**Software Requirements Document for Cyfighter**

Author: Group 18

Edwin Benggawan

Zachariah Brase

Brendan McCluer

Yangxiao Wang

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Change** |
| 0.1 | 02/18/16 | Grp18 | Initial Document |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**TABLE OF CONTENTS**

[Introduction](#h.gjdgxs)

[Purpose](#h.30j0zll)

[Scope](#h.1fob9te)

[Definitions, acronymns, abbreviations](#h.3znysh7)

[References](#h.2et92p0)

[Overview](#h.tyjcwt)

[Overall Description](#h.3dy6vkm)

[Product Perspective](#h.1t3h5sf)

[Concept of Operations](#h.4d34og8)

[Major User Interfaces](#h.2s8eyo1)

[Example Screenshot and description](#h.17dp8vu)

[Hardware Interfaces](#h.3rdcrjn)

[Software Interfaces](#h.26in1rg)

[Communication Interfaces](#h.lnxbz9)

[Memory Constraints](#h.35nkun2)

[Operations](#h.1ksv4uv)

[Site Adaptation Requirements](#h.44sinio)

[Product functions](#h.2jxsxqh)

[Example use case](#h.irs95vqq28kz)

[User characteristics](#h.38dncrby9ua3)

[Constraints](#h.4i7ojhp)

[Assumptions and Dependencies](#h.2xcytpi)

[Specific Requirements](#h.1ci93xb)

[External Interface Requirements](#h.dadbt3hci1it)

[User Interfaces](#h.vvqxg6tare77)

[Hardware Interfaces](#h.18eip8wdr78y)

[Software Interfaces](#h.ooqvsjefznai)

[Communications Interfaces](#h.5mr3bc273toz)

[Features](#h.26azbzj6fov8)

[Performance requirements](#h.muu1w71wwir)

[Design Constraints](#h.q7eqsg1pj6zs)

[Software System Attributes](#h.thj7r5kku4b2)

[Reliability](#h.jza4ie6sy4o2)

[Availability](#h.tpri7wjg351s)

[Security](#h.1e9zsgdjiwsr)

[Maintainability](#h.2ehlww73rxcu)

[Portability](#h.qtvssbq4kqi4)

[Other Requirements](#h.v6makehh74cd)

# Introduction

## Purpose

The purpose of this document is to present a detailed description of the Cyfighter mobile video game on the Android OS. It will explain the features and variables, interfaces and screens, use cases and actors, and the constraints under which the system must operate. The target audience is intended for both stakeholders and developers of the system.

## Scope

The system will be a mobile video game for all users of android devices. This system is designed to allow users to play 10 different levels in a single player mode or co-op mode. The system will calculate and save the highest score for a particular level, as well as post the score to a global leaderboard. Using coins and Cys, the system will allow the user to purchase upgraded aircrafts.

## Definitions, acronymns, abbreviations

|  |  |
| --- | --- |
| Term | Description |
| Actor | Person who interacts with the system. |
| Administrator | User who has the privileges of a debugger, as well as changing prices, disabling and enabling features. |
| Android Studio | The IDE used for development of the game. |
| Appwarp | The Library used for multiplayer sessions. |
| Coin | A reward earned during levels of the game. The coins can be used to purchase upgrades in the game. |
| Co-op Mode | State in which the system is connected to Appwarp, sending and receiving messages to/from users**.** |
| Cy | A reward earned during levels of the game. The Cys can be used to purchase upgrades in the game. |
| Debugger | A Player user who is granted special privileges, i.e. access to all features, levels, and logs. |
| Host Player | A Player user who will act as the “host” in a multiplayer session (Player 1). In Co-op Mode, this user has higher privileges than the Joining Player**,** such as choosing the level to play, as well as the difficulty. |
| Joining Player | A Player user who will join the Host Player in a multiplayer session (Player 2). In Co-op Mode, this user has lower privileges than the Host Player. |
| LibGDX | The platform/library used to develop the specific gaming tasks, i.e. camera, sprites, game states, assets, animation, handling input, updating, and rendering. |
| MySQL | The database used by the system. |
| Leaderboard | The collection of high scores viewable by all players online. |
| Linux Server | The server contained online. |
| Player | A user with the lowest privileges, i.e. the consumer. |
| Room | A session created by Appwarp to hold multiple users on multiple systems, with whom will send and receive messages between them. |
| Server Database | The database used by the serverto hold leaderboard information. |
| Software Requirements Specification | A document describing all the functions of a system, as well as the constraints under which it must operate, i.e. this document. |
| SQLite | The database stored on an individual android system. |
| Stakeholder | Person with an interest in the project who is not a developer. |
| System | The android program running the software. |

## References

Appwarp Software: <https://apphq.shephertz.com/appWarp>

LibGDX Framework: <https://libgdx.badlogicgames.com>

## Overview

[NONE]

# Overall Description

Cyfighter is a fun, mobile arcade game designed for Android. It designed as a vertical, aircraft shooter. Like many of its popular counterparts, Cyfighter gives players the ability to choose multiple weapons, aircrafts, and bombs to take down their enemies. As the title implies, the game is focused on Iowa State University school pride! Red and yellow fighters as well as Iowa State collectable tokens are featured throughout the game. In addition, players can upload their highscores, choose difficulties, and play levels with a friend in Co-op mode.

## Product Perspective

The Google Play store has multiple mobile games that are similar to the system described in this document, i.e “STRIKERS 1999”, “Raiden war 2015”, “STRIKERS 1945-2”, “raiden 2048 HD”. These games are similar in that they are vertical scrolling games in which the user navigates the screen and eliminates enemies to complete the levels. There are multiple aircrafts that can be purchased.

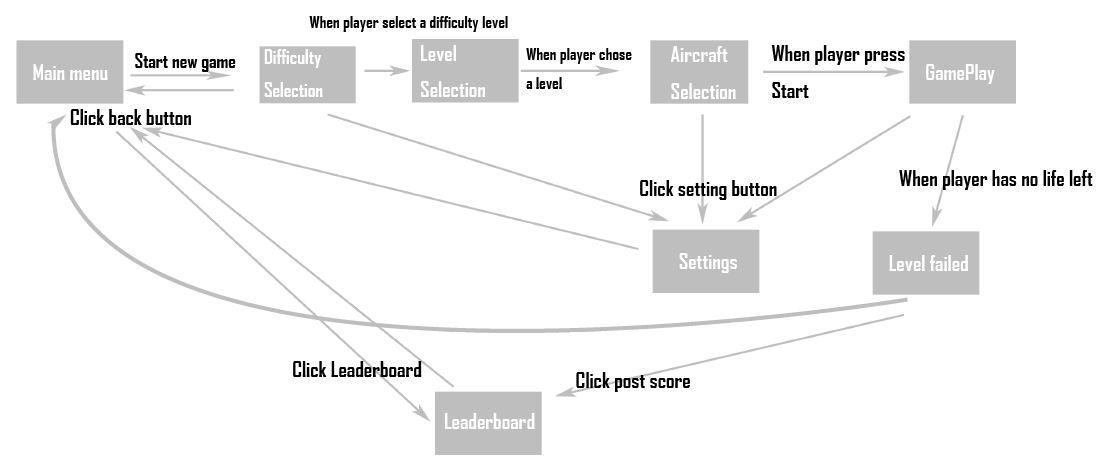
### Concept of Operations

Cyfighter will use Java code translated into android code by the LibGDX framework (see References). Information on the number of cys and coins will be stored in the system’s SQLite database. Unlocked features will also be stored in this database. It will access an online MySQL database hosted on the Linux Server to determine prices of weapons, bombs, and aircrafts, as well as leaderboard scores.

In addition, the system will use Appwarp to support co-op play. Appwarp is an online service used for hosting multiplayer sessions (see References). Using these sessions, players will connect two android devices and play levels with each other in real time.

### 

### Major User Interfaces



#### Example Screenshot and description

See the Appendix for a visual representation of each game screen outlined in the Major User Interfaces.

### Hardware Interfaces

The hardware required for this system is a physical device capable of running the Android OS with a minimum of Android SDK 8.

### Software Interfaces

Appwarp and Libgdx libraries will be used by the system for code development and co-op play.

### Communication Interfaces

[NONE]

### Memory Constraints

[NONE]

### Operations

[NONE]

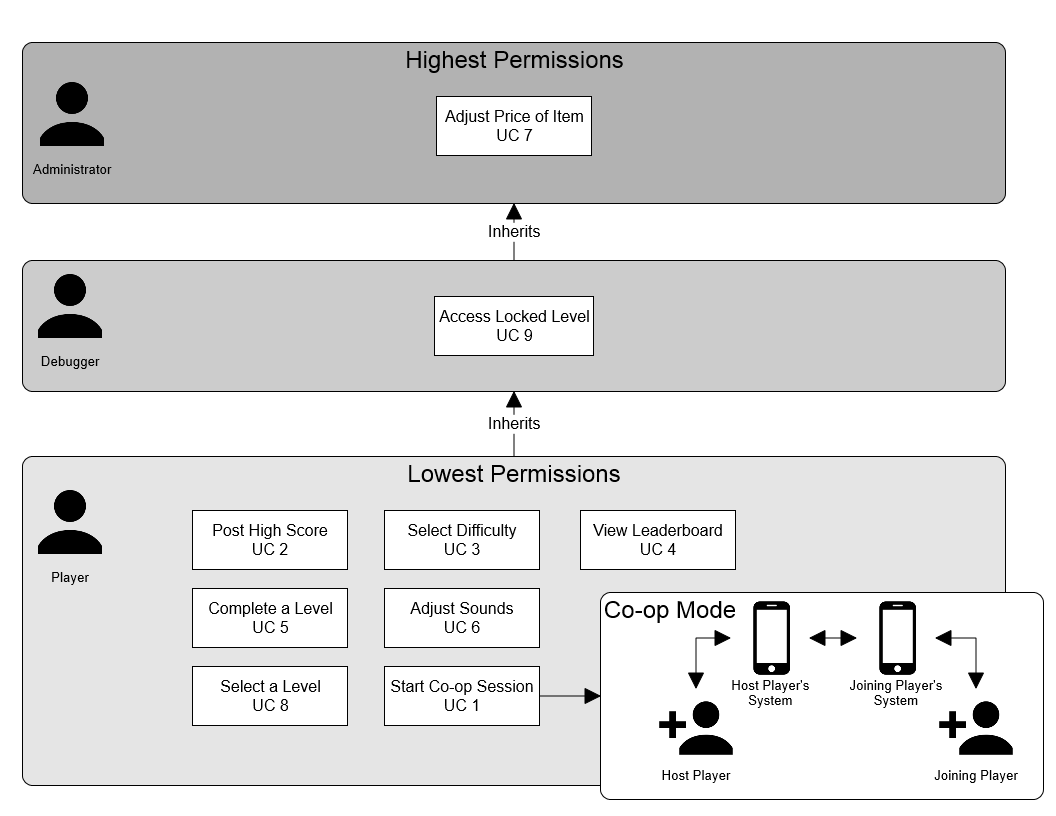
### Site Adaptation Requirements

[NONE]

## 

## 

## Product functions



### There are three main actors: **Player**, **Debugger**, and **Administrator**. The Player has the lowest level of access but is the most frequent user with many use cases. The Debugger is a special type of user that may access any locked feature (levels, weapons, and aircrafts) for testing purposes. All other use cases are inherited from the Player. The user is identified by entering a unique debug password into the system. The Administrator is a special type of user that inherits Debugger and Player use cases. In addition, the Administrator may adjust prices of levels, weapons, and aircrafts. The user is identified by entering a unique admin password into the system.

The major use case for this system revolves around the Player. The Player is the central focus and plays the game to experience all enjoyable features the it has to offer.

### 2.2.1 “Start Co-op Session” –**UC 1** (Start a level in co-op multiplayer mode, **Host Player**/**Joining Player**, Brendan McCluer)

In co-op mode, two aircrafts are controlled side by side with two players. Each player has their own copy of the game, selects their aircraft and weapons, controls their own aircraft, and posts their initials onto the leaderboard. This use case defines the steps to connect the two players and begin a session.

Main Scenario

1. **Host** **Player** selects “Co-op” from the Start screen.

2. **Host Player** selects “Host Session”.

3. **Host Player’s system** creates a new **room**.

4. **Host Player’s System** displays the **room ID.**

5. **Joining Player** selects “Co-op” from the Start screen.

6. **Joining Player** selects “Join Session” and enters the **room ID.**

7. **System** enters the **Joining Player** into the **room**.

Extensions

1a. **Host Player** is an **Administrator** or **Debugger.**

1a1. System gives the **Host Player Administrator** or **Debugger** privileges (ability to select any aircraft, weapon, and level).

3a. Failure in creating a new **room**

3a1. System displays an error message to the **Host Player** and exits.

5a. **Joining Player** is an **Administrator or Debugger**

5a1. System gives the **Joining Player Administrator** or **Debugger** privileges (ability to select any aircraft and weapon)

7a. Failure in joining a **room**

7a1. System displays an error message to the **Joining Player** and exits.

### 2.2.2 “Post High Score” –**UC 2** (Upload the score after game over, **Player**, Brendan McCluer)

Upon reaching the Game Over screen, the user may view the Leaderboard Screen. From here the user can post his/her score to the **server** where it may be viewed by other players.

Main Scenario

1. **Player** selects emblem and types initials.

2. **Player** submits to the **leaderboard**.  
 3. **System** sends the emblem and initials to the **server**.

4. **System** displays success message to **player**.

Extensions

3a. **Server** is offline/fails

3a1. **System** displays “server unavailable” message to **player**.

2.2.3 “Select Difficulty” –**UC 3**

(Choose a difficulty level, **Player**, Edwin Benggawan)

The Difficulty Selection screen appears after the user starts a new game. There are three levels of difficulty: easy, normal, and hard. The **system** will set the multiplier for obstacles corresponding to the chosen difficulty.

*Main scenario:*

1. **Player** selects one of the three difficulty levels (easy, medium, or hard).
2. **System** sets the difficulty multiplier.
3. Screen is displayed with UI to show the available levels based on the selected difficulty.

*Extensions:*

[None]

2.2.4 “View Leaderboard” **–UC 4**

(View scores from previous games, **Player**, Edwin Benggawan)

The user can view worldwide single scores and co-op scores as well as scores for different difficulty levels using tabs. This screen appears by choosing “Leaderboard” from the main menu. The user can also view the Leaderboard screen after a game ends.

*Main scenario:*

1. **Player** selects “Leaderboard” from the Start screen.
2. **System** retrieves data from **Server Database**.
3. Screen displays the scores along with emblems and initials for single player and easy level by default.

*Extensions:*

2a. **Server** is offline/fails:  
 2a1. **System** displays “server unavailable” message to **player**.

3a. **Player** selects other difficulty levels tab and/or co-op:   
 3a1. Screen displays the scores with emblems and initials for the selected

difficulty level and/or co-op.

2.2.5 “Complete a Level­­” **–UC 5**

(Complete a current level unlock next level, **Player,** Yangxiao Wang)

Once player start playing a level/map, user can complete or fail the level.

*Main scenario:*

1. **Player** start a level/map and Select Difficulty.

2. **Player** select a fighter and start playing.

3. **Player** survived and defeat a “Boss”.

4. Level is cleared and **system** display level complete screen and unlock next level.

*Extensions:*

3a. Player lose all lives.

3a1. **System** displays level failed screen.

3b. Player quit current game

3b1. **System** will not store the progress so player will need to start from the beginning of the level.

2.2.6 “Adjust Sounds” - **UC 6**

(Adjust volume level for sound effects, music, **Player**, Zach Brase)

The user can adjust the volume for both the sound effects and music in increments

of 10%, ranging from 0%-100%.

*Main Scenario:*

1. **Player** clicks on the settings icon in the top right corner of the screen (this icon is present in *every* screen in the game).
2. **Settings screen** is displayed to the **Player.**
3. The player clicks on the volume bar (for either MUSIC or SOUNDS) at a point representing the percentage volume that is intended.
4. **System** variables are updated appropriately, i.e. a floating point value is sent to the sounds class associated with either MUSIC or SOUNDS).

Extensions:

1. **Player** clicks on the icon to the *left* of the volume bar (for either the MUSIC or SOUNDS) and the volume will be set to 0%, i.e. mute.
2. **Player** clicks on the icon to the *right* of the volume bar (for either the MUSIC or SOUNDS) and the volume will be set to 100%, i.e. max volume.

2.2.7 “Adjust Price of Item” - **UC7**

(Change the price of item in game include: planes, weapons, unlocking maps/level. **Administrator,** Yangxiao Wang)

**Administrator** has top level authority: unlock all levels, weapons, planes and adjust the price of item in game. The user can enter certain password in game and the game would be changed to admin mode.

*Main scenario:*

1. **Admin** enters certain password.

2. Password is validated.

3. **System** switches toadmin mode (unlock all level, weapon, fighter).

4. **Admin** enters store/shop and adjust price of item.

5. Price of **item** is changed.

*Extension:*

2a. Passcode is not validated.

2a1. Display error message.

2.2.8 “Select Level” - **UC 8**

(Select the level to play, **Player**, Zach Brase)

The **Player** can select one of the 10 levels to play.

*Main scenario:*

1. **Player** navigates to “Select level” screen (see Section 2.1.2).
2. **Player** clicks on one of the ten levels.
3. **Player** is directed to “Aircraft Selection” screen (see Sections 2.1.2)

Extensions

1. A level is locked and is not accessible.

2.2.9 “Access Locked Level” - **UC 9**

(Select the level to play, **Debugger**, Brendan McCluer)

**Debugger** has medium level authority: unlock all levels, weapons, and planes. Even if a level is locked, the **Debugger** can play it in debug mode.

*Main scenario:*

1. **Debugger** enters certain password.
2. Password is validated.
3. **System** switches todebug mode (unlock all levels, weapons, fighters).
4. **Debugger** navigates to “Select level” screen (see Section 2.1.2).
5. **Debugger** clicks on one of the ten levels that is locked.
6. **Debugger** is directed to “Aircraft Selection” screen (see Sections 2.1.2 and 2.2.9)

Extensions

2a. Passcode is not validated.

2a1. Display error message.

## User characteristics

[NONE]

## Constraints

[NONE]

## Assumptions and Dependencies

[NONE]

# Specific Requirements

## External Interface Requirements

[NONE]

### User Interfaces

[NONE]

### Hardware Interfaces

[NONE]

### Software Interfaces

[NONE]

### Communications Interfaces

[NONE]

## Features

[NONE]

## Performance requirements

[NONE]

## Design Constraints

[NONE]

## Software System Attributes

### Reliability

[NONE]

### Availability

[NONE]

### Security

[NONE]

### Maintainability

[NONE]

### Portability

[NONE]

## Other Requirements

[NONE]