**Appendix A**

**(Game Design Document)**

1. **GEEKS VS BUGS**
   1. **Game Name**

The name has originated from the word that is being used to describe eccentric or non-mainstreamed people. The term ‘Geeks’ has been constantly used to label people who are technically knowledgeable and is often seen to be operating with a computer.

Bugs, on the other hand is a jargon term used by software professionals to point an unnecessary issues or errors within a software. Bugs are parts of a software program that is not properly written according to the programmer’s purpose. But in this game, bugs are portrayed in a typical form—insect.

As ‘Geeks’ is a label used for IT literate people—mostly programmers, and ‘Bugs’ as their all-time antagonists, this game will revolve around the conflict between the two entities. It involves puzzle to locate bugs and remove it from the hardware devices within the computer. The player, as the geek, will be entitled to pick from two Nanobots—a robot with a size of an insect. The player will be able to control the Nanobot through encoding a single thread of instruction, which will be executed once.

1. **Game Overview**
   1. **Game Concept**

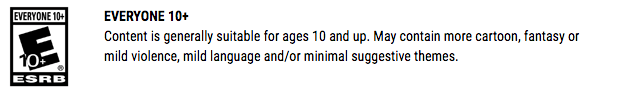
Geeks VS Bugs is an educational puzzle game that aims to capture the attention of its audience but with underlying purpose of introducing basic computer programming concepts to the players. The story of the game revolves around the story of a computer error and bug that will be posing threats in the future—year 2100AD.

* 1. **Genre**

Geeks VS Bugs is a multiple genre game. It is an educational video game that uses the method of puzzle to stimulate logic and challenge to the player.

* 1. **Target Audience**

The game is recommended and designed for the needs of ages 14–22 years old but holds a rating of “Everyone 10+”. This rating follows the standard of “Entertainment Software Rating Board” (ESRB).

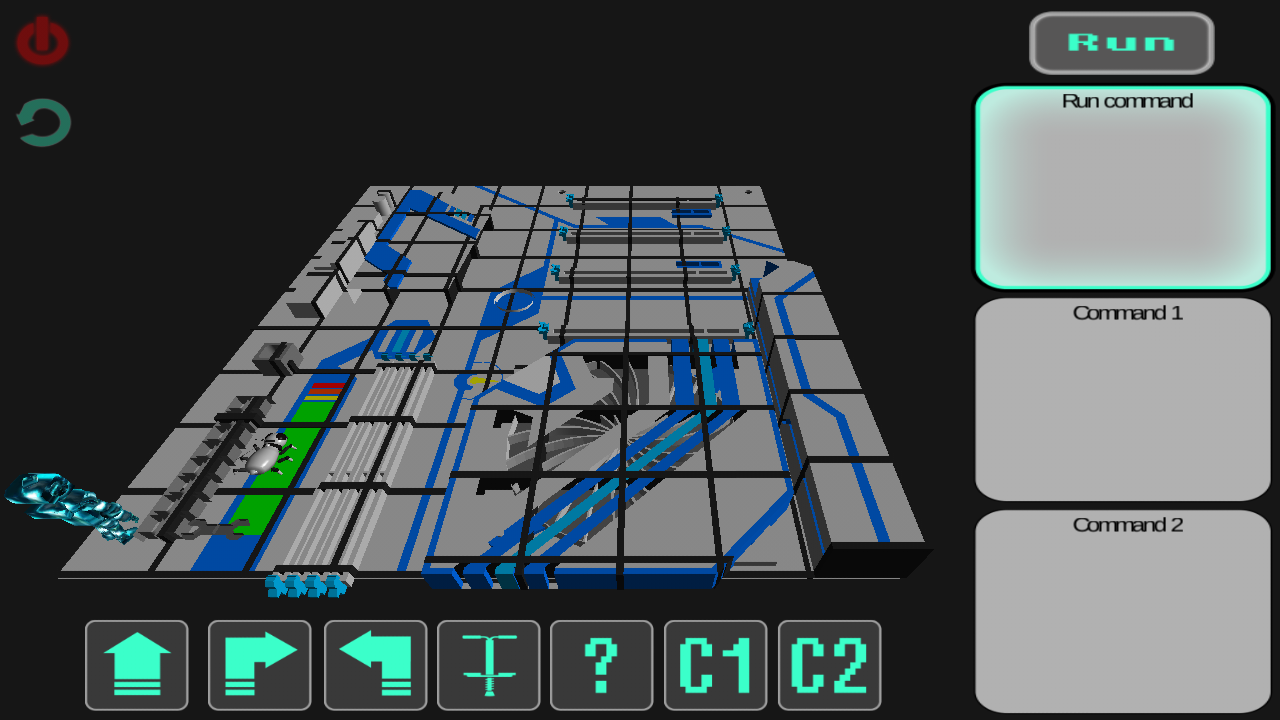


* 1. **Game Flow Summary**

To play the game, the player—as the geek who controls the robot—has to encode instructions to a thread. After compiling, the game will first check if all encoded values are logically possible. If, in any case, the game detected a logical error within the game the Nanobot will refuse to execute the given set of instructions and require the gamer to check and recode the thread of commands. The goal of the Nanobot is to locate and remove the bugs that are in the computer hardware. In most instances, the Nanobot will be required to take a necessary tool that in the map to accomplish it’s mission.

* 1. **Look and feel**

The game is design in a futuristic theme as for the story has a connection with the future. It applies 3D design for the characters and game world while 2D was used for buttons and other graphics.



1. **Gameplay and Mechanics**
   1. **Gameplay**
      1. **Game progression**

The game consists of 4 stages with 6 levels each. Each level requires the player to be able to accomplish tasks though providing instructions to the character—the Nanobot. After accomplishing the tasks, the player will be allowed to proceed to the next level. Each stage requires a minimum number of medals that are being awarded by each level depending on the player’s acquired score.

* + 1. **Mission/Challenge Structure**

The game has various missions and challenges.

In early stages and levels, the player’s mission is to program the Nanobot to reach certain spots from the map. As the player’s level raise, the mission changes to finding tools or locating and killing bugs. The missions seem simple but the game world’s maze and the method of programming the Nanobot makes complex and challenging.

* + 1. **Puzzle Structure**

There are two kinds of puzzle in the game. First is the maze of the game world, second is the combination of commands the player would use in order to accomplish its mission.

* + 1. **Objectives**

The main objective of the game is to remove the bug on the computer components by moving the character using the set of commands provided by the player.



* + 1. **Play flow**

The game flows with the succession of levels and stages. Each level may start with the explanation of new element for the player. Introduction of level’s mission will then be presented. After the mission has been laid out, the player will be allowed to input combinations of commands to program the Nanobot’s travel within the map’s maze. Then when the player is already satisfied with his provided instructions, they will compile the instructions. The game will then check for possible logical errors that the compiled commands may result to. If there are no errors with the commands, the robot will then execute the instruction to accomplish its mission. If the Nanobot was able to accomplish its mission, the score will be computed and the equivalent number of medals will be awarded to the player and next level will be unlocked.

* 1. **Mechanics**

The game will begin with only one level unlocked.

The player will encode instructions to the Nanobot and compile it. After compilation, the Nanobot will automatically execute the commands and travel within thin the game map.

Each level requires the previous level’s mission to be completed. After completing the mission, next level will be unlocked.

Stages on the other hand will be unlocked only with meeting the minimum number of medals. The player could play previous accomplished levels in order to gain their missing medals.

Scores will vary on steps, commands and errors number.

Medals will be awarded each level according to its equivalent value in the score.

* 1. **Replaying and Saving**

Replay of the current level will be accessible only right after the Nanobot completed the execution of commands. Previous performance aside from the current will not be accessible.

Levels and stages that are completed will still be accessible for the player anytime within the game application.

The game could be saved within the mobile’s local memory. Saving game will only save the current Nanobot’s status—level, stage, score and number of medals, character attribute and Nanobot name.

1. **Story, Setting, and Character**
   1. **Story and Narrative**

The story started in year 2100 when computers are most advanced as ever. But users suddenly experienced an error with their computer system and no one in their time is able to fix it. The person who can only fix this bug is a geek from the past. Fortunately, in their time, the time machine that is able only to return in the past was already invented. Their most knowledgeable IT engineer used the time machine in order to get that geek person from the past and bring him to their time. He brought along with him the most advanced robot they have—Nanobot.

The Nanobot has the ability to learn new instructions that could be combined with his previous ones. Its size is about an inch that the IT engineer was able to shove it inside his pocket.

After his travel through time, the IT engineer arrived to his destination—year 2016. He was inside a classic computer laboratory that has flat screen monitors and their computer units are so spacious. He then found a middle-aged man who is wearing a wide, thick glasses and his hair was like a bird nest. He remembered that man from the pictures in their time’s most successful company. He realized that the man is the one who invented the computer technology that they possess in year 2100.

He knew this was the man he was looking for. He hurriedly approached the man and introduced himself. He explained the tragedy of their time. Understandingly, the man did not refuse to help him.

After arriving back in year 2100, he introduced the device that they suspect to be from where the error first emerged. He also introduced the man the Nanobot that would help him solve the problem. Leaving the man in the laboratory, he entrusted him the future of the future world.

* 1. **Game World**
     1. **General look and feel of world**
        1. **(Areas)**

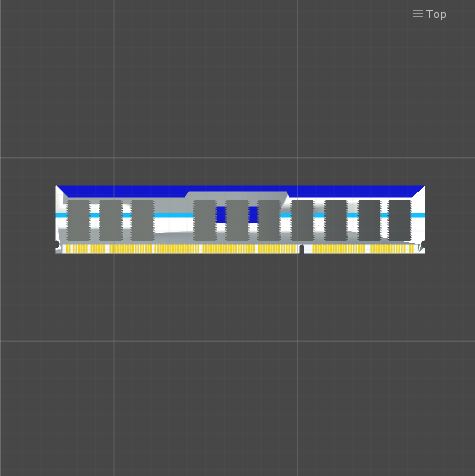
**Background:** the background has been design resembling from the internal look of a circuit board

**Game worlds:** All four stages have different game worlds

**Tutorial Stage:** Random Access Memory (RAM)

Random Access Memory (RAM) is a type of data storage used in computers that is generally located on the motherboard. It is a type of memory that is volatile and all of the information stored was lost when the computer is turned off. Physically, it is a green circuit board with a gold plated edge that plugs into a RAM port of a motherboard.

This will be displayed as a tiled area consists of 3 rows.

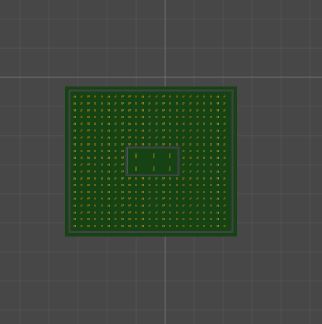


**Stage 1:**  Central Processing Unit (CPU)

Central Processing Unit (CPU) is commonly known as the brain of the computer, which handles and controls all the instructions cycling to the system of a computer. It is a silicon integrated circuit, sometimes called a computer chip. This is because it consists of a square piece of crystallized silicon that is very, very thin.

It has a very intricate pattern on top, which consists of the transistors and wires that have been processed into the silicon: a silicon chip contains millions of tiny transistors connected by very tiny wires. The chip is also connected to the rest of the computer with very tiny strands of wire.

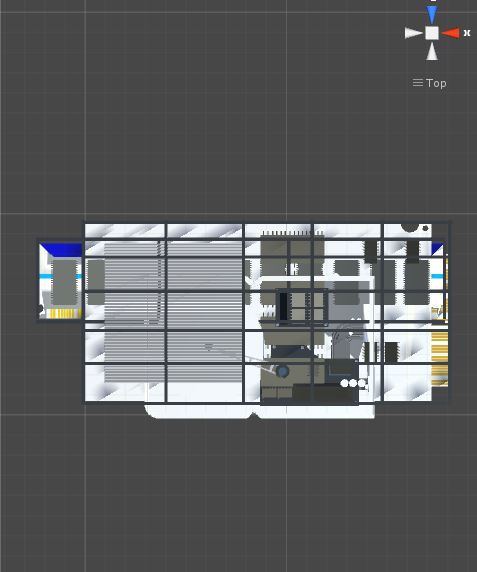
This will be displayed as a tiled area consists of 3 rows and 3 columns.



**Stage 2:**  Video card

Video card is a printed circuit board controlling output to a display screen. It is also referred as graphic cards.

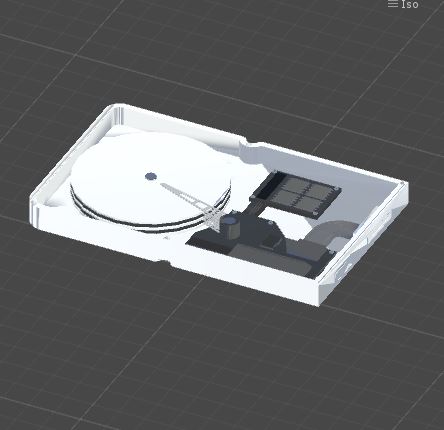
This will be displayed as a tiled area consists of 5 rows and 5 columns.



**Stage 3:** Hard Disk Drive

Hard Disk Drive is the main and usually the largest data storage hardware device in a computer. It is usually with the size of a paperback book but much heavier and the sides of it has a pre-drilled threaded holes for easy mounting in the drive bay in computer case.

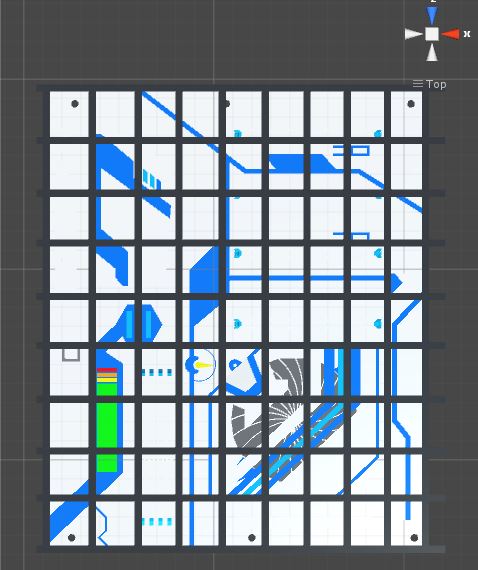
This will be displayed as a tiled area consists of 7 rows and 7 columns.



**Stage 4:**  Motherboard

Motherboard is the one of the most essential part of the computer system. It holds together many of the crucial components of a computer such as Central Processing Unit (CPU), memory, and connectors for input and output devices. On a typical type of it, the circuitry is imprinted or affixed to the surface of a firm planar surface.

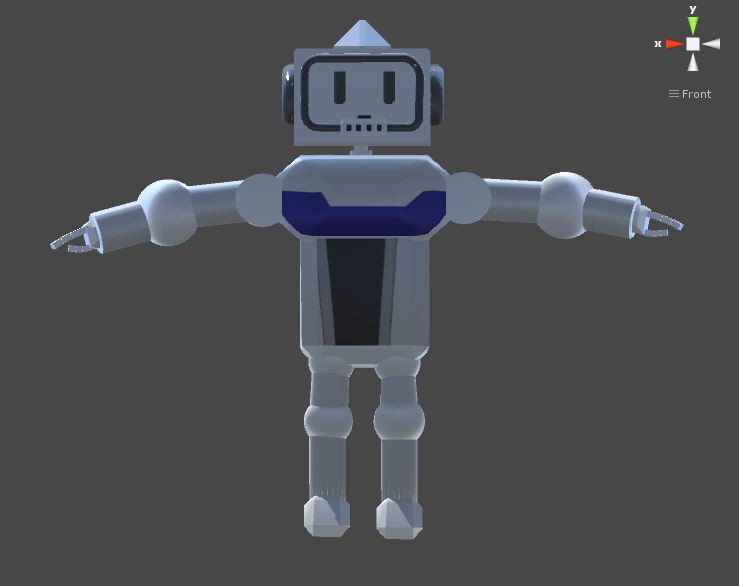
This will be displayed as a tiled area consists of 9 rows and 9 columns.



* + - 1. **(Characters)**

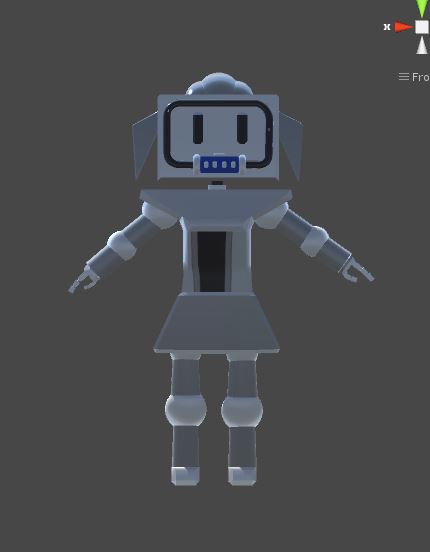
**Male:**

It is a Nanobot, which will be programmed by the geek. It has an animation of walking, turning, jumping, picking up object and removing the bug. It has a masculine texture for its body components.



**Female:**

It is a Nanobot, which will be programmed by the geek. It has an animation of walking, turning, jumping, picking up object and removing the bug. It has a feminine texture for its body components.



1. **Levels**
   1. **Levels**

**Stage 1:**

**Level 1-3:**

For the levels 1-3 which is considered as the basic levels of the game, the player will expect a single tiled CPU in the game world and as the player passes the levels, the higher the level, the far the dynamic allocation of bug in the tiled CPU will occur.

The player has also a set of commands that can be used to program the movement of Nanobot. The commands are the Walk Forward, Turn Right, Turn Left, and Fix.

Also, the levels have the Run command, which will contain the set of commands made by the player.

**Level 4-6:**

For this level, the player will expect a 2-tiled CPU for the harder thinking of strategy to make a set of commands. The options for set commands are the same as with levels 1-3 as well as the Run command.

**Stage 2:**

**All levels:**

The option for the set of commands the player can use will have an additional “Jump command”.

The set of commands made by the player will be a subject for evaluation, this evaluation will check if the commands are achieved the shortest path, also the number of commands that the player can make is limited.

The “Command 1” box is present, which is an additional container for the set of commands the player want to make.

The dynamic allocation of bug in the tiled video card will be farther as the level arises.

**Stage 3:**

**All levels:**

The player will be given an additional option command, which is “Pick command”.

The Nanobot is needed to have first the tool before removing the bug. This present in all of the levels of this stage. The allocation of the tool in the tiled Hard Disk Drive (HDD) is dynamic as the player load the level.

**Stage 4:**

**All levels:**

The “Command 2” box is present, which is an additional container for the set of commands the player want to make.

Also it has an added challenge for the player to fight with the bug after the player reaches the tile location of the Bug. The player will fight the Bug by the use of tapping the screen, as the level arises the bug becomes stronger, so the tap needs to be faster to defeat the bug. This feature will be presented to the player in a way of augmented reality.

* 1. **Training Level**

This level will train the player how will be the commands will function in the Nanobot.

This is done by the use of a tiled Random Access Memory (RAM).

1. **Interface**
   1. **Visual System**

The Head Up Display (HUD) includes:

“Back button”, which is responsible to go to the levels option.

“Restart button”, this will restart the current level, which will give a dynamic allocation of the Bug or tools in the game.

“Audio mute”, this will allow the user to enable or disable the sound effects of the game.

“Run Command button”, this button is responsible for execution of the set of commands made by the player.

“Set of available commands”, this is the set of movements or commands that the player can use for the Nanobot to move or attain its goal.

“Command containers”, these are the available boxes that will handle of contain the set of commands that will be made by the player.

* 1. **Control System**

The player can control the Nanobot by having the set of commands and executing it by clicking the “Run” button. The set of commands that the player can use are:

**Walk forward -** This will move the Nanobot to the next tile in front of it.

**Face Right –** This will turn the rotation of the Nanobot 90 degrees, this will make the Nanobot to face to its right.

**Face Left –** This will turn the rotation of the Nanobot Negative (-) 90 degrees, this will make the Nanobot to face to its left.

**Jump –** This will make the Nanobot jump to the next tile if the is an object elevated to it.

**Condition –** This will make the Nanobot decides whether what to do when encounters the Bug or the tool in the same tile location.

**Run –** This will execute the set of commands given by the player.

* 1. **Audio, Music, sound effects**

The game will have background music while player is not yet in the game play scene. If the player is in the game play scene, the sound effects are only available which is can be enabled or disabled by the player.

* 1. **Help System**

The game has a help system that will inform the player of the invalid sequence of commands or the commands that is invalid to use base on the different scenarios.

1. **Artificial Intelligence**
   1. **Opponent and Enemy AI**

**Enemy AI**

This will decides when will be the best time to counter attack the player. This is present in Stage 4, which has an additional feature to have a fight between the Nanobot and the Bug.

**Shortest path**

This will automatically generate the shortest path base on the location of the bug and the tool.

1. **Technical**
   1. **Target Hardware**

Android Device with 4.3 Operating System Versions

* 1. **Developmental hardware and software**
* **Hardware**
* Laptop
* Mouse
* External Hard drive
* **Software**
* Unity 5
* Blender
* Cinema4d
* Adobe Photoshop CS3
  1. **Network requirements**

No network requirements used nor needed.

1. **Game Art**

The game was developed using Unity 5 and other modeling tool such as Cinema 4d and Maya LT 2016.

The game has an intended style of gaming rig PC units.