# Requirements and Analysis Document for Need for Ghetto

## Contents



**Version**: 4.1

**Date**: 31-05-2015


**Author**: Amar, Jonathan, Marcus and Anton.


This version overrides all previous versions.

# 1 Introduction

This section gives a brief overview of the project.

## *1.1 Purpose of application*

The group behind the application aims to create a modern version of a classic arcade game genre, the “top-down shooter” and “bullet hell” genre. Modern should be interpreted as the theme of the game.

## *1.2 General characteristics of application*

The application is created as an Android Application, which means it will only work for Android devices. The game will not support multiplayer but it will support high score challenges. The graphical user interface will be adapted to work with modern smartphones.

The application is designed for one user who will control a car and try to shoot down enemy cars. The enemy cars are any other vehicle in the game. Some of the enemies will shoot back and if the player get hit he will lose 1 life. When the player has lost 3 lives he's dead and it's Game Over. The user will be taken to a screen to submit his score to the high score list.

## *1.3 Scope of Application*

The application will include simple computer-controller enemies. They will continuously fire bullets. The game should be played alone as it will not support multiplayer at the time of this document. See possible future directions.

## *1.4 Objectives and success criteria of the project*

The success of the project will be reached if a user can play until death (or for at least 1 minute) in an infinitely long level without the game crashing. If the player dies his or her score should be saved to the high score list.

There should be two or more different types of enemies. There should also be at least two different weapons that the player and enemies can use.

## *1.5 Definitions, acronyms and abbreviations*

- GUI, graphical user interface.
- Java, programming language.
- Android, a primarily mobile platform where applications are developed in Java.
- LibGDX, a framework for games. Has built-in functions for hardware accelerated graphics.
- Player, the car that the user controls.
- Enemy, the cars/units the user is supposed to shoot down.

# 2 Requirements

In this section we specify all requirements.

## *2.1 Functional requirements*

Create a list of high level functions here (from the use cases).

- Player should be able to move.
- Player should be able to shoot down enemies.
- Enemies should spawn in a pre-defined pattern and order.
- Player should be able to crash into enemies.
- Enemies should be able to shoot down the Player.

## *2.2 Non-functional requirements*

### 2.2.1 Usability

NA

### 2.2.2 Reliability

NA

### 2.2.3 Performance

It should be able to run on the most common Android devices. The application should not drop to less than 25 fps on a Samsung Galaxy S3, which is one of the groups test devices.

### 2.2.4 Supportability

The application should be able to run on most Android devices running Android version 4.0 and above. That includes the most popular phones that have high-end performance. Older devices are there for left out.

There should be JUnit tests for collisions between player, enemy and bullets. These are important to test for as they are vital for the game to work. There will also be some extra test for other vital parts, like adding bullets to weapons and moving them around.

### 2.2.5 Implementation

The code is written in Java using the Android Studio editor. The application is built upon the framework libGDX. Android Studio gives the supports for the application to be installed on Android devices.

### 2.2.6 Packaging and installation

The application will be compiled as an .apk file which is standard for Android Applications. The application will not be uploaded to Google Play.

### 2.2.7 Legal

All the images in the game is modelled and created by us and their copyright belongs to us. That applies to everything except the background in game and the background in the main menu and high score screen. Those images are used with permission but with the condition of not making money. The music in the application is also copyright protected and belongs to the project group.

## *2.3 Application models*

### 2.3.1 Use case model

The text explaining the use cases and the UML diagram can be found in APPENDIX.

### 2.3.2 Use cases priority

1. Move
2. Shoot
3. Start
4. Quit
5. Pause
6. Resume

### 2.3.3 Domain model

The domain model can be found in the APPENDIX.

### 2.3.4 User interface

The application will have a fixed GUI designed for Android devices with high pixel density. With high pixel density we target about 350 pixels per inch.

## *2.4 References*

NA

# APPENDIX

## *GUI*

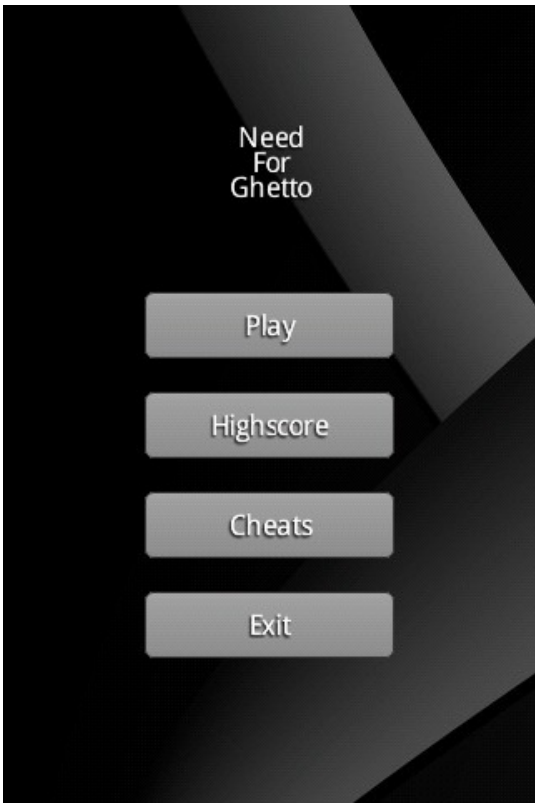



*Illustration 1: The main menu of Need for Ghetto*

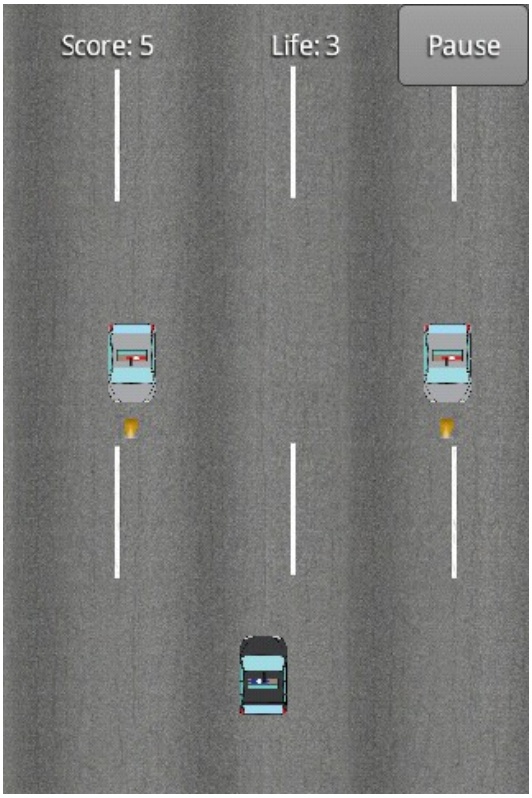
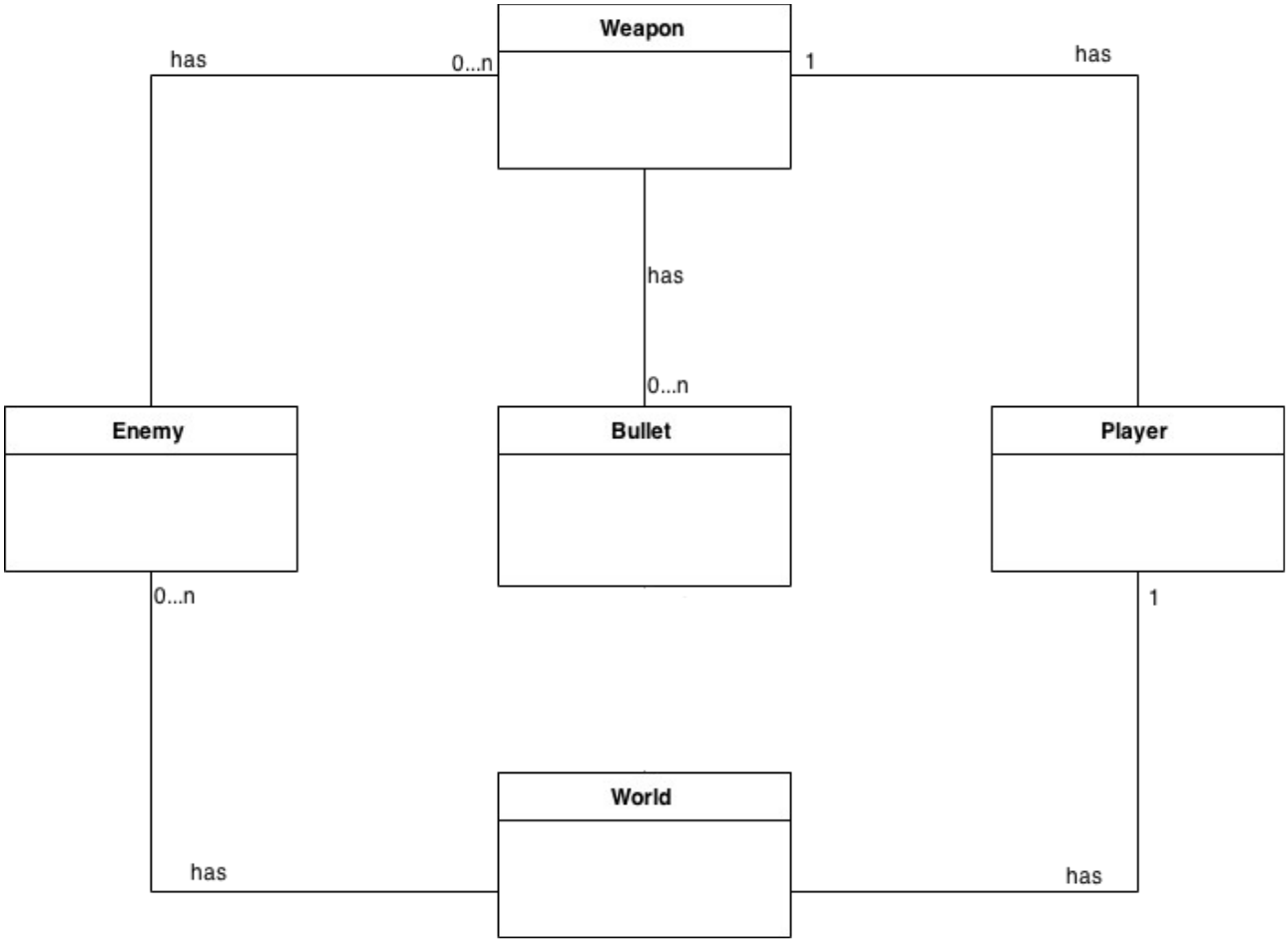



*Illustration 2: The in-game GUI*


*Illustration 3: The high score screen*

## *Domain model*



## *Use cases*

## Move

**Summary:** This is how the user moves his or her player on the screen. Every other UC can preceded this UC except UC quit and pause. Some time before this UC the UC Play will have been completed.

**Priority:** High

**Extends:** -

**Includes:** -

**Participators:** The application user.

## *Normal flow of events*

This is the only flow of events for move.

| | **Actor** | **System** |
|---|---|---|
| 1 | Puts finger on screen | |
| 2 | | Moves player to finger position |
| 3 | Moves finger on screen | |
| 4 | | Moves player to finger position |
| 5 | Removes finger from screen | |
| 6 | | Player stops moving |

## *Alternate flow of events*

**Flow 2.1 Player moves into enemy**

| | **Actor** | **System** |
|---|---|---|
| 2.1.1 | | Moves player to finger position |
| 2.1.2 | | Enemy removed from game<br>Player returns to start position and loses 1 life. Player becomes invulnerable for 4 seconds. |

**Flow 2.1.1 Player don't have any lives left**

| | **Actor** | **System** |
|---|---|---|
| 2.1.2.1 | | Game Over |

**Flow 2.2 Player moves into a bullet belonging to an enemy**

| | Actor | System |
|---|---|---|
| 2.2.1 | | Moves player to finger position |
| 2.2.2 | | Bullet removed from game<br>Player returns to start position and loses 1 life. Player becomes invulnerable for 2 seconds. |

**Flow 2.2.1 Player don't have any lives left**

| | Actor | System |
|---|---|---|
| 2.2.2.1 | | Game Over |

## Shoot

**Summary:** This is how the user fires bullets from the players weapon. This UC happens at the same time as UC Move. With the same preceding UCs.

**Priority:** High

**Extends:** -

**Included:** -

**Participators:** User

### *Normal flow of events*

This is the only flow of events for Shoot.

| | Actor | System |
|---|---|---|
| 1 | Puts finger on screen | |
| 2 | | Player starts to shoot |
| 3 | Removes finger from screen | |
| 4 | | Player stops shooting |

## Start

**Summary:** This happens when the user press the play button to launch the game. This happens before every other UC can happen.

**Priority:** Medium

**Extends:** -

**Includes:** -

**Participators:** User

### *Normal flow of events*

There is only one flow of events for this UC.

| | **Actor** | **System** |
|---|---|---|
| 1 | Pressing Start button | |
| 2 | | New level loads, places player at default position, resets score and starts to spawn enemies |

## Quit

**Summary:** This is the last UC to happen. Turns the application off.

**Priority:** Medium

**Extends:** -

**Includes:** -

**Participators:** User

### *Normal flow of events*

There is only one flow of events for this UC.

| | **Actor** | **System** |
|---|---|---|
| 1 | Pressing Quit button | |
| 2 | | Closing application |

## Pause

**Summary:** This happens when the user press the Pause button when he is in the game.

**Priority:** Low

**Extends:** -

**Includes:** -

**Participators:** The user

### *Normal flow of events*

This is the only flow of event that's possible for this UC.

| | **Actor** | **System** |
|---|---|---|
| 1 | Pressing Pause button | |
| 2 | | Pausing game logic. Player, bullets and enemies stops moving and shooting. |

## Resume

**Summary:** This is what happens after the user press the resume button. The game is already paused before this can happen so UC Pause precedes this.

**Priority:** Low

**Extends:** -

**Includes:** -

**Participators:** The user

### *Normal flow of events*

There is only one flow of events for this UC.

| | **Actor** | **System** |
|---|---|---|
| 1 | Pressing Resume button | |
| 2 | | Resumes the game where it left of. |

## *Use Case Diagram*