

CHALMERS



Final Report

Design and implementation of a single-player first-person shooter game using XNA game development studio

Master of Science Thesis in the Department of
Computer Science and Engineering

Hatice Ezgi TUGLU
Kahraman AKYIL

The Author grants to Chalmers University of Technology and University of Gothenburg the non-exclusive right to publish the Work electronically and in a non-commercial purpose make it accessible on the Internet.

The Author warrants that he/she is the author to the Work, and warrants that the Work does not contain text, pictures or other material that violates copyright law.

The Author shall, when transferring the rights of the Work to a third party (for example a publisher or a company), acknowledge the third party about this agreement. If the Author has signed a copyright agreement with a third party regarding the Work, the Author warrants hereby that he/she has obtained any necessary permission from this third party to let Chalmers University of Technology and University of Gothenburg store the Work electronically and make it accessible on the Internet.

Design and implementation of a single-player first-person shooter game using XNA game development studio

Hatice Ezgi TUĞLU
Kahraman AKYIL

© Hatice Ezgi TUĞLU, October 2010.

© Kahraman AKYIL, October 2010.

Examiner: Per ZARING

Chalmers University of Technology
University of Gothenburg
Department of Computer Science and Engineering
SE-412 96 Göteborg
Sweden
Telephone + 46 (0)31-772 1000

Department of Computer Science and Engineering
Göteborg, Sweden October 2010

HUMANKILLERS

Will you keep your promise?

Abstract

“Humankillers” is a name of the game that was developed for Master Thesis in Computer Science Department at Chalmers University of Technology. The game is a 3D single player first-person shooter game creating using C# in XNA game development studio. The aim of this document is to describe how the problem was analyzed, development methods, development processes carried out for creating a working product and provides information on what went right and wrong about the project and the lessons learned from experiences that have been gained during a 20-week project period and our discussions.

Table of Contents

- Abstract 3
- 1 INTRODUCTION 5
 - 1.1 Background..... 5
 - 1.2 Project Objectives & Deliverables 6
 - 1.3 Project Organization 6
 - 1.4 Definitions, Acronyms and Abbreviations 6
- 2 TECHNICAL SUMMARY 7
 - 2.1 System to be developed 7
 - 2.2 Intended Users 8
 - 2.3 Development Tools and Equipment 8
- 3 METHODOLOGY 9
 - 3.1 Proposed Development Process..... 9
 - 3.2 Development Process 10
- 4 RESULT 11
- 5 DISCUSSION 12
 - 5.1 What Went Right 12
 - 5.1.1 A Clear Vision..... 12
 - 5.1.2 Great Communication and Team 12
 - 5.1.3 3D Models 13
 - 5.1.4 Physics Engine 13
 - 5.2 What Went Wrong..... 13
 - 5.2.1 Limited Time..... 13
 - 5.2.2 Late Feature Addition..... 14
 - 5.2.3 Integration of 2D and 3D 14
- 6 CONCLUSION 15
- 7 REFERENCES 16
- Appendix A – Screen Shots 17
- Appendix B – Project Plan 23
- Appendix C – Game Design Document 34
- Appendix D – Test Plan 57
- Appendix E – Test Protocols..... 79
- Appendix F – Gantt Chart 95

1 INTRODUCTION

“Humankillers” is a 3D FPS game is created as a master thesis project for Computer Science department at Chalmers University of Technology. The aim of this document is to describe the development process and the results of this project .

1.1 Background

Despite the economic instability and crisis deeply affecting the world, the analysts published that the game industry has grown at a rate of 57% surprisingly. Even as we type these words, millions of people are playing game in front of their computers. The reason of this growth can be stated that the game industry can appeal any users with different tastes. Microsoft realized the opportunities in this field and developed a new technology-XNA game development studio for game developers.

The idea of developing game is not new for us. It was one of the options for our senior project of bachelor degree at our home university. However, our team consisted of four students and other two did not want to take such a risk. Because, XNA was a new technology and none of us was familiar with game development. Thus, we had to waive a chance to implement our ideas.

Actually, at the begining of the project we had still no experience in developing a game. But, this time there was a big difference that we were be more determined than ever. As everyone knows, there are several types of computer games. First person shooter is one of the best-known game genres. A first-person game is a game which makes the player feel within the game world; as if the main character has the game camera fixed at his eyes. After the success of Doom which is accepted as the first FPS game, many companies took a crack at this type of game. Nowadays, the best seller game is a kind of first-person shooters.

Consequently, we decided to achieve as our master thesis work what we did not dare for our senior project.

1.2 Project Objectives & Deliverables

The purpose of the project is to design and implement a 3-dimensional game written in C# using XNA game development studio. The project includes a complete level of game with documentation. The level will include everything that should be available in a FPS game. The game will be a single-player. The team members don't take aim at developing an instructive game; instead the aim is action, action and more action.

Project Deliverables:

- Project Plan
- Game Design Document
- Test Plan
- Test Report
- Final Report
- Product
- User Manual

1.3 Project Organization

Because the team consists of two members, we don't define any specific role for any members. Each member takes responsibilities of each role in each phase to guarantee that the project can continue.

Please refer to section 4.1 in Project Plan [1] for role specific responsibilities and refer to Gantt Chart document [2] for detailed work distribution of the project.

1.4 Definitions, Acronyms and Abbreviations

Term	Definition
Humankillers	Name of the game
FPS	First person shooter
GDD	Game Design Document
3D	Three Dimensional

2 TECHNICAL SUMMARY

2.1 System to be developed

“Humankillers” is an action based 3D first-person shooter game where the player takes role of the main character to destroy enemies. The game is inspired from a science fiction novel “I am Legend” by Richard Matheson, which tells the story of the last man in the world where entire humanity has returned into mindless human killers. New and crowded community afraid of him and ones like him and is trying to destroy them. The game starts exactly at the point where the main character of the novel realizes that the only way to protect from them is to kill all of them.

The gameplay is action-based with no strategic or role-playing elements; instead, the game entirely provides a rush of adrenaline. The only goal is to kill all enemies until even one is not left before they kill our character.

The version in this project consists of one level. We will add upper levels in our future works. The game starts in New York City, where the end begins. In addition, the game involves no checkpoints; the player will start the game at level entries.

The player plays these levels from a first person view in a 3D environment. The player experiences the game world through the eyes of a main character. This makes one feel one is a part of the game.

The player controls the main character named Dr. Hero, who is one of the healthy survivors. The abilities of our character as a player are walking and running in all four directions, crouching, jumping, using-reloading-changing weapons, and collecting the pick-ups.

The game has a design with only single-player mode. In the single-player mode one can find three sub-modes: easy, normal and hard. Depends on sub-modes, the number of enemies and pick-ups will vary. When the game becomes harder, the number of enemies will increase and the number of pick-ups will reduce.

The player starts the game with two weapons and certain number of ammo. During the game, one has the ability to pick up more weapons and ammos. Along the way one face off against certain number of enemies which depends on the type of sub-mode.

By default, the player has one hundred health units. The attack of enemies reduces the health primarily. The effect of damage varies depending on the type of attack. The enemies can bite; they do not have the ability to use gun and shoot. However, the player can pick up the power-ups that increase health. The player has only one life. When his/her health reduces to zero or lower, one will die. When our main character dies, the player loses the game. If the player kills all the enemies before the health reduce to zero or lower, one wins the game.

Please refer to Game Design Document [3] for detailed information about the game.

2.2 Intended Users

Because of violent activities, the system/game is mainly intended for adults (age 16+) who also are referred to as players.

Players	Intended users for the game
Project Examiner	Responsible for project evaluation
Team Members	Responsible for system development

2.3 Development Tools and Equipment

Here is the list of tools and equipment used to develop the proposed system.

- Microsoft Visual C# 2008 Express Edition
- XNA Game Studio 3.1
- Blender
- Softimage Mod Tool 7.5

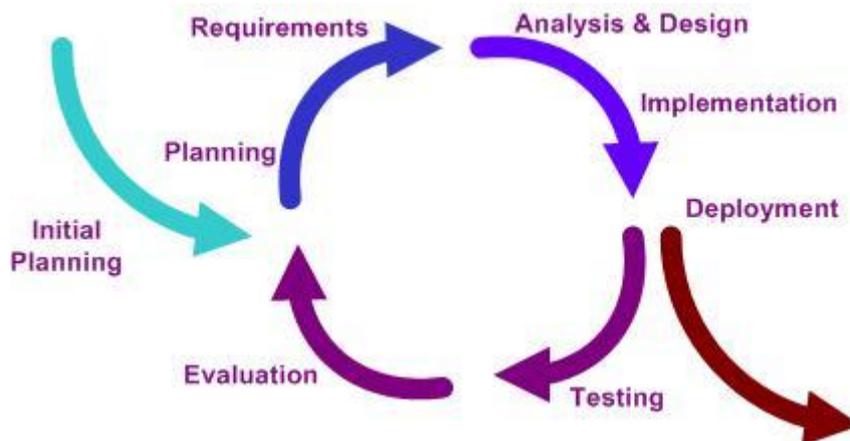
3 METHODOLOGY

3.1 Proposed Development Process

Even if one can prepare a well-thought-out design document, some features do not give the same effect in gameplay as on the paper. During implementation phase many features are added, removed or modified. Accordingly, one needs to make some modifications on game design and requirement analysis. Thus, one of the most suitable development methodologies for game development is iterative development process. Most of game developers [4] [5] have the same idea about the advantages of iterative development process for game developers.

As Ian [6] said, “The more times you iterate, the better your final game will be”.

We prefer the iterative development process with specified milestones and deliverables for our project.



3.2 Development Process

We planned the project over a period of 20 weeks and divided it into four iterations. We planned the first iteration for planning, second iteration for game design, third iteration for coding and the final iteration for testing and finalizing the product.

In the first iteration, we focused on Project Plan determined as the first planned milestone of the project as we mentioned in Project Plan document. Planning is essential for starting of upcoming milestones and delivering a finished project on time. Successful completion of a project is heavily dependent on effective planning. Iteration 1 was set to approximately one week (8 days) to figure out the detailed activities in order to minimize risk to the final result and delivery.

The second iteration started by brainstorming among group members on what the game would be. Each group member denoted the attributes or properties of the game that one dreamed to implement. We gathered suggestions together and chose the ones that was possible to be implemented within a 20-week-project time. As soon as the game concept became clear, we made some early decisions on basic requirements of the project in order to more easily reach the development goals. GDD was also the main deliverable in this iteration. Game design document was meant to be a living document. In other words, throughout the production process the document was updated, if needed.

As we had GDD in hands, it was used as baseline to start implementation. In the third iteration coding in C# using XNA game development studio was under way. None of us had experience in game programming. Therefore, most of time in this iteration was dedicated for internal training sessions. In this iteration, we needed to achieve four milestones each was dependent on the previous one.

GDD was also in favor of Test Plan. Last iteration was planned for testing and finalizing the product. The testing process is an iterative process. We performed the testing process in four iterations. The successful testing process of software requires a good plan. Therefore, after the requirements of the project are confirmed, the future testing of the system and the code were planned. The test plan provided information on how and when the testing will be executed. In the second iteration, test cases were designed for the planned tests. In iteration three, the designed test cases were executed alongside the module testing and usability testing. During the last iteration, according to the result of the tests, the test reports were documented properly and the bugs were reported after the testing is completed.

4 RESULT

Concerning the managerial aspects, the members acquired experience in how to apply an iterative process for game development. Due to the fact that the group consists of two members, both have the responsibilities of all the roles for continuity of the project. Therefore, both team members participated in producing each document. The responsibility of steering the group and keeping the project on track is on both members' hands.

The result of the first iteration of the process is the documentation about project planning that is necessary for the successful progress. Project Plan has not been updated and has been used throughout the process.

Because of lack of experience in game development, we faced with some difficulties while producing the game design document. All members of the group must have the overall idea of the game concept and mechanics clearly to do implementation. At the beginning, we could not estimate which properties could be implemented within project time. On this account, the requirements as presented in the initial version of GDD are based on early decisions. In the upcoming stages, GDD was updated and iterated throughout the development process.

Due to time shortage, we had to put 3D graphics design and creation out of project scope. Therefore, we needed to find eligible free 3D models from the websites [9], [10] where the 3D graphics are sold. Luckily, we easily found the models in a short time and integrate them into our gameplay seamlessly.

The testing process of this project went mostly as we had planned; however, the test plan schedule was pretty much followed. Because the implementation took longer time as we expected, we had to spend less time on testing process. Due to time issues, the testing process that we planned to perform with test tool was left out. After testing finished, it was time to write the test protocols which contains the detailed information about the results of the tests and comments for each test if necessary.

The end result of the implementation is a game that almost fulfills all of its functional requirements. The only part that has not been implemented is the mini map. For a start, the mini map was not essential part of the game but more an addition to the esthetics. In our upcoming works, we can improve the city map, accordingly add a mini map and make our game as bigger as the games in the market. Please refer to Test Protocols [5] for the detailed information about the parts that has problems.

The final stage was used to create the compulsory final report. We established a document template according to the Chalmers University standards [8] and fill out the information about the progress, results of the project, but also our vision about the project.

5 DISCUSSION

5.1 What Went Right

5.1.1 A Clear Vision

As Marcus Annaeus Seneca said, “If a man does not know what port he is steering for, no wind is favorable to him.” From the beginning, we have had a grand vision of what the game would be. This was possible for the following reasons.

In game market, there exist a considerable number of FPS game samples. We had a chance to observe existing games .This was a good start to think about setting out the game. While informing the design of the game, we took notice of the common features of these games. However, while making decision about game genre, we thought with a trader brain. We followed best-seller lists and decided to create FPS game because of the successful sales chart. In the next steps, this decision has turned into an advantage for us.

In addition, for the back story we inspired from the best-known science fiction novel. Being the novel that was adapted to film in different ways, provide us to form a picture of gameplay into our brains clearly. The gameplay was not enigma for us anymore. We know that the FPS lovers are not interested in back stories. Thus, we keep our game simple without overcharging the player with back stories.

The lesson learned from this is that one cannot achieve the goals, if he does not know what they are. If one wants to create a game without any pain, one has to be clear about what to create.

5.1.2 Great Communication and Team

The biggest problem for most of teams, communication, was one of the best for our project. The main reason is that we live in the same apartment. Throughout the project when one of the team members had difficulties or thought something unclear, the other was a step away.

We know each other for a long time. We have taken place in many projects including senior project together. Working as a team is not new for us and we could use the benefits of this situation.

Both of us were aware of undertaking a hard work and focused on success. To make a great game, we worked hard and regular. Through each made the other feel his/her support, we provided the project end as well as possible. We feel lucky about having a chance to be in such a project together.

5.1.3 3D Models

Because of limited time, the design and creation of 3D models are out of scope of the project. We planned to use free ready-to-use 3D models in the project. However, at the beginning this was a nightmare for us. If we did not find suitable models, making a game would be a fantasy and accordingly the project would fail. Fortunately, we found the suitable models in unexpected short time. We selected models from free models on the web sites selling models. At the end of the project, we realized that while we think finding 3D models as a nightmare, it becomes a stage that gives rapid results and makes us very happy.

5.1.4 Physics Engine

We used JigLibX physics engine [7] which is one of the favored open-source physics engines for physics and collision detection. Using physics engine was a one of the best decisions about the project. We did not encounter any problems about the engine although it was our first time to use physics engine and we had some doubts because the engine is beta version. The engine allowed us to focus more on gameplay, instead of spending time on creating our own physics library.

5.2 What Went Wrong

5.2.1 Limited Time

The main problem for the project was limited time. Although the project is small, we think that 63 days are not enough for implementation. Due to the fact that we had no game development experience, we had to do a considerable amount of research during implementation. This did lead to slow down the progress of implementation of the game. Fortunately, we did not come up against the case of unfinished implementation work. This situation resulted in just putting increasing stress on us and acting as if we had panic attacks during the entire implementation phase. We felt the impacts of limited time during not only implementation stage but also testing stage. We did not use test tools because of lack of time combining with the case of being unfamiliar with test tool as we defined as a risk in Test Plan. The lesson learned is that the game development needs more than 20 weeks.

5.2.2 Late Feature Addition

After alpha phase was end, we realized that we left one important point out of account while documenting the design of game. When all the weapons that the player had were out of bullets and collectable ammo packs and weapon packs were not left, the player did not have any equipment to defend himself and was waiting to be killed despairingly. Unfortunately, this was more than enough to raze charm of the game.

The easiest solution was to add a weapon that did not need bullets. First, we tried to add knife. However, we could not integrate free 3D knife model that we found into our game world. Thus, we had to give up this option because we did not have enough time to waste and then we started to think about new options. Finally, we decided to use punch animation. After integrating into the game, we accordingly updated our design document and test cases. We learned how right ones who suggest iterative development process for game development are.

5.2.3 Integration of 2D and 3D

We built our game in a 3D world and our menu system in a 2D platform. Our game and menu systems were working properly before integration. While they were working separately, everything was perfect. When they came together, the problem showed up. We found out very good sample codes for 3D game world and GUI. We reviewed all the samples we could find. However, almost all were for the sections where we have already not encounter any problem. We could not foresee how difficult to find a simple sample showing the way of the two working together.

The leading cause was that we implemented the 3D world and GUI as two separate game frameworks. Due to being two different games, XNA game engine selected one of them and run it.

We fixed the problem by referencing the game framework including menu systems to the one including core game.

6 CONCLUSION

Despite the fact that we have involved in many projects up until now, this project is our first game development project. Both of us were very inexperienced in this field. Although we both have a little bit of fear because of our naivety, this was an unquestionably important experience for us. In the following section, we mentioned about the lessons that we have learned during this project process.

Having regard to the suggestions of the authors who wrote the articles about game development process, we decided to apply iterative development methodology to our project. Especially because of two reasons, we realized that this is a very fortune decision. We faced with the first reason causing us think like that during the preparation of GDD. At the beginning we had a very clear vision about what we wanted to implement. Our game would roughly be a single player FPS game with a known back story inspired from a well-known novel. However, due to lack of experience we were not able to forecast how many percentages of the game functional requirements could be implemented within a prescribed project period. We established the final version of the GDD after a few updates. The second was during the testing process. As we know, the testing is an iterative process. Once the bugs are fixed, the testing has to be done repeatedly. In addition, while checking the test cases we realized that there existed some test cases missing. Accordingly, we went back and made some modifications and additions on Test Plan. Shortly, the chosen development methodology is the most suitable for our project without a shadow of a doubt.

At the end of the project, the other lesson that we learned and think to apply to our future projects is making a good project plan, remaining true to this plan as possible and documenting everything we do. By doing this, whenever we have a problem we can go back and see what we did the last time we had a similar problem.

Throughout the project, the biggest problem was limited time. Many times during the project we had the devil of a time because of time shortage. The fact that we had examinations addition to our thesis work caused us to take breaks, albeit short ones. After these breaks, we had some difficulties in adapting to the thesis again.

We, who developed a game for the first time, think that the decision on game genre is one of the most suitable decisions that we have made throughout the project. There exist so many alternatives to examine in the game market because of intense interest of the people in FPS games. Therefore, it was a good start for us. Our thought is “the more examples one has near at hands, the easier the implementation will be.”

As a result, 20 weeks were full of new great experiences and these experiences that we have gained will help us in our upcoming projects. Finally, the idea of developing game that was a dream for us since our bachelor education became real and we have experienced the pride of doing such a thing at the end of such an education.

7 REFERENCES

1. Project Plan-Appendix B
2. Gantt Chart- Appendix F
3. Game Design Document- Appendix C
4. Test Plan- Appendix D
5. Test Protocols- Appendix E
6. Schreiber, Ian, “Level 2: Game Design / Iteration and Rapid Prototyping”, 02 July 2009. 02 June 2010, <<http://gamedesignconcepts.wordpress.com/2009/07/02/level-2-game-design-iteration-and-rapid-prototyping/>>
7. 25 May 2010, <<http://jiglibx.codeplex.com/>>
8. 25 May 2010, <http://wiki.portal.chalmers.se/CHOCS/pmwiki.php/Resources/Writing>
9. 24 March 2010, <http://www.thegamecreators.com/>
10. 24 March 2010, <http://www.turbosquid.com/3d-models/zombie-maps-normal-max-free/327193>

Appendix A

Screen Shots

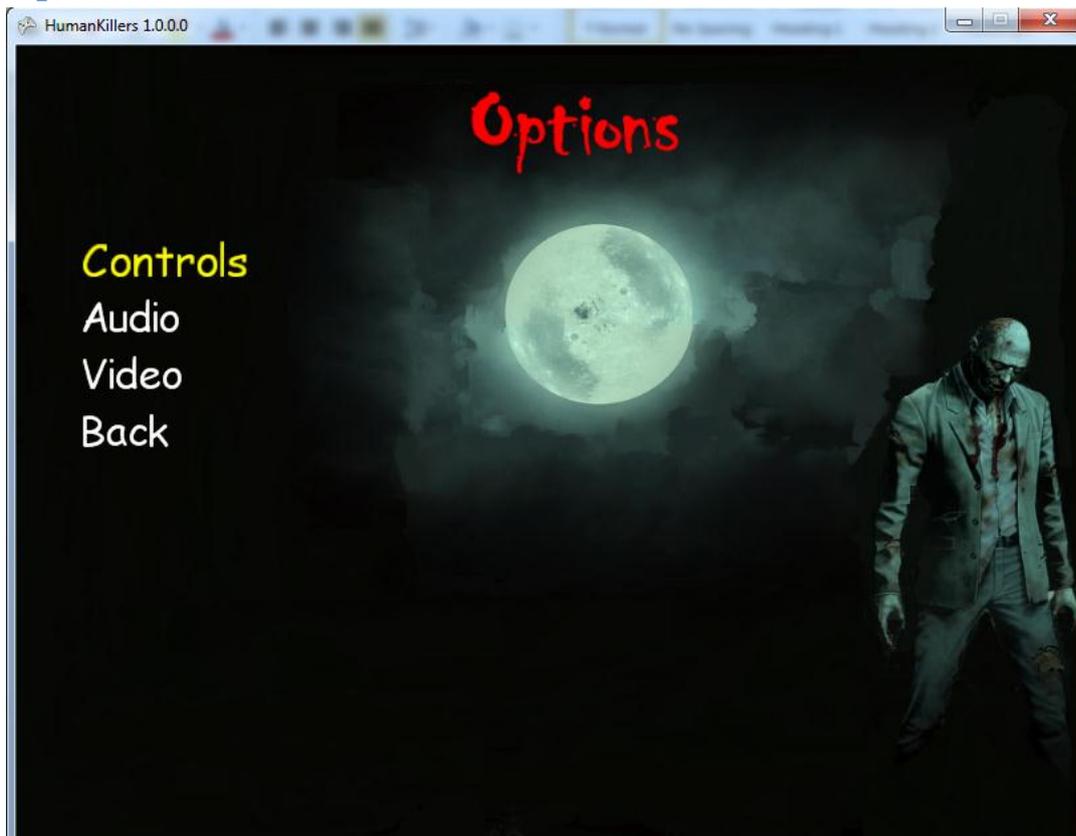
Main Menu:



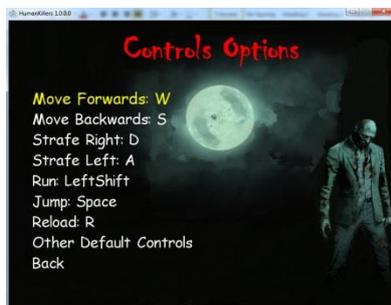
New Game Menu:



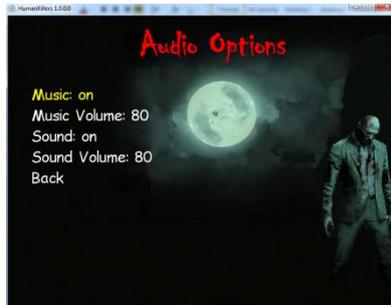
Options Menu:



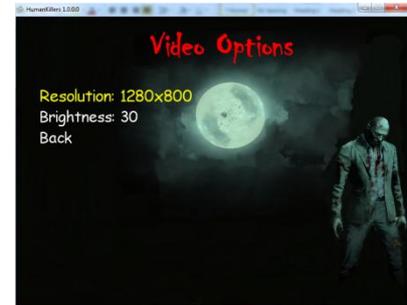
Control Options Menu:



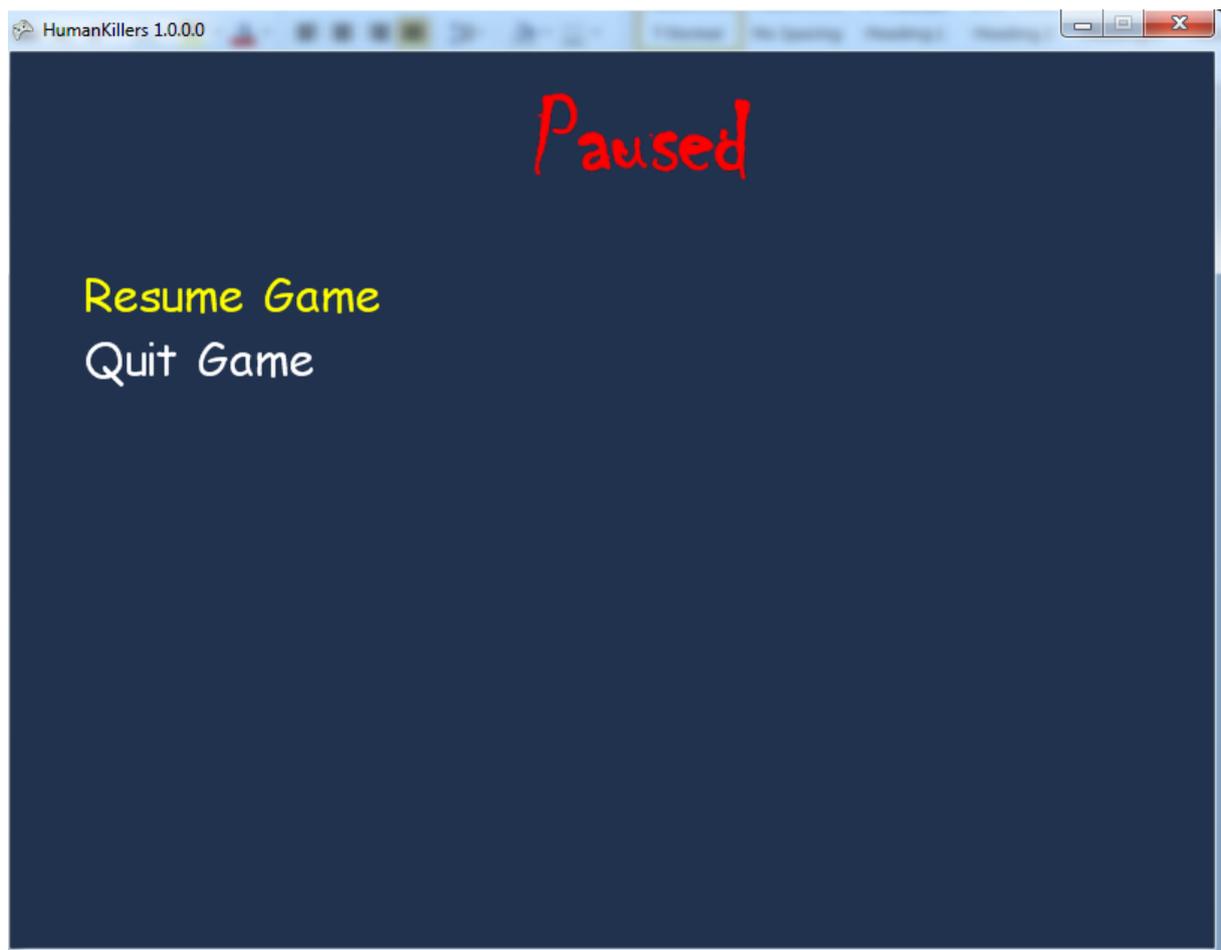
Audio Options Menu:



Video Options Menu:



Pause Menu:

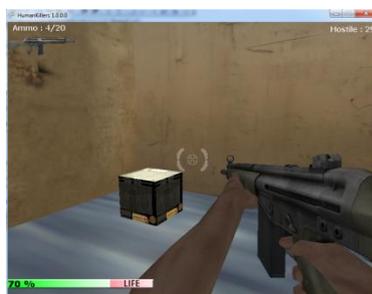


Enemy and HUD:



Pickups:

Ammo Pickup:



Weapon Pickup:



Health Pickup:

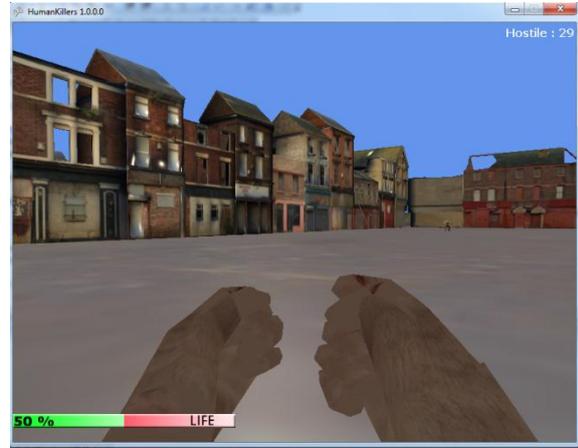


Terrain:

Terrain Sample1:



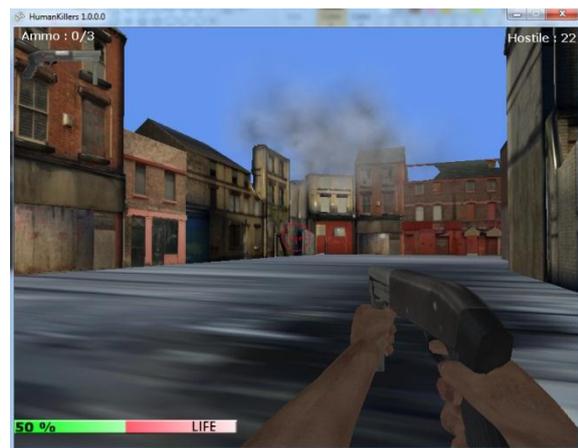
Terrain Sample2:



Terrain Sample 3:



Terrain Sample 4:



Terrain Sample 4:



Terrain Sample 5:



Appendix B

Project Plan

1 PROJECT OVERVIEW

1.1 Background

Despite the economic instability and crisis deeply affecting the world, the analysts published that the game industry has grown at a rate of 57% surprisingly. Even as we type these words, millions of people are playing game in front of their computers. The reason of this growth can be stated that the game industry can appeal any users with different tastes. Microsoft realized the opportunities in this field and developed a new technology-XNA game development studio for game developers.

As everyone knows, there are several types of computer games. First person shooter is one of the best-known game genres. A first-person game is a game which makes the player feels within the game world; as if the main character has the game camera fixed at his eyes. After the success of Doom which is accepted as the first FPS game, many companies took a crack at this type of game. Nowadays, the best seller game is a kind of first-person shooters.

As we are two master students of Chalmers University of Technology, we decide to select topic for our master thesis in the field full of opportunities. In this way, we can use an opportunity to further gain skills and specialize in game development before stepping into the game industry.

1.2 Purpose

The purpose of the project is to design and implement a 3-dimensional game written in C# using XNA game development studio. The project includes a complete level of game with documentation. The level will include everything that should be available in a FPS game. The game will be a single-player. The team members don't take aim at developing an instructive game; instead the aim is action, action and more action.

Because of the project, the team members will acquire experience in developing a game and increase our skills in different areas of game development project.

1.3 Deliverables and Documentations

- Project Plan
- Game Design Document
- Test Plan
- Test Report
- Final Report
- Final Product
- User Manual

1.4 Project Scope

In Scope

The project will be based on creating an FPS game with the goal in mind of being fun.

- Single Player
- PC based
- 3D platform
- Single level
- Action-based
- Science-fiction
- 2D platform for GUI and menu systems
- Testing for 0 bugs in game
- Written in C # for XNA

Out of Scope

- Will not be multiple player
- Will not be consoled based
- Will not be multi-level

2 PLANS

2.1 Milestone Plan

The table below provides information on milestones of the project.

Milestone	Scope	Output	Date
Milestone 1	Pre-production	Project Plan	March 17,2010
Milestone 2	Game Design	Game Design Document	March 22,2010
Milestone 3	Test Planning	Test Plan	March 25,2010
Milestone 4	First-playable	Working prototype of game	April 28,2010
Milestone 5	Alpha Testing	Prototype of game updated with AI and some features as animation etc.	May 16,2010
Milestone 6	Beta Testing	Prototype of game updated with user interface and music	May 24,2010
Milestone 7	Game Release	Test Report/Final Product	June 1,2010
Milestone 8	Post-production	Final Report/ User Manual	June 3,2010

2.2 Project Phases

The table below provides information on the activities and expected documented output from each project phase.

Phase	Activities	Output
Documentation Phase	Project Plan	ProjectPlan.pdf
Game Design Phase	Game Design Document	GameDesignDocument.pdf
Pre-Alpha Phase	Test Plan, Basic world, Controls	TestPlan.pdf
Alpha Phase	Collision Detection, Physics, Animation, Enemy AI	Alpha Version Product
Beta Phase	Player interface, Sounds and music	Beta Version Product
Postmortem Phase	Bug fixed, Integration Test, System testing	TestReport.pdf/Humankiller.exe(Final Product)
Post-Product Phase	Final Report, User manual	FinalReport.pdf/UserManual.pdf
Presentation Phase	Present the project	Presentation.ppt

2.3 Documentation Plan

The table below provides information on dates when the documents related to the project will be delivered.

	Purpose	Date
Project Plan	Organize the project	March 17,2010
Game Design Document	Describe the design of game	March 22,2010
Test Plan	Define all test activities	March 25,2010
Test Report	Define all test cases	June 1,2010
Final Report	Summarize the project	June 3,2010
User Manual	Describe how to play the game	June 3,2010

2.4 Time Schedule

Please refer to the Gantt chart document (GanttChart.gif) for detailed time plan of the project.

3 REQUIREMENTS

3.1 Development Requirements

- Microsoft Visual C# 2008 Express Edition
- XNA Game Studio 3.1
- Blender
- Softimage Mod Tool 7.5

3.2 System Requirements

- Windows XP Service Pack 3 or later
- DirectX 9.0c or later
- Shader Model 2.0 or later
- 1 GB RAM or higher

4 PROJECT ORGANIZATION

4.1 Roles and Responsibility

The table below provides information on the roles within game development project and their responsibilities.

Project Team Role	Responsibilities
Level Designer	Will be responsible for creating a playable level for the game
Project Manager	Will be responsible for project plan & managing schedule
Developer	Will be responsible for implementing code and maintain coherency in the code
Game Designer	Responsible for game design document
Art Designer	Responsible for creating the 3d assets for the game
Audio Designer	Responsible for creating the soundtrack, special effects, and sound effects
Tester	Responsible for finding the bugs and reporting them.

Because the team consists of two members, we don't define any role for any members. Each member takes responsibilities of each role in each phase to guarantee that the project can continue. In this way, both team members will be responsible for anything related to the project. The work of each member is clearly specified in the Gantt chart document. Please refer to the GanttChart.gif for detailed work distribution of the project.

4.2 Resource Plan

The table below shows the time estimation on human resources specified by each role.

Task Name	Estimate	Resources
Level Design	200 hours	Level Designer
3D Art	200 hours	Art Designer
Game Mechanics	100 hours	Game Designer
Project Plan	80 hours	Project Manager
Development	320 hours	Developer
QA	80 hours	Tester
Audio	50 hours	Audio Designer

5 QUALITY PLAN

The goal of the plan is to ensure that “Humankillers” is an error free game. Throughout the development, testers will be brought in to test the mechanics and the levels of the game to ensure there are no errors in the game. Furthermore, testers will comment on the fun and experience they have with “Humankillers” and will give the team feedback on how to improve the game.

We will divide quality testing into three main phases:

- **Module testing** will perform during coding by using debug messages to check that the written code produces wanted results. An important requirement is that the code will compile with zero bugs.
- **Integration testing** will perform after finish module testing in order to validate if each module can work fine with each other.
- **System testing** includes two phases: functional testing and usability testing. These will perform after the product reaches its final version. During functional test phase, the tester will test if the product meets the game requirements. The usability test will perform to understand how easy it is to learn to play the game. Any person out of the team members will perform this test by playing the game.

6 RISK ANALYSIS

6.1 Overview

The major risks to this project will be time, and whether the objectives of the vision will be succeeded. Some changes to the scope of the project will be acceptable as long as it is delivered on time but any overtime from presentation date will not be acceptable and can drop the project. Due to fixed time, the amount of man hours needed for the project can be increased to complete the project.

6.2 Project Risks

- High Risks
 - New to programming for XNA
 - Find more sample games built in XNA and try to understand
 - Problems with meeting the milestone deadline
 - Work harder to meet the deadline
 - Unrealistic project scheduling
 - Re schedule the project

- Medium Risk
 - Problems with graphics and audio
 - Try to find eligible graphics and audio
 - Problems within development environment
 - Review and re prioritize functional requirements

- Low risks
 - Insufficient effort of team members
 - Take a little break for motivation

7 TEAM INFORMATION

7.1 Team Members

- Kahraman AKYIL
kahramanakyil@gmail.com
840507-0718
Networks and Distributed Systems
- Hatice Ezgi TUGLU
het.ezgi@gmail.com
850912-3041
Software Engineering and Technology

7.2 Examiner

- Per ZARING
per.zaring@chalmers.se

Appendix C

Game Design Document

Design History

Version 4	○ Game Modes section updated
Version 3	○ Development Section added ○ Project Overview section updated
Version 2	○ Game Concept Section updated ○ Game Mechanics Section updated
Version 1	○ Initial version of Game Design Document.

1 PROJECT OVERVIEW

1.1 Purpose

The purpose of the project is to design and implement a 3-dimensional game written in C# using XNA game development studio. The project includes a complete level of game with documentation. The level will include everything that should be available in a FPS game. The game will be a single-player. The team members don't take aim at developing an instructive game; instead the aim is action, action and more action.

1.2 Project Scope

In Scope

The project will be based on creating an FPS game with the goal in mind of being fun.

- Single Player
- PC based
- 3D platform
- Single level
- Action-based
- Science-fiction
- 2d platform for GUI and menu systems
- Testing for 0 bugs in game
- Written in C # for XNA

Out of Scope

- Will not be multiple player
- Will not be consoled based
- Will not be multi-level
- Creation of 3D models

1.3 Team Members

- Kahraman AKYIL
kahramanakyil@gmail.com
840507-0718
Networks and Distributed Systems
- Hatice Ezgi TUĞLU
het.ezgi@gmail.com
850912-3041
Software Engineering and Technology

2. GAME CONCEPT

2.1 Overview

“Humankillers” is an action based 3D first-person shooter game where the player takes role of the main character to destroy enemies. The game is inspired from a science fiction novel “I am Legend” by Richard Matheson, which tells the story of the last man in the world where entire humanity has returned into mindless “Humankillers”. New and crowded community afraid of him and ones like him and is trying to destroy them. The game starts exactly at the point where the main character of the novel realizes that the only way to protect themselves is to kill all of them.

2.2 Story

Throughout history, every century has witnessed an epidemic of its own. For centuries, humankind was forced to deal with diseases such as leprosy, plague and tuberculosis. In modern world, cancer was accepted as the most frightening disease. It became the nightmare of the modern world until the year 2015.

In 2015, a group of army virologists had created a virus as a cure for cancer with a 100% success rate. Entire world thought that an important breakthrough was made. However, nobody could imagine that this invention would cause a global disaster and it was just a beginning for the end.

The virus mutated into a lethal weapon that spread worldwide and caused billions of people die. Only a few pockets of people (10% of humanity) could survive. Unfortunately, less than half were naturally immune to the virus. The rest turned into bald, pale like a dead body and aggressive beings were seeking the immune humans to kill.

After the great tragedy has occurred, the most developed nations of the world have returned into the “Humankiller” masses. Due to the “Humankillers” under the stimulus of hunger for all living thing, the death walk the streets all over the world. People in places with high-powered weaponry can fight against “Humankillers”, but in other places the people needs help.

Dr. Hero KAHEZ, who is one of healthy survivors, was a successful virologist working for U.S. Army. Before epidemic, he had been a happy life with his beautiful wife and 5-year-old daughter in New York. At the same time, he was a member of the group of the virologists who created a cure for cancer. Thus, he has accepted himself as a cause of this global tragedy. Because of this human-made epidemic, he has lost his family and his friends one by one. He was left as the last healthy human from his vicinity.

At the time when his wife and his little daughter died in his arms, he promised them to put a stop to it somehow before it is too late. Unfortunately, the only way to stop destroying humanity and to save the remaining healthy humans is to kill the degenerated survivors. He has devoted himself to save at least the remaining world for correcting his fault at the risk of his own life.

At this point, he will learn the true price of survival.

2.3 Locale

The year is around 2016, a few months after the global disaster. The game takes place in the world where humanity has been almost destroyed. There is no remainder of the world, as one knows anymore. Cities in all counties are as they merge into a huge ghost village. Buildings became almost useless. Humanity split into two halves desperately, those who returned into non-human beings while they were waiting for a hope of recovery from cancer and, those who are healthy humans cannot be happy to survive. However, both sides have to escape from or destroy the other side to survive. The dead silence is rampant in all the streets.

2.4 Genre

The game is a science fiction first person shooter played on the PC platform where the player controls Dr. Hero in a 3D environment against murderous “Humankillers”.

3 GAME MECHANICS

3.1 Gameplay

The gameplay is action-based with no strategic or role-playing elements; instead, the game entirely provides a rush of adrenaline. The only goal is to kill all enemies until even one is not left before they kill our character.

The game actually consists of more than one level. However, the number of levels has not been determined yet, because we will work on the first level in the first episode. In all levels, the mission will be the same as in first level, killing all “Humankillers”. The only difference will be the locations where the levels take place. Every level will take place in different cities. The game starts in New York City, where the end begins. In addition, the game involves no checkpoints; the player will start the game at level entries.

The player plays these levels from a first person view in a 3D environment. The player experiences the game world through the eyes of a main character. This makes one feel one is a part of the game.

The player controls the main character named Dr. Hero, who is one of the healthy survivors. The abilities of our character as a player are walking and running in all four directions, crouching, jumping, using-reloading-changing weapons, and collecting the pick-ups. In the later episodes, we will neither change the main character nor add any additional characters that can be controlled by the player.

The game has a design with only single-player mode. In the single-player mode one can find three sub-modes: easy, normal and hard. Depends on sub-modes, the number of enemies and pick-ups will vary. When the game becomes harder, the number of enemies will increase and the number of pick-ups will reduce.

The player starts the game with two weapons and certain number of ammo. During the game, one has the ability to pick up more weapons and ammos. Along the way one face off against certain number of enemies which depends on the type of sub-mode.

By default, the player has one hundred health units. The attack of enemies will reduce the health primarily. The effect of damage will vary depending on the type of attack. The enemies can bite; they do not have the ability to use gun and shoot. However, the player can pick up the power-ups that increase health. The player has only one life. When his/her health will reduce to zero or lower, one will die. When our main character will die, the player will lose the game. If the player kill all the enemies before the health reduce to zero or lower, one will win the game.

3.2 Level Design

We design this episode of the game with one level. In this way, one can find the details of the level design in the rest of the document. In later episodes of the game, we will add levels in which the main character will be the same and there exist additional weapons. In addition, the city for each level will vary.

3.3 Game Modes

The game has a design with only single-player mode. In the single-player mode, one can find three sub-modes: easy, normal and hard. Depends on sub-modes, the number of enemies and pick-ups will vary. When the game becomes harder, the number of enemies will increase and the number of power-ups will reduce. The table below provides information on the number of power-ups and enemies in each mode.

Mode	Enemy #	AmmoPack#	WeaponPack#	HealthPack#
Easy	30	15	6	5
Normal	40	10	4	4
Hard	50	5	2	3

3.4 Game Flow

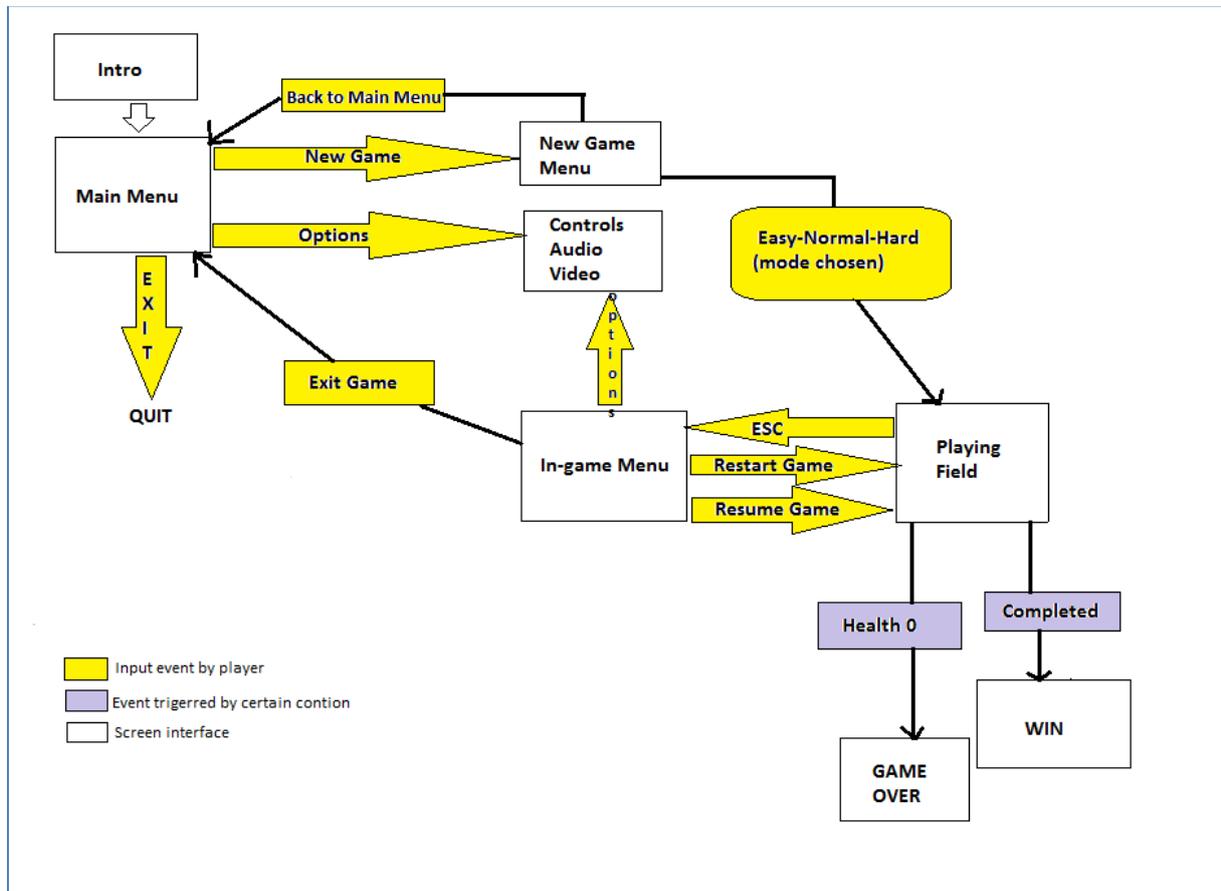


Figure: Game Flow

3.5 Game Control

The game will utilize the mouse and keyboard for input. Here is the list how to control the game.

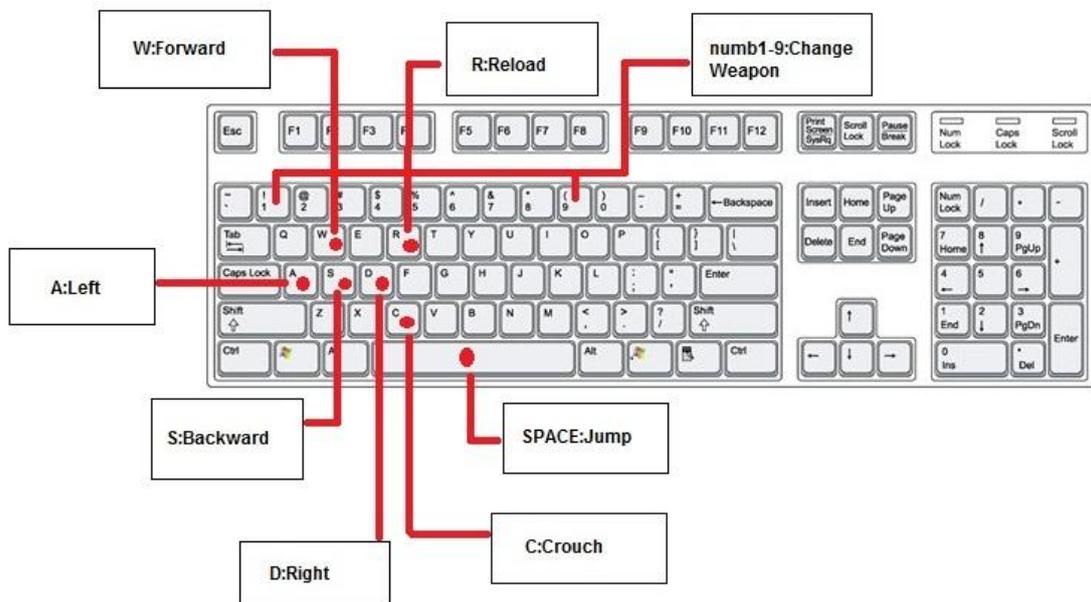
3.5.1 Movement

- Left - A
- Right - D
- Forward - W
- Backward - S
- Jump – Space
- Crouch-C

3.5.2 Actions

- Looking/Aiming – Mouse
- Fire Weapon- Left Mouse
- Use Equipment-Right Mouse
- Change weapon - numb 1-9
- Reload - R
- Use – E

See the illustration below for all keyboard operations:



3.6 Winning and Losing

The object of the game is to kill all enemy “Humankillers” before they eat the player. The player has only one life. When his/her health reduces to zero or lower, the player will die. When the player dies, he/she will lose the game. If the player shoots all enemies before dying, the player wins the game.

3.7 Power-ups

Power-ups will be collectible objects that are placed on different points on the terrain. The amount of power-ups will base on gameplay mode. There are three types of power-up: Health pack, Ammo pack, Weapon pack.

3.7.1 Health Pack

Health Pack is a kind of first-aid box used for recovering health of the player with some predefined amount. The common amount of healing is 20. They grant health up to the maximum of 100. When health is 100, they do not affect the health. The player cannot save health packs. One can use them only one time when picking up.



This picture shows how the health pack should look like. This picture is just a representational.

3.7.2 Weapon Pack

Weapon pack is an equipment box includes various types of weapon. There exist two kinds: shotgun, sniper rifle. One can find detailed information about weapons in weapon section.

These packs provide increase the player's defense against the enemies. The player can save these packs and use them at any time during game. The visual design of the packs will be same with the related weapon.

3.7.3 Ammo Pack

Ammo packs are equipment boxes includes various types of ammos. They will provide additional ammo. For each weapon there will exist several ammo packs that can be collectible. These packs will contain a predefined number of ammo. Ammo packs can increase ammo number of each weapon up to the maximum of 60. The player can save these packs and use them at any time.

Here is the screenshot from Half Life game showing ammo packs. Our design will be similar.



3.8 System Requirements

- Windows XP Service Pack 3 or later
- DirectX 9.0c
- Shader Model 2.0 or later
- 1 GB RAM or higher

4 PLAYER INTERFACE

The user interface will have logically two sections: the main menu and the in-game areas. The user interface will be simple, with a focus on making its use a quick and easy process.

4.1 Main Menu

The screen will have a well-designed background image that will affect the user and will have three buttons on it.

1. New Game

- Easy
 - Typing user name and start the game
- Normal
 - Typing user name and start the game
- Hard
 - Typing user name and Start the game
- Back To Main Menu

2. Options

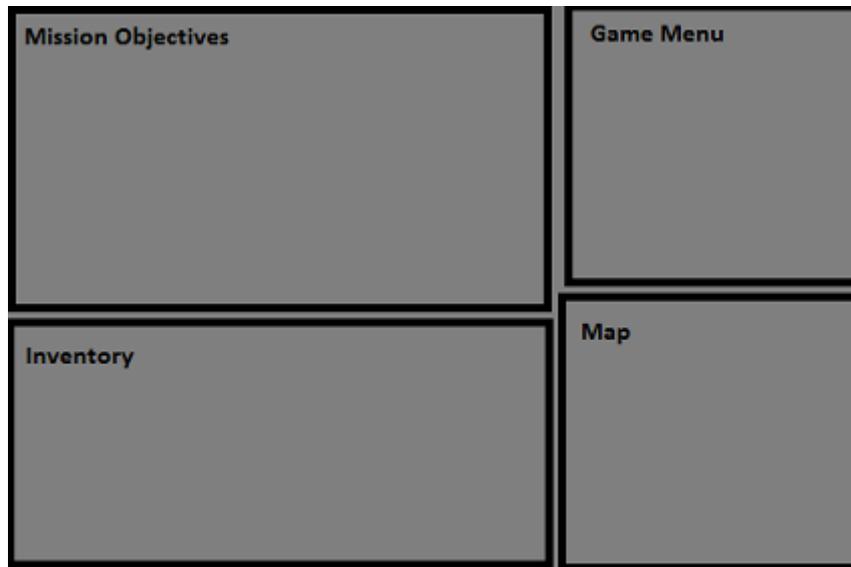
- Controls: Changing Mouse and Keyboard Setting or not
- Audio: Changing Volume settings or not
- Video: Changing Graphic settings or not
- Back to Main Menu

3. Exit

- Yes
- No

4.2 In-Game Menus

The sample drawing below illustrates the proposed in-game menu.



Once the player pauses the game-presses “Esc” button, one will see a menu screen with the following options:

1. Inventory
 - a. Collected item list
2. Map
 - a. Terrain View
3. Game Menu
 - a. Resume Game
 - b. Restart Game
 - c. Options
 - d. Exit Game
4. Mission Objectives
 - a. Mission Description

4.3 Head-up Display (HUD)

The HUD is made up of several different parts. We will place the HUDs with a focus on not making gameplay difficult. The drawing below illustrates the proposed localizations of the HUDs.



Here is the list of HUDs:

1. Health Bar
2. Mini Map
 - a. Terrain
 - b. Visible Units
3. Hostile Counter
 - a. Showing the number of remaining enemies
4. Ammo Counter
 - a. Showing the number of remaining ammos

5 CHARACTERS

5.1 The Player: Dr. Hero KAHEZ

The player assumes the role of Dr. Hero, who is the protagonist of the game. He is a 36-year-old brown-haired, green-eyed man who was a successful virologist worked for U.S Army. Besides, he was one of the virologists who created virus was designed to cure cancer.

He is well educated and well rounded; he likes art, classical music and reading. He had a happy life with his wife and his 5-year-old daughter in New York City. Because of the disease, he has lost his family and his friends one by one. Accordingly, he felt guilty for being a cause of the global disaster and devoted the rest of his life to destroy new “Humankillers” community and save the healthy survivors.

Throughout the rest of his life, he spends his days checking the radio frequencies for survivors, making maps for locations ha had swept and driving around the cities searching for “Humankillers”.

5.1.1 Player Actions

The abilities of our character as a player are:

- Walking and running in all four directions,
- Crouching,
- Jumping,
- Punching,
- Using-Reloading-changing weapons,
- Collecting the pick-ups.

5.1.2 Player Weapons

The player starts the game with two guns and can collect two more weapons during the game. This section provides information on the player’s weapons. We accept Punching as a weapon and the player can use this action by using change weapon control and fire control.

Hand Gun 9mm: SMIT-Wesson 6906



Features:

Damage: 15
Reload Quantity: 17
Range: 1650
Accuracy: 5
Place of origin: ABD

Assault Rifle: G3



Features:

Damage: 35
Reload Quantity: 20
Range: 16500
Accuracy: 10
Place of origin: Germany

Cruiser Shotgun: Mossberg 500



Features:

Damage: 10
Reload Quantity: 6
Range: 2000
Accuracy: 15
Place of origin: ABD

Sniper Rifle: H&K MSG90



Features:

Damage: 50
Reload Quantity: 10
Range: 33000
Accuracy: 0
Zoom Accuracy: 25
Place of origin: Germany

Punch:



Damage:10

5.2 Enemy: Humankillers



Humankillers are the characters, which give the name of the game. They are the degenerated survivors due to virus. Their bodies are no longer functioning. They are essentially rotting. That is why they have pale skin. Since their body is dead, it will not heal and so neither the wound will. The damage will still be visible on their bodies. They like hiding in buildings and dark places during the day. According to some rumors, the reason is that very bright lights have a blinding effect on them. In addition, they are fast, agile, aggressive and lack of communication skills. Unfortunately, they have the ability to smell the livings. They are only interested in one thing: eating the immune humans. They will immediately try to eat one the moment they saw him/her.

“Humakillers” do not have ability to use any weapons; they can only bite the player.

5.2.1 Enemy AI

This section describes how the artificial intelligence of the enemy works and how the player interacts with the AI. The following diagram illustrates the states of enemy AI.

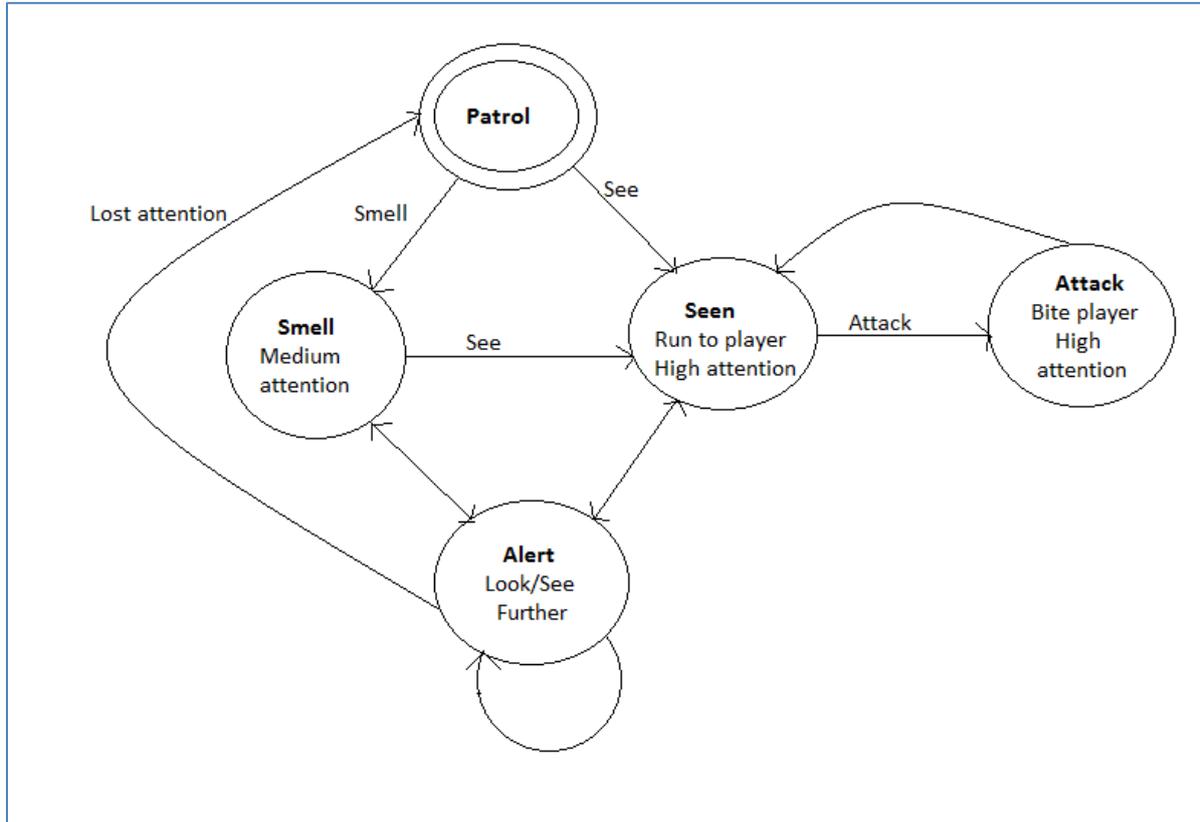


Figure: States of Enemy AI

At patrol state, each enemy AI can follow a unique path where they turn around. The smell and seen states represent that the AI detects the player. At first one, AI checks if the player is within a radius. AI pays attention to the player when the player is within a radius. AIs within a smelling range can turn toward the player and try to see. If the AI is at an alert state, this radius is larger. At second one, AI checks the three conditions.

- The player is within visual distance
- The player is within the AI's Field of View
- No objects are directly between the player and the AI

The AI scans for the player in the range where all these conditions are provided. AIs within a seen range can run to player. When the AI is at alert, enemy can see further. While running, AI checks the distance between itself and the player. When the distance reduces to or under the certain value, AI can attack the player. AIs within an attack range can return to the seen state if the distance increases to or over the certain value.

6 ART and SOUND

6.1 Art Requirements

Format: Bitmap/*dds/tga*->.x , Bitmap/*dds/tga*->.fbx

Style: Realistic future

6.1.1 Art List

- I. Terrain
 - A. Ground/Floor
 - B. Buildings
 - C. Bridges
 - D. Train Tracks
 - E. Fences
 - F. Worn Fences
 - G. Steps
 - H. Step Walls
 - I. Step Pillars
 - J. Concrete Walls
 - K. Monument
 - L. Rubbishes
 - M. Waste-bins

- II. Main Character
 - A. Generic Human Model
 - B. Character Animations
 1. Generic Human Model animations
 - a. Move
 - b. Jump
 - c. Crouch
 - d. Firing
 - e. Reload
 - f. Death
 - g. Punch

- III. Human Killers
 - A. Degenerated Human Model
 - B. Human Killers Animations
 1. Idle
 2. Move
 3. Attack
 4. Death

IV. Inventory Items (2d art)

A. Weapons

1. Handgun
2. Shotgun
3. Assault Rifle
4. Sniper Rifle

B. Equipment

1. Health Pack
2. Weapon Pack
3. Ammo Pack

V. GUI

A. Cursor

1. Menu cursor (arrow)
2. Action cursor (crosshair)
 - a. On
 - b. Off

B. Heads Up Display

1. Health Bar
2. Mini Map
3. Hostile Counter
4. Ammo Counter

C. Menu Window

1. Background image
2. Button image
3. New game button
4. Options
 - a. Controls
 - b. Audio
 - c. Video
5. Credits
6. Exit button

D. In Game Menu Window

1. Background image
2. Inventory
3. Map
4. Game Menu
 - a. Resume Game
 - b. Restart Game
 - c. Options
 - a. Controls
 - b. Audio
 - c. Video
 - d. Exit Game

6.2 Sound Requirements

Format: Wave -> .xap

Style: Realistic Future

6.2.1 Sound List

I. Main Character

- A. Move
- B. Pain
- C. Death
- D. Punch
- E. Firing
 - 1. Handgun
 - 2. Shotgun
 - 3. Assault Rifle
 - 4. Sniper Rifle
- F. Reload
 - 1. Handgun
 - 2. Shotgun
 - 3. Assault Rifle
 - 4. Sniper Rifle
- G. Medicate
- H. Pick-up

II. Human Killer

- A. Attacking
- B. Pain
- C. Death

III. Menu sounds

- A. Main Menu Sound (looped)
- B. In Game Menu Sound (looped)

7 DEVELOPMENT

7.1 Major Development Task

- **Game engine** will bring all of the following engines together to create the game. It will also handle AI and physics routines.
- **Graphics engine** will be responsible for rendering text, 2D images, and 3D models on screen.
 - Drawing models
 - Drawing sprites
 - Drawing text
 - Texturing models
 - Animation
- **Sound engine** will be responsible for playing music and sound effects.
 - Multithreading
 - Playing sounds
- **Input engine** will be responsible for transferring mouse and keyboard input upon request to the game engine.
 - Retrieving Input
- **AI** will be responsible for behaviors of non-playable characters.
 - Path finding
 - Decision-making
- **Menu Engine** will handle all menus in game.

7.2 Development Tools

- Microsoft Visual C# 2008 Express Edition
- XNA Game Studio 3.1
- Blender
- Softimage Mod Tool 7.5

Appendix D

Test Plan

Design History

Version 2	○ Test Cases section added
Version 1	○ Initial version of Test Plan

1 INTRODUCTION

1.1 Purpose

This document is a test plan for the sci-fi based first person shooter game “Humankillers”. The purpose of this document is to describe testing strategy to validate the quality of the product. This document is a guide to plan and control the test processes of the game. The document contains a comprehensive list of tests that will provide to complete the product successfully without any bugs.

The objectives of this test plan for” Humankillers” game are to identify the requirements to be tested, outline the testing approach to be used, identify the test cases and identify the expected results for each test.

1.2 Overview

The purpose of the project is to design and implement a 3-dimensional game written in C# using XNA game development studio. “Humankillers” is an action based 3D FPS game where the player takes role of the main character to destroy enemies. The game is inspired from a science fiction novel “I am Legend” by Richard Matheson, which tells the story of the last man in the world where entire humanity has returned into mindless “Humankillers”. The project includes a complete level of game with documentation. The level will include everything that should be available in a FPS game. In addition, the game will be a single-player.

1.3 Glossary

Term	Definition
Humankillers	Name of the game
FPS	First person shooter
GDD	Game Design Document
TP	Test protocols

1.4 Deliverables

- Test plan document.
- Test protocols.

1.5 Referenced Documents

- GDD
- Gantt Chart document
- TP

1.6 Resource Requirements

- **Hardware**
 - DirectX 9.0c or later
 - Shader Model 2.0 or later
 - 1 GB RAM or higher
- **Software**
 - XNA Game Studio 3.1
 - Microsoft Visual C# 2008 Express Edition
 - Modeling Tools: Blender, Softimage Mod Tool 7.5
 - Test Tools:
 - We will use the Microsoft XNA Test Tools to help test the game.
- **Human**
- **Testers**
 - Hatice Ezgi TUĞLU
 - Kahraman AKYIL
- **Tester for usability testing(out of group)**

2 TEST APPROACH

The test approach consists of a series of different tests. The primary goal of these tests is to ensure that Humankillers is an error free game. Throughout the development, testers will be brought in to test the mechanics and the levels of the game to ensure there are no errors in the game. Furthermore, testers will comment on the fun and experience they have with “Humankillers” and will give the team feedback on how to improve the game.

2.1 Test Levels

The test approach is divided into three main phases: Module testing, integration testing and system testing. In addition, the system testing includes two sub-phases: functional and usability testing. These planned tests are explained briefly below.

- **Module testing** will perform during coding by using debug messages to check that the written code produces wanted results. An important requirement is that the code will compile with zero bugs.
- **Integration testing** will perform after finish module testing in order to validate if each module can work fine with each other. Integration Test proves that system works as integrated unit when all the fixes are complete.
- **System testing** includes two phases: functional testing and usability testing. These will perform after the product reaches its final version. During functional test phase, the tester will test if the product meets the game requirements. The tester tests the requirements using the use cases listed below in Test Cases section. The usability test will perform to understand how easy it is to learn to play the game. Any person out of the team members will perform this test by playing the game.

2.2 Test Schedule

The table below provides information on the start and finish dates of test levels. Please refer to Gantt chart document (GanttChart.gif) for detailed information.

	Start Date	Finish Date
Module Testing	March 26,2010	May 24,2010
Integration Testing	May 25,2010	May 28,2010
System Testing	May 29,2010	June 1,2010

3 RISKS

- The test plan and test schedule are depending on the current GDD. Any changes to the design document will affect the test plan and schedule.
- The group members are unfamiliar with test tool so the test result will be inaccurate.
- Developing software will not finish on time so the schedule for testing process will be changed.

4 TEST CASES

In this section, the use cases to be tested during functional testing are listed and testing steps and expected results are explained in detail.

1 Main Menu System:

1.1 Main Menu

UseCaseID: I
Precondition: none
Flow: 1-User wants to play game 2-User runs game 3-Main Menu containing New Game,Options,Quit pops up.
Postcondition: A main menu is displayed.

1.1.1 New Game Menu

UseCaseID: I.A
Precondition: Main Menu
Flow: 1-User wants to play game 2-User runs game 3-Main Menu pops up 4-User chooses "New Game" menu item 5-Submenu containing Easy,Hard,Normal,Back to Main Menu pops up.
Postcondition: "New Menu" submenu is displayed.

1.1.1.1 Easy

UseCaseID: I.A.1
Precondition: New Game Menu
Flow: 1-User wants to play game 2-User runs game 3-Main Menu pops up 4-User chooses "New Game" menu item 5-Submenu pops up. 6-User chooses "Easy" menu item 7-The game with easy mode starts.
Postcondition: User enters a game with easy mode.

1.1.1.2 Normal

UseCaseID: I.A.2
Precondition: New Game Menu
Flow: 1-User wants to play game 2-User runs game 3-Main Menu pops up 4-User chooses “New Game” menu item 5-Submenu pops up. 6-User chooses “Normal” menu item 7-The game with normal mode starts.
Postcondition: User enters a game with normal mode.

1.1.1.3 Hard

UseCaseID: I.A.3
Precondition: New Game Menu
Flow: 1-User wants to play game 2-User runs game 3-Main Menu pops up 4-User chooses “New Game” menu item 5-Submenu pops up. 6-User chooses “Hard” menu item 7-The game with hard mode starts.
Postcondition: User enters a game with hard mode.

1.1.1.4 Back to Main Menu

UseCaseID: I.A.4
Precondition: New Game Menu
Flow: 1-User wants to play game 2-User runs game 3-Main Menu pops up 4-User chooses “New Game” menu item 5-Submenu pops up. 6-User selects “Back to Main Menu” menu item 7-Main menu is displayed.
Postcondition: Main menu is loaded.

1.1.2 Options Menu

UseCaseID: I.B
Precondition: Main Menu
Flow: 1-User wants to play game 2-User runs game 3-Main Menu pops up 4-User selects “Options” menu item 5-Submenu containing Controls,Audio,Video,Back to Main Menu pops up.
Postcondition: “Options” submenu is displayed.

1.1.2.1 Controls

UseCaseID: I.B.1
Precondition: Options Menu
Flow: 1-User wants to play game 2-User runs game 3-Main Menu pops up 4-User chooses “New Game” menu item 5-Submenu pops up. 6-User selects “Controls” 7-Submenu pops up 8-User changes control settings 9-User returns to “Options” menu
Postcondition: User changes control settings.

1.1.2.2 Audio

UseCaseID: I.B.2
Precondition: Options Menu
Flow: 1-User wants to play game 2-User runs game 3-Main Menu pops up 4-User chooses “New Game” menu item 5-Submenu pops up. 6-User selects “Audio” 7-Submenu pops up 8-User changes audio settings 9-User returns to “Options” menu
Postcondition: User changes audio settings.

1.1.2.3 Video

UseCaseID: I.B.3
Precondition: Options Menu
Flow: 1-User wants to play game 2-User runs game 3-Main Menu pops up 4-User chooses “New Game” menu item 5-Submenu pops up. 6-User selects “Video” 7-Submenu pops up 8-User changes video settings 9-User returns to “Options” menu
Postcondition: User changes video settings.

1.1.2.4 Back to Main Menu

UseCaseID: I.B.4
Precondition: Options Menu
Flow: 1-User wants to play game 2-User runs game 3-Main Menu pops up 4-User chooses “New Game” menu item 5-Submenu pops up. 6-User chooses “Back to Main Menu” menu item 7-Main menu appears.
Postcondition: Main menu is displayed.

1.1.3 Exit

UseCaseID: I.C
Precondition: Main Menu
Flow: 1-User wants to play game 2-User runs game 3-Main Menu pops up 4-User decides to quit game 5-User selects “Exit” 6-Game exits.
Postcondition: User exits game.

2 The Game:

UseCaseID: II
Precondition: Main Menu functions correctly
Flow: 1- User runs game 2- Main Menu pops up 3-User enters a game 4-Play begins 5a-When player kills all enemies or is killed by enemies,the play ends 5b-User exits game via in-game menu 6-User returns to Main Menu.
Postcondition: Game begins and ends successfully.

2.1 Graphics:

UseCaseID: II.A
Precondition: none
Flow: 1- User runs game 2- Main Menu pops up 3- The main menu graphics are displayed 4- User starts game 5- In-game graphics are displayed.
Postcondition: The graphics requested are displayed.

2.1.1 Enemies

UseCaseID: II.A.1
Precondition: The game is running
Flow: 1- User encounters enemies 2- The game requests the graphic model for enemies 3- Enemy model is displayed.
Postcondition: Enemies are displayed.

2.1.2 Terrain

UseCaseID: II.A.2
Precondition: The game is running
Flow: 1- The game determines user direction 2- The game displays the appropriate graphic models.
Postcondition: A 3D terrain is displayed.

2.1.3 Fire Effect

UseCaseID: II.A.3
Precondition: The game is running
Flow: 1- User fires the weapon 2- The fire particle effect is requested from graphics engine 3- The game displays the effect.
Postcondition: A fire particle effect is displayed.

2.1.4 Bullet-Hit Effect

UseCaseID: II.A.4
Precondition: The game is running
Flow: 1- User fires the weapon 2- The bullet hits the object 3- The bullet-hit effect is requested from the graphics engine 4- The game displays the effect where the bullet hits.
Postcondition: A bullet-hit effect is displayed.

2.1.5 Weapons

UseCaseID: II.A.5
Precondition: The game is running
Flow: 5- User presses key to change weapon 6- The game needs to display the new weapon 7- The weapon model is requested from the graphics engine 8- The game displays the new weapon.
Postcondition: A weapon is displayed.

2.2 Sound:

2.2.1 Music

UseCaseID: II.B.1
Precondition: Main menu or In-game menu is loaded
Flow: 1- User runs the game 2- Main menu pops up 3- The soundtrack of music is played until main menu disappears.
Postcondition: The soundtrack of music is played during the menus are displayed.

2.2.2 Sound Effects

UseCaseID: II.B.2
Precondition: The game is running
Flow: 1- Game needs to play sound effects 2- The game program accesses sound engine 3- Game plays appropriate sound effect for corresponding event.
Postcondition: Appropriate sound effects will be played for corresponding events.

2.3 Physics:

2.3.1 Aiming

UseCaseID: II.C.1
Precondition: The game is running
Flow: 1- User moves the mouse 2- Objects move according to physics engine 3- The view of player moves accordingly.
Postcondition: Players' view has been changed.

2.3.2 Collisions:

2.3.2.1 Player to Enemy

UseCaseID: II.C.2.a
Precondition: Player and enemy models are displayed.
Flow: 1- Player collides with Enemy 2- If the player is with Punch weapon, damage is assigned to enemy, 3- Otherwise,damage is assigned to player
Postcondition: Player or Enemy takes damage.

2.3.2.2 Enemy to Enemy

UseCaseID: II.C.2.b
Precondition: Enemy models are displayed.
Flow: 1- Enemy shares pace with another Enemy 2- Game detects and accesses physics engine 3- The physics engine determines the behavior of enemies.
Postcondition: Enemies interact with each other.

2.3.2.3 Enemy to Bullet

UseCaseID: II.C.2.c
Precondition: Enemy and bullet graphics are displayed.
Flow: 1- A bullet collides with enemy 2- The bullet is detonated 3- Damage is assigned to enemy.
Postcondition: Enemy takes damage.

2.3.2.4 *Player to Terrain*

UseCaseID: II.C.2.d
Precondition: Player and terrain graphics are displayed.
Flow: 1- Enemy collides with walls,ground or any concrete objects 2- The player is blocked 3- The player velocity decreases accordingly.
Postcondition: The velocity of player decreases.

2.3.2.5 *Enemy to Terrain*

UseCaseID: II.C.2.e
Precondition: Enemy and terrain graphics are displayed.
Flow: 4- Enemy collides with walls,ground or any concrete objects 5- The enemy is blocked 6- The enemy velocity decreases accordingly.
Postcondition: The velocity of enemy decreases.

2.4 Gameplay:

2.4.1 Combat:

2.4.1.1 Fire

UseCaseID: II.D.1.a
Precondition: The player has a weapon,ammo.
Flow: 1- User presses the fire button 2- The current weapon fires 3- The ammo counter decreases accordingly.
Postcondition: The player has fired weapon.

2.4.1.2 Take Damage

UseCaseID: II.D.1.b
Precondition: The player must be alive.
Flow: 1- The player is bitten by enemies 2- The health decreases accordingly.
Postcondition: The player's health is changed.

2.4.1.3 Die

UseCaseID: II.D.1.c
Precondition: The player health is over zero
Flow: 1- The player is bitten by enemies 2- The health decreases accordingly 3- The health reaches or gets below zero 4- The player dies 5- The game ends accordingly.
Postcondition: The player dies and the game ends successfully.

2.4.1.4 Change Weapon

UseCaseID: II.D.1.d
Precondition: The player has other weapons.
Flow: 1- User presses a key to change weapon 2- Current weapon changes.
Postcondition: The player changes current weapon.

2.4.1.5 Reload Weapon

UseCaseID: II.D.1.e
Precondition: The ammo number is below maximum for current weapon.
Flow: 1- User presses a key to reload weapon 2- The ammo counter for current weapon increases accordingly.
Postcondition: The ammo counter is changed.

2.4.2 Move:

2.4.2.1 Jump

UseCaseID: II.D.2.a
Precondition: Player is standing on ground.
Flow: 1- User presses a key to jump. 2- User moves in “up” direction.
Postcondition: Player position changes.

2.4.2.2 Run

UseCaseID: II.D.2.b
Precondition: Player is able to move in direction.
Flow: 1- User presses a key to run 2- Player gains movement in direction 3- Player velocity increases.
Postcondition: Player position and velocity changes.

2.4.2.3 Walk

UseCaseID: II.D.2.c
Precondition: Player is able to move in direction.
Flow: 1- User presses a key to walk 2- Player gains movement in direction.
Postcondition: Player position changes.

2.4.2.4 Crouch

UseCaseID: II.D.2.d
Precondition: Player is standing on ground.
Flow: 1- User presses a key to crouch 2- User moves in “down” direction.
Postcondition: Player position changes.

2.4.3 Pickups:

2.4.3.1 Weapon(Pickups)

UseCaseID: II.D.3.a
Precondition: User moves onto a space that contains a takable weapon.
Flow: 1- User moves to a space with a takeable weapon 2- User picks up the weapon 3- Weapon disappears from world.
Postcondition: User picks up the new weapon.

2.4.3.2 Ammo(Pickups)

UseCaseID: II.D.3.b
Precondition: User moves onto a space that contains a takable ammo pack, The number of player ammo pack is below 5.
Flow: 1- User moves to a space with a takeable ammo pack 2- User picks up the ammo pack 3- Ammo pack disappears from world.
Postcondition: User picks up the additional ammo pack.

2.4.3.3 HealthPack(Pickups)

UseCaseID: II.D.3.c
Precondition: User moves onto a space that contains a takable health pack, The health of player is not full.
Flow: 1- User moves to a space with a takeable health pack 2- User picks up the health pack 3- The player health changes accordingly 4- Health pack disappears from world.
Postcondition: User picks up the health pack.

2.4.4 HUD:

2.4.4.1 HealthHUD

UseCaseID: II.D.4.a
Precondition: Gameplay begins.
Flow: 1- User enters a game 2- Play begins 3- HUD is displayed. 4- The health of player is shown on the HUD. 5- The health bar is updated whenever health of player changes.
Postcondition: The health of player is shown on the HUD.

2.4.4.2 AmmoCounterHUD

UseCaseID: II.D.4.b
Precondition: Gameplay begins, The player has weapon.
Flow: 1- User enters a game 2- Play begins 3- HUD is displayed. 4- The amount of ammo for current weapon is shown on the HUD. 5- The counter is updated whenever the amount changes.
Postcondition: The ammo counter is shown on the HUD.

2.4.4.3 HostileCounterHUD

UseCaseID: II.D.4.c
Precondition: Gameplay begins.
Flow: 1- User enters a game 2- Play begins 3- HUD is displayed. 4- The amount of remaining hostile is shown on the HUD. 5- The hostile counter is updated whenever the player kills a hostile.
Postcondition: The number of hostile is shown on the HUD.

2.4.4.4 MinimapHUD

UseCaseID: II.D.4.d
Precondition: Gameplay begins.
Flow: 1- User enters a game 2- Play begins 3- HUD is displayed. 4- The mini map is shown on the HUD. 5- The mini map is updated depending on the location of player and objects.
Postcondition: The minimap is shown on the HUD.

2.4.5 In-game Menu

UseCaseID: II.D.5
Precondition: User is in game.
Flow: 1- User presses “Esc” button 2- In-game menu containing Resume Game, Restart Game, Options, Exit Game appears.
Postcondition: In-game Menu appears.

2.4.5.1 Resume Game

UseCaseID: II.D.5.a
Precondition: In-game menu is loaded.
Flow: 1- User presses “Esc” button 2- In-game menu appears 3- User selects “Resume Game” 4- User returns to game play.
Postcondition: User returns to game.

2.4.5.2 Restart Game

UseCaseID: II.D.5.b
Precondition: In-game menu is loaded.
Flow: 1- User presses “Esc” button 2- In-game appears 3- User selects “Restart Game” 4- User restarts the game.
Postcondition: User restarts the game.

2.4.5.3 Exit Game

UseCaseID: II.D.5.c
Precondition: In-game menu is loaded.
Flow: 1- User presses “Esc” button 2- In-game appears 3- User selects “Exit Game” 4- User leaves game play 5- User returns to main menu.
Postcondition: User exits game.

2.5 Enemy:

2.5.1 Enemy Combat:

2.5.1.1 Die

UseCaseID: II.E.1.a
Precondition: Enemy is alive.
Flow: 1- The player shoots the enemy 2- The health of enemy decreases accordingly 3- The health reaches or gets below zero 4- The enemy dies.
Postcondition: The enemy dies.

2.5.1.2 TakeDamage

UseCaseID: II.E.1.b
Precondition: The health of enemy is over zero.
Flow: 1- The player shoots the enemy 2- The health of enemy decreases accordingly.
Postcondition: The enemy health is changed.

2.5.2 Enemy AI:

2.5.2.1 Walk(Pathfinding)

UseCaseID: II.E.2.a
Precondition: The player is out of the radius.
Flow: 1- The game starts 2- Enemies follow a unique path where they turn around.
Postcondition: Enemy walks in a unique path where they turn around.

2.5.2.2 Run to Player

UseCaseID: II.E.2.b
Precondition: The player is within a seen range.
Flow: 1- The player walks through the enemy 2- The player is within a seen range 3- The enemy starts to run to player.
Postcondition: The enemy runs to player.

2.5.2.3 *Bite*

UseCaseID: II.E.2.c
Precondition: The player is within an attack range.
Flow: 1- The enemy runs to player 2- The player is within an attack range 3- The enemy attacks and bites the player.
Postcondition: The enemy attacks to player.

Appendix E

Test Protocols

Design History

Version 4	○ Test Status and Comments updated
Version 3	○ Test Status and Comments updated
Version 2	○ Test Status and Comments updated
Version 1	○ Initial version of Test Protocols

1 Main Menu System:

Test Case ID	Test Case	Actions	Excepted Result	Test Status	Comment
I	Main Menu	1-User wants to play game 2-User runs game	A main menu is displayed.	Pass	
I.A	New Game Menu	1-User runs game 2-Main Menu pops up 3-User chooses "New Game" menu item	"New Menu" submenu is displayed.	Pass	
I.A.1	Easy	1-User runs game 2-Main Menu pops up 3-User chooses "New Game" menu item 4-Submenu pops up. 5-User chooses "Easy" menu item	User enters a game with easy mode.	Pass	
I.A.2	Normal	1-User runs game 2-Main Menu pops up 3-User chooses "New Game" menu item 4-Submenu pops up. 5-User chooses "Normal" menu item	User enters a game with normal mode.	Pass	
80 Page					

I.A.3	Hard	1-User runs game 2-Main Menu pops up 3-User chooses "New Game" menu item 4-Submenu pops up. 5-User chooses "Hard" menu item	User enters a game with hard mode.	Pass	
I.A.4	Back to Main Menu	1-User runs game 2-Main Menu pops up 3-User chooses "New Game" menu item 4-Submenu pops up. 5-User selects "Back to Main Menu" menu item	Main menu is loaded.	Pass	
I.B	Options Menu	2-User runs game 3-Main Menu pops up 4-User selects "Options" menu item	"Options" submenu is displayed.	Pass	
I.B.1	Controls	1-User runs game 2-Main Menu pops up 3-User chooses "New Game" menu item 4-Submenu pops up. 5-User selects "Controls"	Control settings are changed.	Pass	Control options sub-menu is displayed. However, settings cannot be changed.

		6-Submenu pops up 7-User changes control settings 8-User returns to "Options" menu			
I.B.2	Audio	1-User runs game 2-Main Menu pops up 3-User chooses "New Game" menu item 4-Submenu pops up. 5-User selects "Audio" 6-Submenu pops up 7-User changes audio settings 8-User returns to "Options" menu	Audio settings are changed.	Pass	Audio options sub-menu is displayed. However, settings cannot be changed.
I.B.3	Video	1-User runs game 2-Main Menu pops up 3-User chooses "New Game" menu item 4-Submenu pops up. 5-User selects "Video" 6-Submenu pops up 7-User changes video settings 8-User returns to	Video settings are changed.	Pass	Video options sub-menu is displayed. However, settings cannot be changed.

		“Options” menu			
I.B.4	Back to Main Menu	1-User runs game 2-Main Menu pops up 3-User chooses “New Game” menu item 4-Submenu pops up. 5-User chooses “Back to Main Menu” menu item	Main menu is displayed.	Pass	
I.C	Exit	1-User runs game 2-Main Menu pops up 3-User decides to quit game 4-User selects “Exit”	Game exits.	Pass	

2 The Game:

Test Case ID	Test Case	Actions	Excepted Result	Test Status	Comment
II-1	BeginGame	1- User runs game 2- Main Menu pops up 3-User enters a game	Game begins	Pass	
II-2	EndGame	1-When player kills all enemies or is killed by enemies 2-User exits game via in-game menu 3-User returns to Main Menu.	Game ends	Pass	
II.A	Graphics	1-User runs game 2-Main Menu pops up 3-The main menu graphics are displayed 4-User starts game	The graphics requested are displayed.	Pass	
II.A.1	Enemies	1- User encounters enemies 2- The game requests the graphic model for enemies	Enemies are displayed.	Pass	
II.A.2	Terrain	1-The game determines user direction 2-The game displays the appropriate graphic models.	A 3D terrain is displayed.	Pass	
II.A.3	Fire Effect	1- User fires the weapon 2- The fire particle effect is	A fire particle effect is displayed.	Pass	

		requested from graphics engine			
II.A.4	Bullet-Hit Effect	<ul style="list-style-type: none"> 1- User fires the weapon 2- The bullet hits the object 3- The bullet-hit effect is requested from the graphics engine 	The game displays the bullet-hit effect where the bullet hits.	Pass	
II.A.5	Weapons	<ul style="list-style-type: none"> 1- User presses key to change weapon 2- The game needs to display the new weapon 3- The weapon model is requested from the graphics engine 4- The game displays the new weapon. 	A weapon is displayed.	Pass	

2.1 Sound:

Test Case ID	Test Case	Actions	Expected Result	Test Status	Comment
II.B.1	Music	1- User runs the game 2- Main menu pops up	The soundtrack of music is played until main menu disappears.	Pass	
II.B.2	Sound Effects	1- Game needs to play sound effects 2- The game program accesses sound engine 3- Game plays appropriate sound effect for corresponding event.	Appropriate sound effects will be played for corresponding events.	Pass	Sound effects: weapon fire, pickups effect work fine. -- weapon reload, weapon change, enemy damage, player damage are ready to use but not implemented yet.

2.2 Physics:

Test Case ID	Test Case	Actions	Excepted Result	Test Status	Comment
II.C.1	Aiming	1- User moves the mouse 2- Objects move according to physics engine 3- The view of player moves accordingly.	Players' view has been changed.	Pass	
II.C.2.a	Player to Enemy	1- Player collides with Enemy 2- If the player is with Punch weapon, damage is assigned to enemy, 3- Otherwise,damage is assigned to player	Player or enemy takes damage.	Pass	
II.C.2.b	Enemy to Enemy	1- Enemy shares pace with another Enemy 2- Game detects and accesses physics engine 3- The physics engine determines the behavior of enemies.	Enemies interact with each other.	Pass	
II.C.2.c	Enemy to Bullet	1- A bullet collides with enemy 2- The bullet is detonated	Enemy takes damage.	Pass	
II.C.2.d	Player to Terrain	1- Enemy collides with walls, ground or any concrete objects 2- The player is blocked 3- The player velocity decreases accordingly.	The velocity of player decreases.	Pass	
II.C.2.e	Enemy to Terrain	1- Enemy collides with walls, ground or any concrete objects 2- The enemy is blocked	The velocity of enemy decreases.	Pass	

2.3 Gameplay:

2.3.1 Combat:

Test Case ID	Test Case	Actions	Excepted Result	Test Status	Comment
II.D.1.a	Fire	User presses the fire button	-The player has fired weapon - The ammo counter decreases accordingly.	Pass	
II.D.1.b	Take Damage	The player is bitten by enemies	The player health decreases	Pass	
II.D.1.c	Die	1- The player is bitten by enemies 2- The health decreases accordingly 3- The health reaches or gets below zero	The player dies and the game ends successfully.	Pass	
II.D.1.d	Change Weapon	User presses a key to change weapon	The player changes current weapon.	Pass	
II.D.1.e	Reload Weapon	User presses a key to reload weapon	The ammo counter for current weapon increases	Pass	

2.3.2 Move:

Test Case ID	Test Case	Actions	Excepted Result	Test Status	Comment
II.D.2.a	Jump	User presses a key to jump.	User moves in “up” direction.	Pass	
II.D.2.b	Run	User presses a key to run	-Player gains movement in direction -Player velocity increases.	Pass	
II.D.2.c	Walk	User presses a key to walk	Player gains movement in direction.	Pass	
II.D.2.d	Crouch	User presses a key to crouch	User moves in “down” direction.	Pass	We found it unnecessary and removed from the game.

2.3.3 Pickups:

Test Case ID	Test Case	Actions	Excepted Result	Test Status	Comment
II.D.3.a	Weapon(Pickups)	1- User moves to a space with a takeable weapon 2- User picks up the weapon	- User picks up the new weapon. -Weapon disappears from world.	Pass	
II.D.3.b	Ammo(Pickups)	1- User moves to a space with a takeable ammo pack 2- User picks up the ammo pack	- Ammo pack disappears from world. -User picks up the additional ammo pack.	Pass	
II.D.3.c	HealthPack(Pickups)	1- User moves to a space with a takeable health pack 2- User picks up the health pack	- User picks up the health pack. -The player health changes accordingly -Health pack disappears from world.	Pass	

2.3.4 HUD:

Test Case ID	Test Case	Actions	Excepted Result	Test Status	Comment
II.D.4.a	HealthHUD	1- User enters a game 2- Play begins 3- HUD is displayed. 4- The health of player is shown on the HUD. 5- The health bar is updated whenever health of player changes.	The health of player is shown on the HUD.	Pass	Updating whenever the player takes damage.
II.D.4.b	AmmoCounterHUD	1- User enters a game 2- Play begins 3- HUD is displayed.	The amount of ammo for current weapon is shown on the HUD.	Pass	Updating whenever player fires and reloads the weapon.
II.D.4.c	HostileCounterHUD	1- User enters a game 2- Play begins 3- HUD is displayed.	The amount of remaining hostile is shown on the HUD.	Pass	Decreases when the enemy dies.
II.D.4.d	MiniMapHUD	1- User enters a game 2- Play begins 3- HUD is displayed.	The minimap is shown on the HUD.	Not Pass	Not implemented due to time shortage.

2.3.5 In-game Menu:

Test Case ID	Test Case	Actions	Excepted Result	Test Status	Comment
II.D.5	In-game Menu	User presses “Esc” button	In-game Menu appears.	Pass	Menu items were minimized. New in-game menu includes Resume and Exit game menu items.
II.D.5.a	Resume Game	1-User presses “Esc” button 2-In-game menu appears 3-User selects “Resume Game”	User returns to game play.	Pass	
II.D.5.b	Restart Game	1- User presses “Esc” button 2- In-game appears 3- User selects “Restart Game”	User restarts the game.	Not Pass	The user have to exit current game to restart the game.
II.D.5.c	Exit Game	1- User presses “Esc” button 2- In-game appears 3- User selects “Exit Game”	User exits the game.	Pass	

2.4 Enemy:

2.4.1 Enemy Combat:

Test Case ID	Test Case	Actions	Excepted Result	Test Status	Comment
II.E.1.a	Die	1- The player shoots the enemy 2- The health of enemy decreases accordingly 3- The health reaches or gets below zero	The enemy dies.	Pass	After a few seconds, the body is disappeared when enemy dies.
II.E.1.b	TakeDamage	The player shoots the enemy	The health of enemy decreases accordingly.	Pass	Take damage according to type of weapon.

2.4.2 Enemy AI:

Test Case ID	Test Case	Actions	Excepted Result	Test Status	Comment
II.E.2.a	Walk	The game starts	Enemies follow a unique path where they turn around.	Pass	
II.E.2.b	Run towards Player	1- The player walks through the enemy 2- The player is within a seen range	The enemy starts to run towards player.	Pass	
II.E.2.c	Bite	1- The enemy runs to player 2- The player is within an attack range	-The enemy attacks and bites the player. -Player takes damage.	Pass	

Appendix F

Gantt Chart

