprises of:

- A simple maze object with gem objects of different values and weight.
- A treasure box which is the player object having a weight limit.
- A camera controller script that focuses on movement of the player. A specific distance of the object is set from the camera. On movement of the player object, the camera moves alongside, maintaining the set distance.
- A player controller script that triggers the animation of the player. It controls its movement though the code and updates its position on the frame. It allows movement

control via keyboard. The script contains a start function that gets position of previous frames and initializes all the variables.

- A weight collection script that deals with colliding objects. Collision can either be against walls of the maze or with the gems. Collision through walls is detected with the help of mesh collider. When the player object collides with another object carrying the tag ‘gem’, the gem object is destroyed and its weight is subtracted from the weight limit of treasure box. On the next update the weight bar of the treasure box is decreased according to the size of the gem. Different animations are executed according to the trigger called. Trigger ‘item collected’ executes the animation of the box. Trigger ‘container full’ executes game over animation when called.
- A score manager script that keeps track of the total profit. Canvas has a UI text that is updated on every addition in the treasure box.
- A game over manager script that keeps a check on total weight of treasure box and upon reaching the weight limit, a game over animation is displayed. A timer of five seconds on the animation is set for it to display for five seconds only before restarting the game.Figure 5.1 is a design model of game 1, level 1.

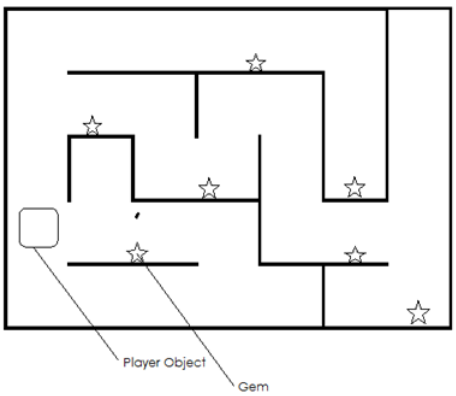

Figure 5.1: G1 L1 Design

5.1.2 Second Iteration

The second iteration resulted in a prototype of the level 2 of first game catering the same logic of “Greedy” algorithm as level 1. The iteration made significant changes, increasing the difficulty level. In addition with complex maze structure, the weight limit of knapsack was reduced. More choices of gem were introduced making it difficult to make a better weight against value choice. The choice of gems is thereby more thought consuming. The scripts were similar to those in level 1. The iteration comprises of:

- A maze object with gem objects of different values and weight.
- A treasure box which is the player object having a weight limit.
- A change in weight collection script was made to decrease the weight limit of treasure box.
- Figure 5.2 is a design model of game 1, level 2.

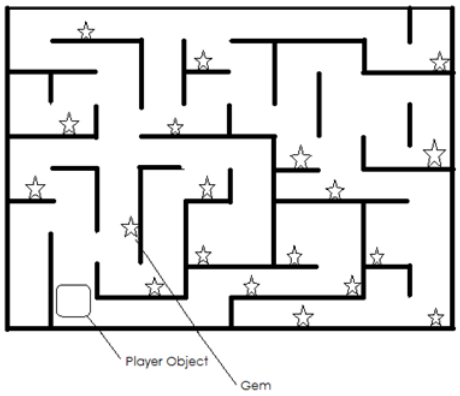

Figure 5.2: G1 L2 Design

5.1.3 Third Iteration

The third iteration resulted in a prototype of the level 3 of first game catering the same logic of "Greedy" algorithm as prior levels. The iteration made significant changes, increasing the difficulty level. In addition with complex maze structure, the weight limit of knapsack was reduced. More choices of gem were introduced making it difficult to make a better weight against value choice. The choice of gems is thereby more thought consuming. The scripts were similar to those in prior levels. The iteration comprises of:

- A complicated maze object with gem objects of different values and weight.
- A treasure box which is the player object having a weight limit.
- A change in weight collection script was made to decrease the weight limit of treasure box.
- Figure 5.3 is a design model of game 1, level 3.

5.1.4 Fourth Iteration

The fourth iteration resulted in a prototype of level 1 of the second game that catered the logic of "Brute Force" algorithm. It is a demonstration of the classical "Toads and frogs"

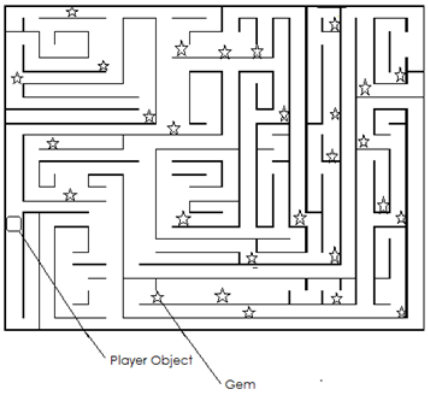

Figure 5.3: G1 L3 Design

problem. The students should be able to apply logic of Brute Force algorithm in practical life scenarios once they have played the game. The iteration comprises of:

- Equal number of toad and frog objects.
- • A script called “toads and frogs array”. It contains an array of all elements. The elements are attached to it via inspector. Three frog objects with tag ‘Frog’, three toads with tag ‘toad’ and one empty object with tag ‘empty’. Raycast physics and mouse click detection is used to check the index of the element in the GameObject array which is clicked. The frog only moves right and the toads only move left to interchange position. The game objects in the array are swapped after some checks.
- Figure 5.4 is a design model of game 2, level 1.

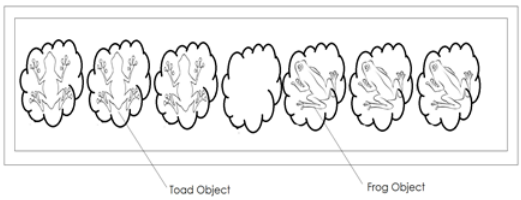

Figure 5.4: G1 L4 Design

5.1.5 Fifth Iteration

The fifth iteration resulted in a prototype of the level 2 of the second game catering the same logic of “Brute Force” algorithm as level 1. The iteration made significant changes, increasing the difficulty level. The number of toads and frogs were increased resulting in greater complexity. The script was similar to that in level1. The iteration comprises of:

- Equal number of toad and frog objects.

- A change in “toads and frogs array” script. The number of toad objects and frog objects was 5.
- Figure 5.2 is a design model of game 2, level 2.

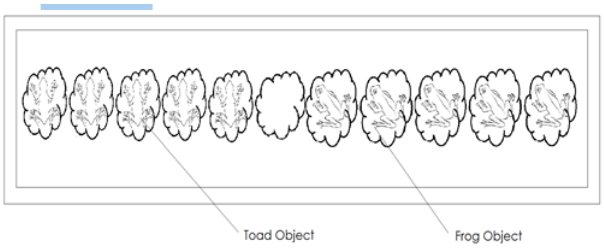

Figure 5.5: G1 L5 Design

5.1.6 Sixth Iteration

The sixth iteration resulted in a prototype of the level 3 of the second game catering the same logic of “Brute Force” algorithm as prior levels. The iteration made significant changes, increasing the difficulty level. The number of toads and frogs were increased resulting in greater complexity. The script was similar to that in level 1 and level2. The iteration comprises of:

- Equal number of toad and frog objects.
- A change in “toads and frogs array” script. The number of toad objects and frog objects was 7.
- Figure 5.6 is a design model of game 2, level 3.

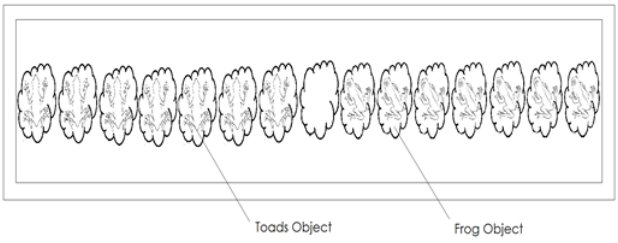

Figure 5.6: G1 L6 Design

5.2 Tools and Technology Used

Tools used in the development of the system are as follows:

- Unity 3D
- Mono-Develop
- Dot-NET Framework
- iClone 6

5.2.1 Unity 3D

Unity3D is an ultimate platform for game development. It creates both 2D and 3D games which can be ported to mobile phones, desktops and consoles. The environment was created making excellent use of Unity3D reusable models. It provided a visually stunning user experience. It's a simple tool to use and develop games.

5.2.2 Mono-Develop

Mono-develop is the scripting tool used to write and edit scripts in Unity 3D. The scripts were then attached to objects in the game environment for them to be interactive and usable. Mono-develop provide liberty for the user to choose between three programming languages (JavaScript, C Sharp and Boo).

5.2.3 Dot-NET Framework

The .NET Framework is Microsoft's complete and stable programming model for building applications that have visually stunning user experiences. It helps create desktop, mobile and web applications. This project is a desktop application using Dot-NET Framework platform

5.2.4 iClone 6

iClone is an easy to use real time 3D animation software. Textures created and significant changes made in existing textures in this project was made possible by iClone 6. It was used often as 3D animation games require different textures and graphical content.

5.3 Development Environment/Languages Used

Development Environment: (Unity 3D) environment. Programming language: C-Sharp.

5.4 Processing Logic/Algorithms

The algorithms used in the application are listed in this section.

5.4.1 Game 1 Pseudo-Code

START
WHILE TreasureboxWeightLimit > 0
DO
Move treasurebox along the maze
IF Collision with gem THEN
TreasureboxWeightLimit – GemWeight
ELSE
keep moving
ENDIF
ENDWHILE
END

5.4.2 Game 2 Pseudo-Code

START
WHILE First three position gameobject.tag=fog AND last three positions gameobject.tag=toad
DO
IF Click Mouse on game object THEN
IF gameobject.tag=frog AND array[i+1] = THEN
Swap
ELSE IF
IF gameobject.tag=toad AND array[i-1] = THEN
Swap
ELSEIF clicked gameobject.tag=frog AND array[i+1].gameobject.tag=toad AND array[i+2].gameobject.tag=
THEN
SWAP array[i] with array[i+2]
ELSEIF clicked gameobject.tag=toad AND array[i-1].gameobject.tag=frog AND array[i-2].gameobject.tag=
THEN
SWAP array[i] with array[i-2]
ENDIF
END

Chapter 6

System Testing and Evaluation

6.1 Graphical User Interface

Test Case ID	**TC1 Graphical User Interface Testing**
Description	To ensure graphical user interface is correctly functional
Applicable for	Developed Application.
Preconditions	Application must be installed.
Initial Condition	Application must run without corrupting.
Test No.	**Test and expected result**
1	Launch application.
2	Verify the startup scene displays.
3	Verify the graphics are displayed.
4	Verify the color scheme is correctly displayed.
5	Verify there are no faulty and incompatible graphics.
6	Verify the user is able to move the character using arrow keys.
7	Verify the game startup scene is displayed upon selection of game.

Table 6.1: Graphical User Interface

6.2 Usability Testing

Test Case ID	**TC2_Application_Startup**
Description	To ensure startup scene is executed when the application is launched.
Applicable for	Developed Application.
Preconditions	Application must be installed.
Initial Condition	Application is set up to be tested.
Test No.	**Test and expected result**
1	Switch to directory where setup for developed application is to be found.
2	Run “GBLforLogic.exe” from directory.
3	Run “GBLforLogic.exe” from directory.
4	Verify the user is able to move the character using arrow keys.
5	Verify the controls are functioning.
6	Verify the selection of game redirects the user to game startup scene.

Table 6.2: TC2_Application_Startup

Test Case ID	**TC2.1_Game1**
Description	Tests that the user is able to interact with the treasure box object with arrow keys and the game finishes when treasure box weight limit is reached.
Applicable for	Developed Application.
Preconditions	Application must be installed.
Initial Condition	The user makes the character collide with the Game1 building.
Test No.	**Test and expected result**
1	Verify the gems, treasure box and the maze is displayed.
2	Verify the treasure box moves in all four directions with the help of arrow keys.
3	Verify the weight limit bar of the treasure box regressing as gems are collided into.
4	Verify the treasure box does not go through walls.
5	Verify the “score” is updated in the score label as gems are collected.
6	Verify the game over animation is displayed alongside total score as the weight limit bar reaches empty.

Table 6.3: TC2.1_Game1

Test Case ID	**TC2.2_Game2**
Description	Tests that the user is able to interact with the toad and frog object with the help of mouse click and the game finishes when positioning of toads and frogs is interchanged.
Applicable for	Developed Application.
Preconditions	Application must be installed.
Initial Condition	The user makes the character collide with the Game2 building.
Test No.	**Test and expected result**
1	Verify the toads and frog objects are displayed.
2	Verify the mouse click on either of two makes them move.
3	Verify equal number of toads and frogs.
4	Verify Game over animation is displayed on reaching the solution.

Table 6.4: TC2.2_Game2

6.3 Software performance testing

Test Case ID	**TC3_Software_Performance_Testing**
Description	To determine speed and effectiveness of the application.
Applicable for	Developed Application.
Preconditions	Application must be installed.
Initial Condition	Application must be installed.
Test No.	**Test and expected result**
1	Application must be installed.
2	Start a stopwatch sensitive to micro seconds at launch.
3	Record the time duration from user interface to display of the game scene.
4	Repeat test 1-3 several times.
5	Record minimum and maximum readings on stopwatch.
6	Measure average and calculate speed.
7	Verify speed is up to standard.
8	Verify efficiency by recording number of application failures.
9	Plot results.

Table 6.5: TC3_Software_Performance_Testing

6.4 Compatibility testing

Test Case ID	**TC4_Compatibility_Testing**
Description	To determine compatibility of application with Windows.
Applicable for	Developed Application.
Preconditions	Setup for developed application and windows is available.
Initial Condition	A work station is available
Test No.	**Test and expected result**
1	Install Windows 7, Windows 8, or Windows NT 3.51 or later.
2	Install the application.
3	Verify the application is functional.
4	Upgrade the computer to Windows 10 or a more recent version of Windows.
5	Install the application.
6	Verify the application is functional.

Table 6.6: TC4_Compatibility_Testing

6.5 Exception Handling

Test Case ID	**TC5_Testing_Exception_Handling**
Description	To test how the application responds to occurrences during computation of exceptions.
Applicable for	Developed Application.
Preconditions	Application must be installed.
Initial Condition	Application must run without corrupting
Test No.	**Test and expected result**
1	Verify a complete error message is displayed in case of exception.
2	Verify the application is rightly recovered.
3	Verify execution status is updated.
4	Verify the error is logged by the application for future references.

Table 6.7: TC5_Testing_Exception_Handling

6.6 Load testing

Test Case ID	**TC6_Load_Testing**
Description	To test how the application responds in both normal and anticipated peak load conditions.
Applicable for	Developed Application.
Preconditions	Application must be installed.
Initial Condition	Application must run without corrupting.
Test No.	**Test and expected result**
1	Verify the application runs efficiently while the system downloads large series of files from the internet.
2	Verify the application runs efficiently while the system runs multiple applications simultaneously.
3	Verify the application runs efficiently while the system gives commands to a printer in a queue.
4	Verify the application runs efficiently while the system writes and reads data to and from the hard disk continuously.

Table 6.8: TC6_Load_Testing

6.7 Installation testing

Test Case ID	**TC7_Installation_Testing**
Description	To ensure application has been installed correctly.
Applicable for	Developed Application.
Preconditions	Setup for developed application is available.
Initial Condition	Equipment setup as per given instructions.
Test No.	**Test and expected result**
1	Switch to directory where setup for developed application is to be found.
2	Extract the setup file to the destination.
3	Launch setup.
4	Verify the startup scene of the application developed loads correctly.
5	Verify the user interface is displayed.
6	Verify the interactive controls work.
7	Verify the level selection buttons takes user to next scene upon selection.

Table 6.9: TC7_Installation_Testing

Chapter 7

Conclusions

Game Based Learning (GBL) environment for logic building has been a successful effort towards fulfilling its aims and objectives. The project's scope and usability proclaims its inarguable success. Keeping in view the literature and context, it can be deduced that such projects immensely benefit students and therefore this methodology of learning will vastly grow over the years to follow. From the use of this application it is concluded that it successfully accomplished its purpose and it is feasible to inculcate this methodology of learning for practical use in university environment.

7.1 Future Enhancements

- Port the games on to android and IOS platform for greater ease of use.
- Enhance the application to allow multiplayer option and for the users to compete against each other over the internet.
- Extend the project and cater more algorithms for logic building.

References

[1] Johnson. *The NMC Horizon Report: 2011 K-12 Edition. Austin, Texas: The New Media Consortium*, 2011. Cited on p. 1.

[2] Huba M. E. & Freed. *). Learner centered assessment on college campuses: Shifting the focus from teaching to learning. Community College Journal of Research and Practice, 24(9), 759-766. Date Accessed: 2nd February, 2015 at 1430hrs.*, 2000. Cited on p. 2.

[3] Eugenio J. Marchiori Angel del Blanco. *Introducing Educational Games in the Learning Process*, 2015. Cited on p. 5.

[4] Levitin and Anany. . *Algorithmic Puzzles. New York, Oxford University Press Inc. . Preface Page ix.*, 2011. Cited on p. 5.

[5] HadiPartovi. *www.code.org. Date Accessed*, 2015. Cited on p. 10.

[6] Marina Papastergiou. *Digital Game-Based Learning in high school Computer Science education*, 2016. Cited on p. 11.