

Object Oriented Software Engineering Project

CS - 319

**SPACE IMPACT**

**Analysis Report**

**Büşra Arabacı, Büşra Oğuzoğlu, Deniz Şen, Alperen Kaya, Gerard Hysa**

**Supervisor: Bora Güngören**

Date: 18/11/2017

**Table of Contents**

[**1 Introduction**](#_cpwejeywnsfv) **3**

[**2 Proposed System**](#_da1f9y2sx3e5) **3**

[2.1 Overview](#_ooll5peierdb) 3

[2.2 Functional Requirements](#_3llhrc8o7yue) 4

[2.2.1 Play Game](#_x0msltml9v7) 4

[2.2.2 Store](#_jtlx96mk7coz) 4

[2.2.3 Pause Game](#_hkug4koeyss) 5

[2.2.4 View Information](#_vgiyogn0h0wm) 5

[2.2.4.1 View Credits](#_ko7ewhvrsf2d) 5

[2.2.4.2 View Tutorial](#_z6f3v2iiou9f) 5

[2.2.5 View Achievements](#_on40gobu1ls6) 6

[2.3.1 Performance](#_iz54whxw2x6f) 6

[2.3.2 User-Friendly](#_fml1wthelqwf) 6

[2.3.3 Reusability](#_jl7mlqq2tp38) 6

[2.3.4 Additional requirements ( 2 iteration)](#_lij27t7g6ozr) 6

[2.3.4.1 Usability](#_l0674i9patx3) 6

[2.4 Pseudo Requirements](#_3t78wbeugxwv) 7

[2.5 System Models](#_v7jsckc63pdf) 7

[2.5.1 Use-Case Models](#_mx1fq8qrbi68) 7

[2.5.2 Object and Class Model](#_9d7a941dm1bc) 12

[2.5.3 Dynamic Models](#_hlh9ha6k8jy) 13

[2.5.4 User Interface](#_so28q8bk869p) 16

[**3 Improvement Summary**](#_n6n0c1bijw01) **18**

[**4 References**](#_vxln1h7c2mxt) **18**

# 

# **1** **Introduction**

Space Impact is a game that has been around for many years so we decided to create our version of it where we will put new features into the game. The game will be implemented in Android so everyone can play it in their phones. The main idea of the game is to shoot alien spaceships with your spaceship and clear the way to the next level.

The game will have great graphics and all the levels will be different from each other, visually and functionally. Additionally powerups and game upgrades will be available to buy from the player with the coins that it will get from killing enemies and completing levels. However, the main idea remains the same with the classic game, to shoot alien spaceships. In order to make the game more attractive and explanatory to the player we have added a short story below.

To return again at home in planet Omega75, Jake and Berry have as their hope the only spaceship left after the destruction of the mothership and death of their comrades. Wandering through space they encounter an alien swarm of spaceships that they need to destroy in order to arrive home safely. In different levels alien spaceships will come and the player has to destroy them. Different alien spaceships will launch rockets towards the player and try to neutralize it. The player has only three lives available and at the end of each level the player will face a boss spaceship. According to the time spent in each level and killing of the enemies, the player will collect points that will be added to its score. In the end of the level the killing of the boss gives extra points.

Will you help Jake and Berry arrive home safely?

# 

# **2** **Proposed System**

## **2.1** **Overview**

Space Impact is an intensive game that keeps the player engaged and eager to play more. The intensive gameplay and the simplicity in playing makes our game immersive for the player.

Basically the player controls its spaceship with its finger and moves it around to avoid other spaceships or dodge enemy bullets. The player hits a button on the screen to shoot bullets from the spaceship and tries to kill enemies. On the other end enemy spaceships shoot bullets too in different patterns. Enemies will be harder to kill with every level increasing.

Some levels will be more difficult by presenting the player with bigger and stronger spaceships that are harder to be killed. These will be called Boss levels and will give more rewards to the player. The player can use the store to buy upgrades for the spaceship and also increase the lifespan or bullet amount.

If the player passes a level with certain specifications such as: not losing once during the level, it will gain some achievements. During the game the player will gain score based on the time spent in the level or number of enemies killed and then these scores can be saved with the player’s name.

The goal here is to kill enemies so to pass levels and reach as far as one can and beat the game.

## 

## **2.2** **Functional Requirements**

### 2.2.1 Play Game

Space Impact is a type of game that requires maximum attention from the player. The game is played by using one finger to control the spaceship by keeping it pressed on the screen and with another finger the player can press the shoot button or power up button.

* The player will have a life bar available in the beginning and game finishes if this bar becomes zero.
* The player loses some percentage of its life bar if one of the bullets of the enemy hits the spaceship. Every enemy type or bullet type have different effects on life bar.
* The level will be passed if the player has a non-zero life bar in the end of the level.
* Bullets will be in a limited amount when the game starts. If the player finishes all the bullets then it can avoid the enemies by dodging the bullets.
* During the game power ups will drop down such as: bullet ammo and the player can continue to shoot again. Different power ups will be available to the player during the game.

### 

### 2.2.2 Store

The player gathers coins during the gameplay and with these coins one can buy different power ups or upgrades in the store. The upgrades will have different costs, some of them require a large amount of coins in order to be bought. The game components that the player can buy include:

* Increase the capacity of life bar. The next time the game starts the player will have slowly decreasing life bar.
* Increase the amount of amo that the spaceship can have.
* Special weapons. If the player buys this power up then when the game starts the shots will be in special mode that shoot more dispersevely or harder.
* Spaceship upgrade. In this upgrade the player can choose the color for his spaceship and also increase durability of the ship.

### 2.2.3 Pause Game

While the game is continuing the player can choose to pause the gameplay and this is made easier because all the player has to do is to lift the finger off the screen. While the finger is not touching the screen the player has two options:

* Resume button. This will continue the game where it was left
* Main menu button. This will take the player to the main menu and lose the progress made until that point.

### 

### 2.2.4 View Information

#### 2.2.4.1 View Credits

The user can see who are the developers of the game and find information about them. Any of the program users can find the developers and contact them about any complaint about the game or new ideas on how to make the game more user friendly.

#### 2.2.4.2 View Tutorial

A player who is having difficulties in playing the game or wants to learn more about the types of enemies or bosses can view information related to his needs. The tutorial will include:

* General information about the game
* Explanations about the general controls
* Enemy types and bosses
* All the kinds of power ups and functions of them

### 

### 2.2.5 View Achievements

The player can complete different goals during the game and reach them. This allows one to gain some achievements and view them. At the end of each level the player will see the gained achievements if any. This feature makes the game more interesting and keeps the player interested in the game.

**2.3**  **Non-functional Requirements**

## 2.3.1 Performance

Space Impact will be an android game and this enables high quality graphics for the game. However, the game performance will be very smooth and the everyone can play without experiencing delays or glitches. The response time is quite immediate and the user will not experience lags. The system can accomplish a lot of tasks in a specified amount of time since the game will include many screen interactions with the user. The game will work in almost every android device containing Android 5.0 and above.

## 2.3.2 User-Friendly

In order to leave the users satisfied the game will be user-friendly. Anyone can learn the game within a few minutes and can play without needing special training. Most of the names in the game will be self-explanatory and easy to understand. It is very important to us to create a game that everyone can use frequently and likes it.

## 2.3.3 Reusability

While creating Space Impact we will implement also a lot of love embedded with other codes. Our project does not finish when this course is done. We will create Space Impact in a way that it can be extended and reused so that our work can continue from us or from others.

## 2.3.4 Additional requirements ( 2 iteration)

### 2.3.4.1 Usability

The user can easily learn to operate, interpret the outputs or prepare the inputs for a system or a component. To make the navigation through the system easier for the user some of the requirements for the usability will include conventions adopted by the user interface, online help, and user documentation. The user must be comfortable with the usage and to help this guidelines will be made clear.

## **2.4** **Pseudo Requirements**

* The game will be implemented in Java and Android
* Some game objects such as, the spaceship, will be done in Adobe Photoshop

## 

## 

## 

## 

## 

## **2.5** **System Models**

### 2.5.1 Use-Case Models

### 

**Name:** StartGame

**Participating actors:** Player

**Flow of events:**

1. Player selects level (include use case SelectLevels).

2. Player selects power ups from current inventory for the game (include use case SelectPowerUps).

3. System shows game play screen.

4. Player plays the game (include use case PlayGame).

**Entry Condition:** Player selects “Start Game” option from Main Menu.

**Exit Condition:** Player finishes the level and sees level summary.

**Name:** SelectLevels

**Participating actors:** Player

**Flow of events:**

1. Player sees a map with levels listed on it.

2. Player selects the level for game.

**Entry Condition:** Player selects “Start Game” option from Main Menu.

**Exit Condition:** Level is selected.

**Name:** SelectPowerUps

**Participating actors:** Player

**Flow of events:**

1. System shows existing power ups in players inventory.

2. Player selects limited numbers among all categories ( protection, health, ammo etc.).

3. Player presses “Continue” button.

**Entry Condition:** Player selects “Start Game” option from Main Menu.

**Exit Condition:** Power-up types and amounts are selected for usage in a particular game.

**Name:** PlayGame

**Participating actors:** Player

**Flow of events:**

1. System begins a countdown, backwards from three.

2. System starts the game.

3. Player controls the spaceship by using finger movements (include use case ControlSpaceship).

4. Player shoots the enemies (include use case ShootEnemy).

5. Player collects the coins and rewards (include use case Collect).

6. Player collects power-ups in game while playing (include use case Collect).

7. If player moves his finger away from the screen, System pauses the game (extend use case PauseGame).

8. Player activates by selected power-ups listed on game screen (include use case ActivatePowerUps).

9. Player plays the game until the level is completed.

10. System shows level summary (include SeeLevelSummary).

9. If the health bar of the Player becomes zero, game is over (extend use case GameOver).

**Entry Condition:** Player selects “Start Game” option from Main Menu.

**Exit Condition:** Player completes the level and starts a new game with different level, OR

Player loses and game is over.

**Name:** PauseGame

**Participating actors:** Player

**Flow of events:**

1. System displays Pause screen.

**Entry Condition:** Player moves his finger away from the screen.

**Exit Condition:** Player selects “Main Menu” to exit, OR

Player selects“Continue” to continue playing the game.

**Name:** GameOver

**Participating actors:** Player

**Flow of events:**

1. System displays GameOver screen.

**Entry Condition:** Player’s health bar becomes zero.

**Exit Condition:** Player selects “Main Menu” to exit, OR

Player selects“Replay” to replaying the game.

**Name:** ControlSpaceship

**Participating actors:** Player

**Flow of events:**

1. Player puts finger on spaceship.

2. Player slides his finger to anywhere on screen and spaceship goes there.

3. Player moves spaceship to places in order to shoot enemies, escape from enemy shots, collect rewards and coins or collect power-ups.

**Entry Condition:** Player put his finger on spaceship.

**Exit Condition:** Player moves his finger away from the screen.

**Name:** ShootEnemy

**Participating actors:** Player

**Flow of events:**

1. Player position spaceship across the enemy ship.

2. Player presses the shooting button.

3. If there is activated power-ups about weapons, special shooting occurs. Else, default bullets are used.

4. If enemy’s health bar becomes zero, enemy explodes.

**Entry Condition:** Player controls spaceship.

**Exit Condition:** Enemy explodes, special rewards and coins are scattered.

**Name:** Collect

**Participating actors:** Player

**Flow of events:**

1. Player moves spaceship on the elements that he wants to collect.

2. If there are coins, they are added to overall account of Player.

3. If there are power-ups, they are added to the current power ups list on the screen waiting to be selected later in game.

4. If there are rewards, they are saved and listed at the end of the level, in level summary.

**Entry Condition:** Player moves spaceship.

**Exit Condition:** Collected coins, rewards and power ups are added to the Player’s inventory.

**Name:** ActivatePowerUps

**Participating actors:** Player

**Flow of events:**

1. System lists selected and collected power-ups.

2. Player selects them while playing game.

3. System activates the power-up.

**Entry Condition:** -

**Exit Condition:** Selected power ups activated while playing game.

**Name:** ShopFromStore

**Participating actors:** Player

**Flow of events:**

1. System shows the list of upgrades and power-ups according to player’s level.

2. System shows current amount of coins Player has.

3. Player selects among them.

4. If the Player lack of enough coins, the selected option cannot be bought (extend use case NotEnoughCoins).

5. System applies the upgrades that Player bought immediately.

6. System puts power ups to Player’s inventory for future use.

**Entry Condition:** Player selects “Store” option from Main Menu.

**Exit Condition:** Upgrades and power ups are selected.

**Name:** ViewAchievements

**Participating actors:** Player

**Flow of events:**

1. System shows the list of achievements that Player achieved in previous games.

2. System shows special rewards that Player collected.

3. Player presses an achievement and sees detailed information.

**Entry Condition:** Player selects “Achievements” option from Main Menu.

**Exit Condition:** Player selects “Main Menu” option from Achievements screen.

**Name:** ViewInformation

**Participating actors:** Player

**Flow of events:**

1. System represens game tutorial and detailed information about game elements such as different types of enemies, power-upsand upgrades, rewards and achievements (include use case ViewGameTutorial).

2. System shows information about developers of the game (include use case ViewCredits).

**Entry Condition:** Player selects “About” option from Main Menu.

**Exit Condition:** Player selects “Main Menu” option.

**Name:** ViewGameTutorial

**Participating actors:** Player

**Flow of events:**

1. System represens game tutorial with main parts.

2. Player selects them to view detailed information.

3. System shows detailed tutorials with selected topic by providing in-game screens and videos.

**Entry Condition:** Player selects “Game Tutorial” option from About screen.

**Exit Condition:** Player selects “Main Menu” option.

**Name:** ViewCredits

**Participating actors:** Player

**Flow of events:**

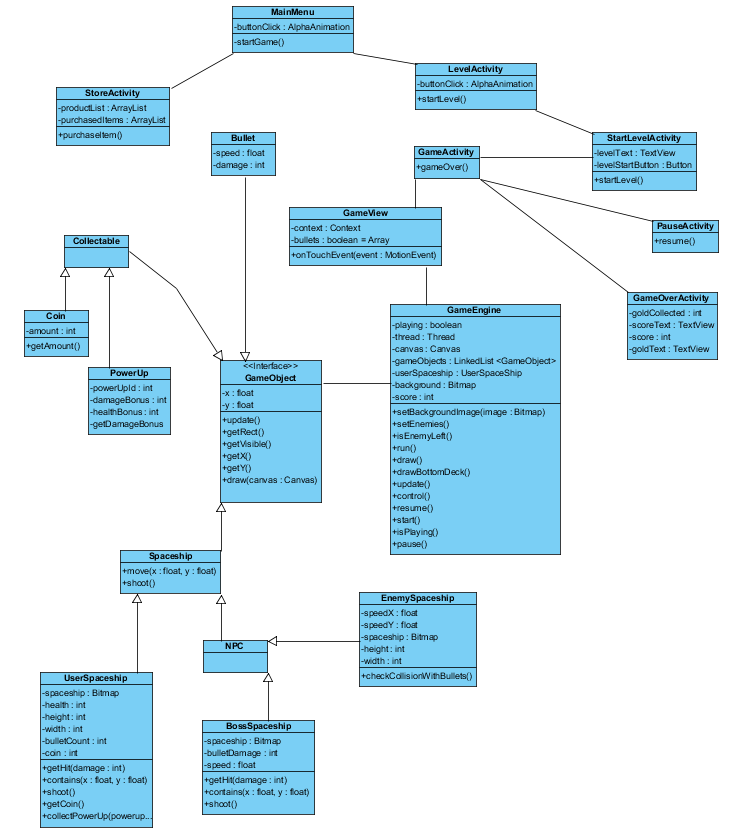
1. Player views credits.

**Entry Condition:** Player selects “Credits” option from About screen.

**Exit Condition:** Player selects “Main Menu” option.

### 

### **2.5.2** **Object and Class Model**

****

The object model of Space Impact is shown above. The game consists of 15 classes and 4 interfaces.

The interfaces of the game are Collectable, GameObject, SpaceShip and NPC.

The GameObject class covers every single object seen on the screen while the game is running. Every object of the game must have coordinates on the screen.

The SpaceShip covers UserSpaceship class and NPC interface as they are strictly spaceships.

In our game, NPC interface does not have any particular function other than gathering CPU controlled objects.

Collectable interface only gathers the objects that appear on the game screen and which can be taken by the user spaceship.

The functions of the other classes are:

* **MainMenu: The first activity that appears on the screen. The user can proceed to LevelActivity or StoreActivity.**
* **StoreActivity: The menu in which the user can buy power-ups for their spaceship.**
* **LevelActivity**: The section which shows the levels. The user can choose which level to play.
* **StartLevelActvity**: A confirmation screen to start the game.
* **GameActivity**: The activity which contains the GameView of the game. It initilizes the game.
* **GameView**: The canvas on which the game is drawn. It handles the touch events, as Android echosystem only allows us to take inputs through the canvas itself.
* **GameEngine**: The class in which the main game logic is running.
* **Coin**: Coins that appear during the game. They have a particular value.
* **PowerUp**: They can appear during the game and have a specific power-up identity which will clearify their use.
* **PauseActivity**: It starts whenever the user releases their finger of the screen. It pauses the game loop. Disposes when the screen is touched.
* **GameOverActivity**: When the user has either won or lost, this screen appears. It shows the coins collected, the achievements done, and the score.
* **UserSpacehip**: The main spaceship that the user can control.
* **EnemySpaceship**: The weaker enemies that appear randomly during the game.
* **BossSpaceship**: The more powerful enemy that only appears at the end of the game. It has more human-like features like shooting.

### **2.5.3** **Dynamic Models**

As our game is built upon the Android system, we are making the menus as activities, since they are easier to implement and more appetizing in terms of looks. Hence why in our class diagram, we did not put every UI component as attributes because the UI elements are View objects in Activity and they are statically declared on the activities’ .xml files. Almost every Boundary object of the diagrams are declared in these files.

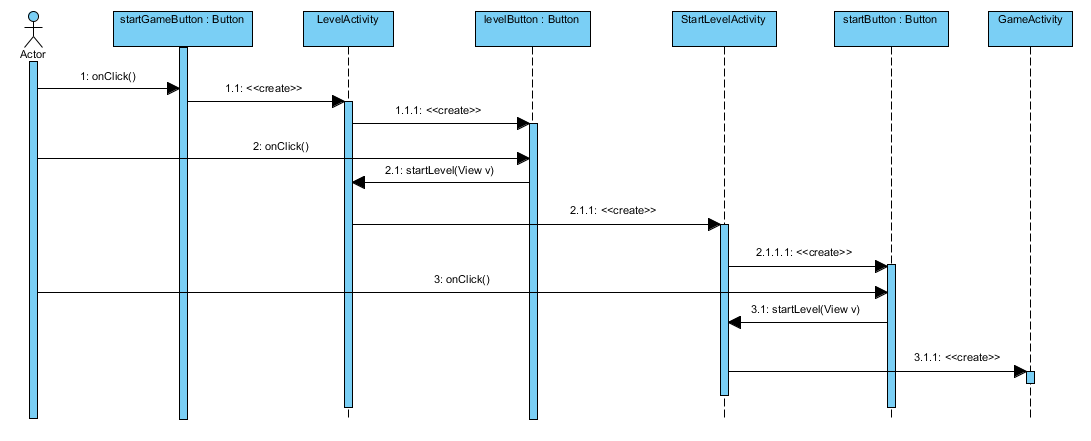


Figure: UML Sequence Diagram for StartGame Use Case

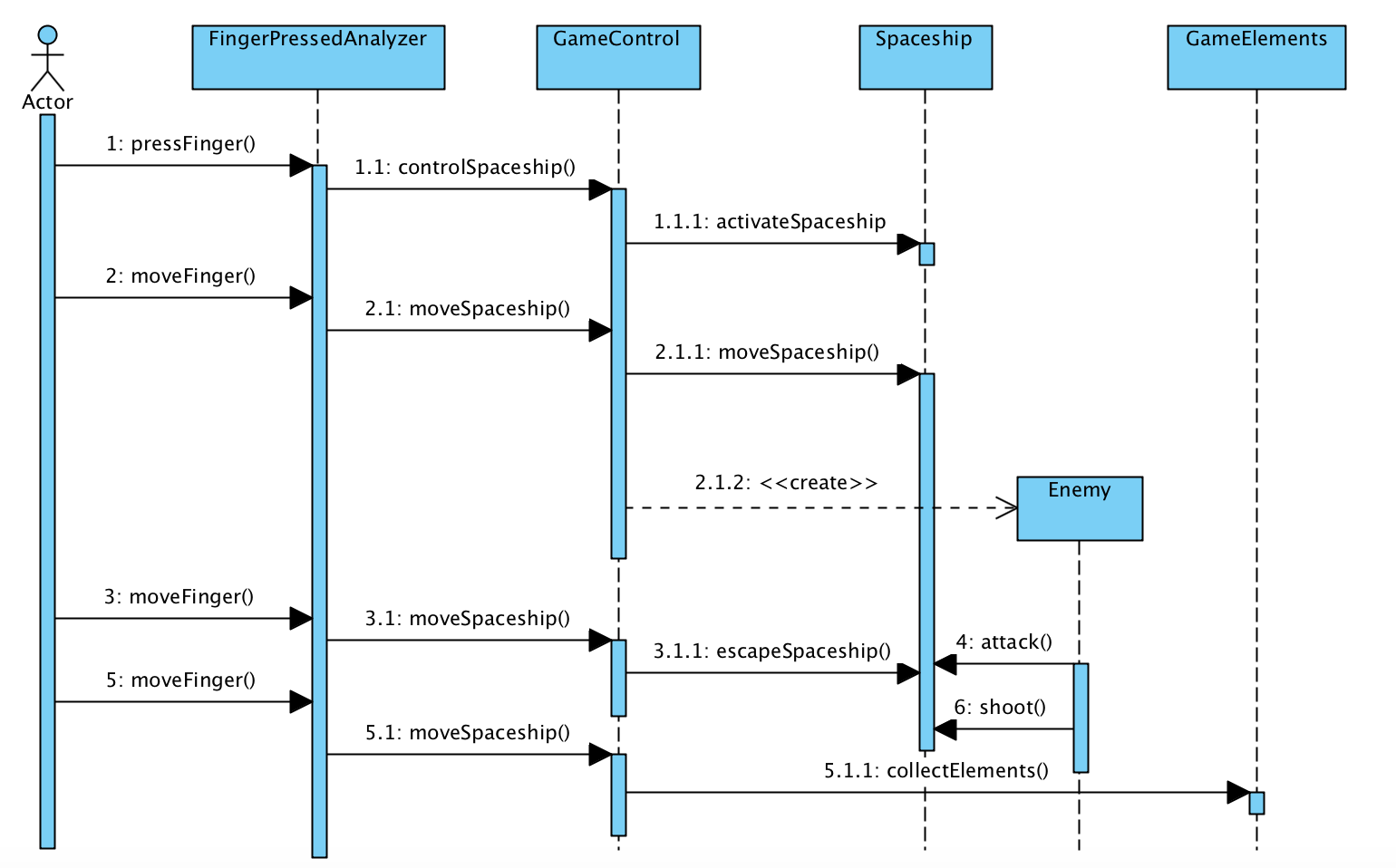


Figure: UML Sequence Diagram for PlayGame Use Case

### 

### 

### 

### **2.5.4** **User Interface**

**Figure 2.5.5.1 Screenshot of the Menu Screen**

**-Menu Screen**

When the application is first started, the menu screen will open. This screen has 3 main options; Start Game, Store and About Us.

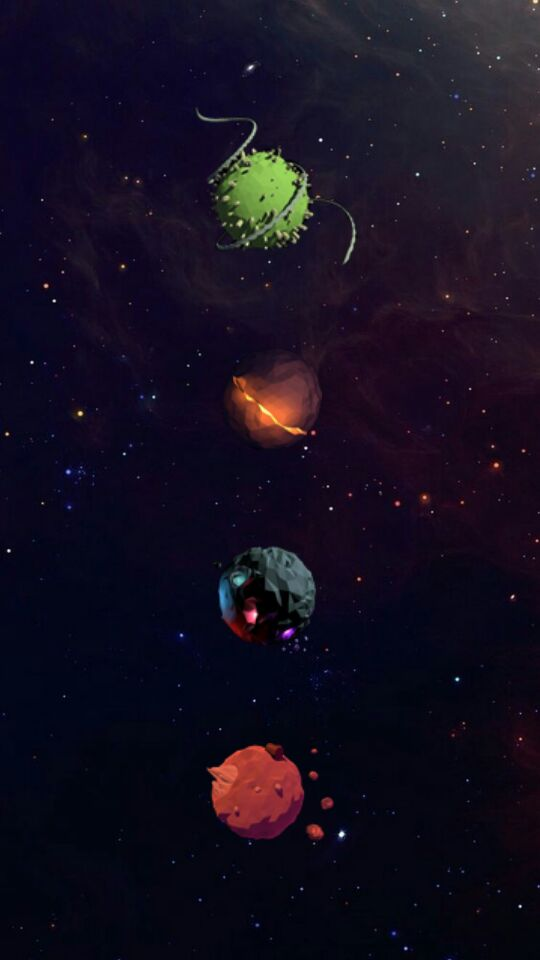
· **Start Game**: The player will be directed to the levels screen.

· **Store**: The store screen will show up.

· **About Us**: The user will see the information of the developers of the game.

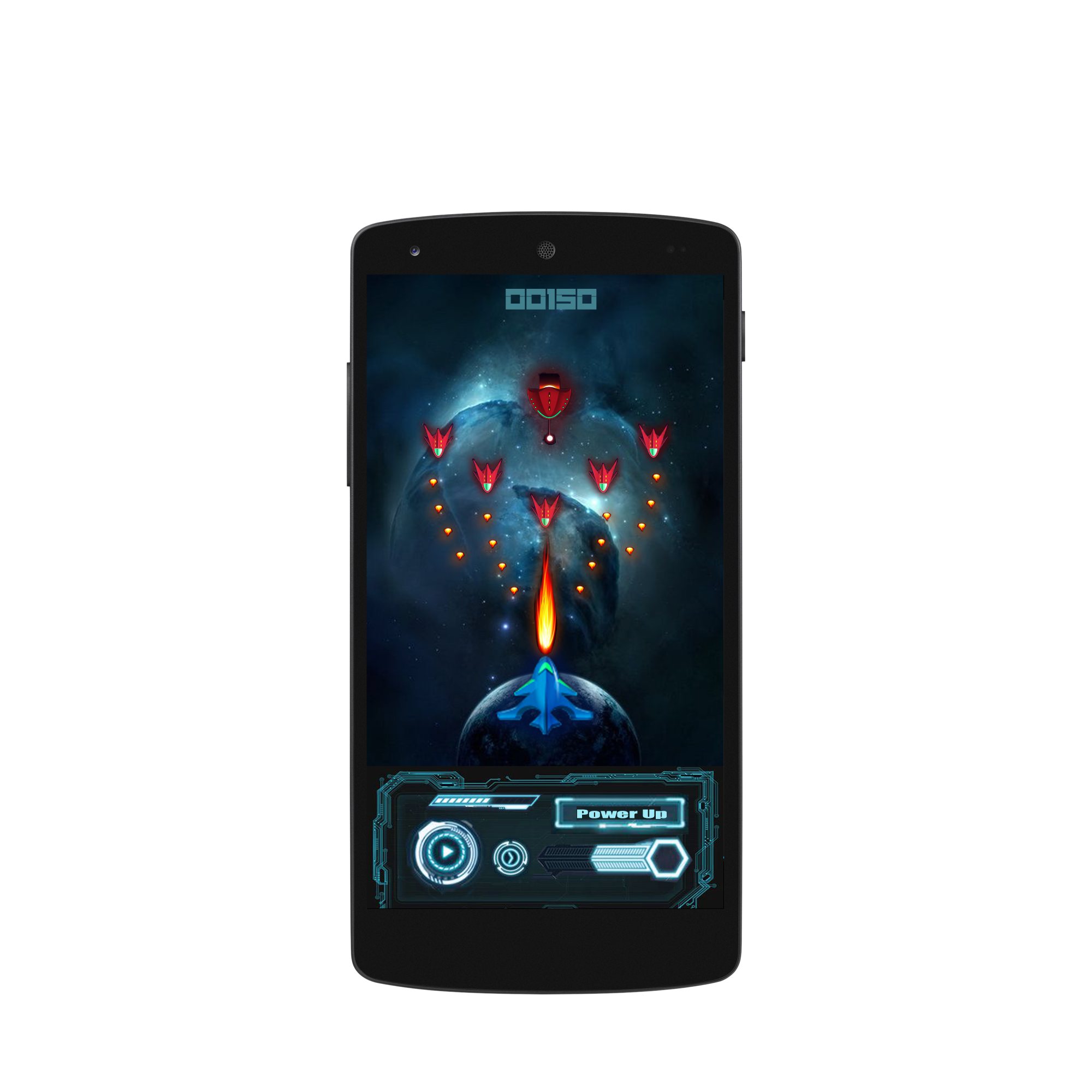
**-Levels Screen**

The user can choose the level which he/she wants to play. This screen also contains the powerup which the player wants to start the game with. During the game the user can activate this power up.



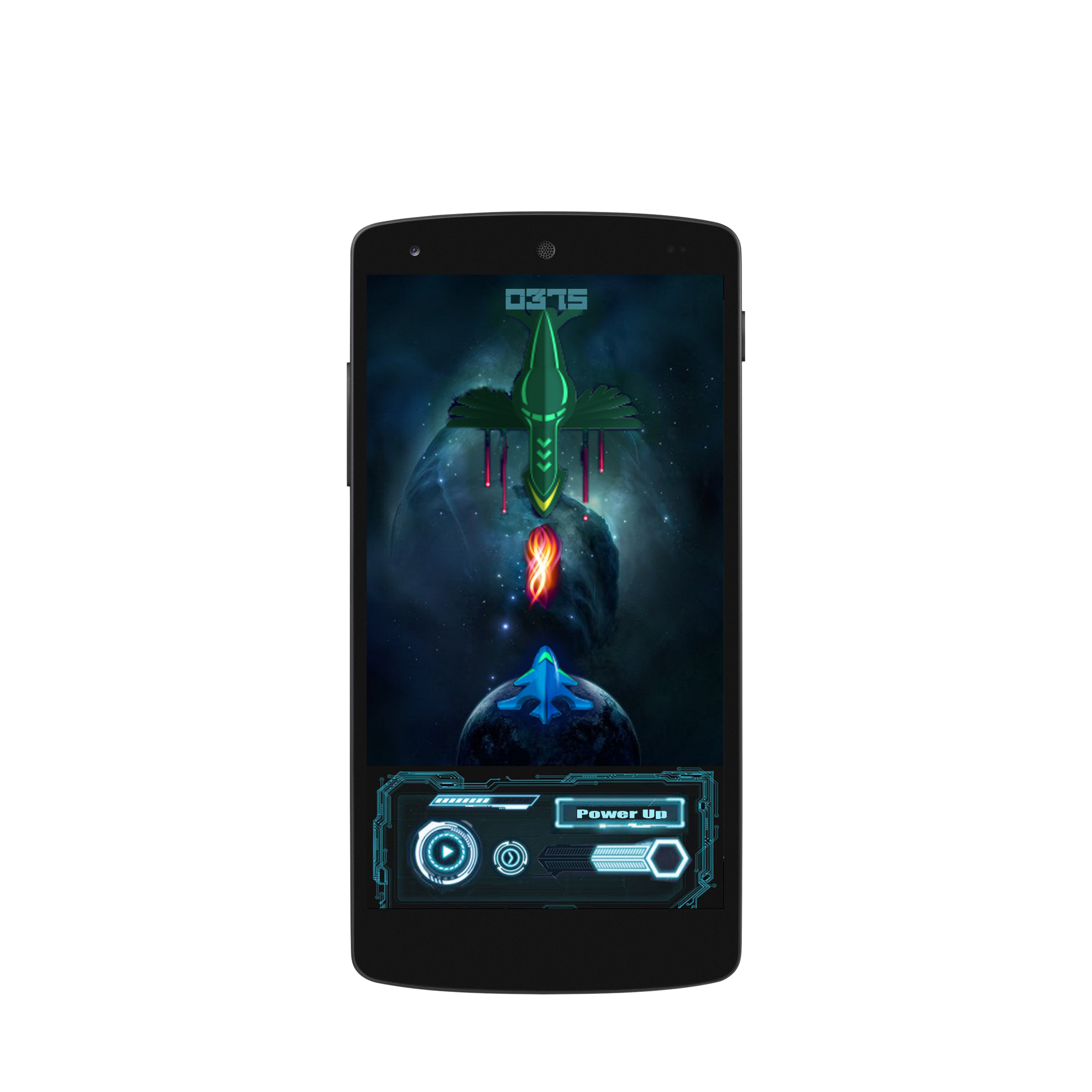
**Figure 2.5.5.2 Screenshots of the Game Screen**

-**Game Play**



**Figure 2.5.5.3 Screenshots of the Game Screen**

Regular game levels will have some basic enemy spaceships which are NPCs. The game screen will have an area on which the player can control the movement of the spaceship, and a dock which has a regular bullet shooting button, a powerup button, a monitor for the remaining health and another monitor for the remaining ammo. On the top of the screen, a score counter is located.



**Figure 2.5.5.4 Screenshot of the Boss Level Game Screen**

Particular levels will have a boss battle, which contain a powerful bigger and stronger NPC. These levels have the same user interface as a regular level.

**-Pause Menu**

**Figure 2.5.5.4 Screenshot of the Pause Menu**

The user can pause the game while playing by only releasing their finger from the screen. The game will automatically stop the game loop and open the pause menu. The pause menu contains 2 buttons:

· Continue: The game will start running as well as the game loop.

· Main Menu: The game loop will stop running and the user will be directed to the menu screen.

# 3 Improvement Summary

After the reclassification of the requirements, we arrived in the conclusion that the usability non functional requirement was necessary to be added. However, all of our functional requirements were complete and unambiguous. After the reprioritization of the requirements we decided that the store requirement needed to be more prioritized compared to the powerups requirement. All of the other use cases did not change. After the reevaluation of the high impact requirements it was decided that it was not needed a change in these requirements since everything was thoroughly analysed.

# 4 **References**

[1] Object-Oriented Software Engineering, Using UML, Patterns, and Java, 2nd Edition, by Bernd Bruegge and Allen H. Dutoit, Prentice-Hall, 2004, ISBN: 0-13-047110-0.