



**Software Design & Development**

Major Project ‘Pius-mon’

Analysis and Design Report

# Table of Contents

[**Table of Contents**](#_q4q8fcms8943) **2**

[**Requirements**](#_kfwxiipbnvbl) **3**

[Introduction](#_q49uk8v40q3h) 3

[Client Needs](#_gt10vwosuqt1) 3

[Functional Requirements](#_g7kodkg6lmuk) 5

[Non-Functional Requirements](#_b518ev2yxtqa) 6

[Gantt Chart](#_1jynhvci6msj) 7

[**Storyboards**](#_uv5bb6keuqew) **8**

[Main Menu](#_nlctqyy9wv7e) 9

[Team Select Menu](#_ss2g9gehln4z) 10

[Options](#_5umr5ka0sbdr) 11

[Credits](#_eegnh4hnbm3a) 12

[Level Preparation](#_glaiw2qxh1fx) 13

[Battle Screen](#_bvfqtt19231b) 14

[**Design Report**](#_pxlxyrh7v3ud) **16**

[Combat Algorithm Diagram](#_1juhw7jqsxxk) 16

[Character Select Algorithm](#_tbx5b9mnr9e3) 17

[Data Dictionary](#_rvfve88iybho) 18

[IPO Diagram](#_fy5sf2s8bvuz) 21

# Requirements

## Introduction

This section of the document will present the requirements to produce an interactive ‘Pius-mon’ style video-game.

Included in this section will be the client needs, a requirements list, and a timeline Gantt chart which documents the development timeline of which the project must abide by to be on schedule.

## Client Needs

The client needs are as follows:

* Combat system consisting of:
  + Rock- paper-scissors counter mechanic
  + Character selection
  + Character swap
* Graphical User Interface (GUI)

**Types:** Three types of Pius mon attacks

1. **Rock**:
   1. Attacks extremely effective against scissors (double damage).
   2. Weak attacks on paper (half damage)
2. **Paper**:
   1. Attacks extremely effective against rock (double damage).
   2. Weak attacks on scissors (half damage)
3. **Scissors** :
   1. Attacks extremely effective against paper (double damage).
   2. weak attacks n rock (half damage)

**Stats:** Each Pius mon will have 3 stats which will combine to 100 points

1. **Speed:** will determine who hits first
2. **Attack:** how strong each monster hits
3. **Life:** how long they will survive

2 Pius mon for their trainer bag

* 1 will be primary mon
* 1 backup mon

Each battle will take place between 2 trainers as follows:

* + primary mon launched attack / special attack
  + swap made
  + results made
  + new mon released if needed

Order of attack

* when in battle a trainer may choose 2 things
  + attack or
  + swap.
* if they choose attack
  + a monster hits for
    - damage = to their attack value (modified by type)
  + the monster with the fastest speed will hit first.
* if you choose to swap and your monster is the fastest he may swap and not get hit
* if you choose to swap and he is the slowest he will take the damage from the attack first.
* trainers may only swap 3 times per battle
* If a PiusMon is Knocked out he is instantly switched with the next PiusMon
* If there are no other Piusmon left the other player is the winner

Each PiusMon must have 3 graphics

1. normal state
2. attacking state
3. getting hit

## 

## Functional Requirements

The system must consist of these features:

A complete main menu screen must serve as the foundation of user interactivity by:

* Being the first screen upon system start;
* Have navigation to all relevant screens:
  + Options Menu
  + Credits Menu
  + Team Select Menu
* Be the fallback screen in case of any system failure which results in a screen error;
* Exit button which closes the program;

The options menu will allow for adjustment of the volumes of sound effects and music separately. There is a button to navigate back to the main menu.

The team select menu allows the selection of two characters. After two characters are selected by clicking on them, the start game button will appear. Clicking this button will bring the user to the level select screen.

The level select screen will consist of one button which will bring them to the next level. The button will always direct the user to the level after the last completed. It will count each level completed, and will direct the user to the subsequent level. The user will automatically be directed to this page after a level complete. After the final (5th) level is completed, the credits screen will instead be the destination.

In the battle screen, two buttons will be available to the user; action and swap. Clicking the action button will initiate the ‘action bar’ mechanic, consists of a marker moving from the left to the right on a multi-layered scale / bar graphic. When the user clicks or presses space, the bar will stop and gather a value depending on the position of the marker.

* Each layer relates to a particular attack effectiveness:
  + **Critical hit**, the smallest layer, in the exact middle. **200%** damage multiplyer.
  + **Headshot**, slightly larger, behind the critical hit layer. **150%** damage multiplyer.
  + **Body shot**, larger than headshot, usually occupying 20-35% of the bar’s total length. Behind both upper layers. **100%** damage multiplyer.
  + **Limb shot**, larger than body, occupying between 40-70% of the bar’s total length. Behind both upper layers. **70%** damage multiplyer.
  + **Miss**, occupies the rest of the bar. **0%** damage multiplyer.

A value is calculated based on the action bar value, the character’s base damage value, and the character type multiplyer. This forms the final damage value inflicted on the health points of the opposing character.

When the hitpoints of the opposing character hits zero or goes into negative values, it is removed from play, until either the user’s characters or the opposing characters total amount are reduced to zero, of which either a defeat or a victory will be portrayed respectively.

The entire ‘campaign’ will consist of 5 subsequent levels, one after the other. Level 1 will consist of 1 enemy, level 2 of two enemies, and so forth, until the 5th level, which will consist of a boss.

In the options screen, the user will be able to use sliders to change the volume of sound effects and background music separately.

## Non-Functional Requirements

Buttons must follow a consistent style which is recognisable to the user.

* Font is consistent and distinct
* Buttons are boxed or highlighted
* Font is very large or follows a large profile

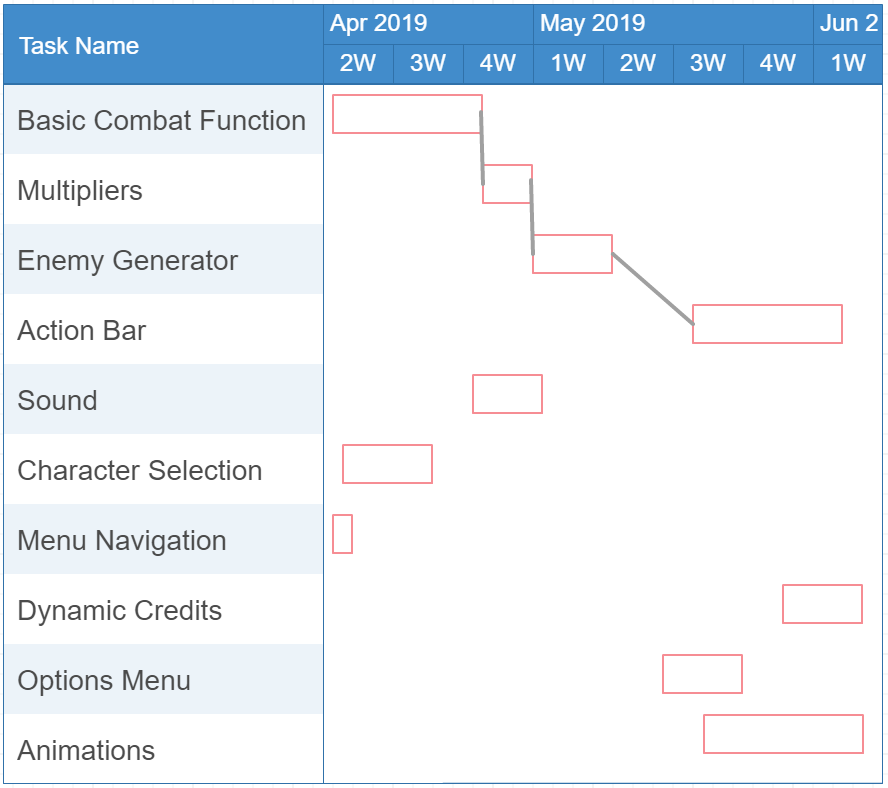
Graphics follow a consistent art style:

* Player characters use blues, while enemies use red
* Blues are commonly used in menus to show that they are player-aligned
* Contrasting colours and shapes are visually attractive and appealing, as well as create contrast to be refreshing and have function be visually apparent

Technical:

* Godot used as the language and engine to build the system.
  + Short term data storage in client, as saving is not necessary.
* All graphics are to be produced from scratch.
* Music sourced from external sources, allowed for personal use. Artists include:
  + Two Steps From Hell
  + Thrice

## Gantt Chart



# Storyboards

All storyboards consist of high-fidelity concept renders which are **not** screenshots. They may or may not retain high visual similarity with the finished product, and only serve to illustrate the capabilities and possible potential of the product.

**Annotation Colour Guide**

|  |  |
| --- | --- |
| **Colour** | **Meaning** |
| Blue | Interactive element eg. button |
| Yellow | Non-interactive element eg. title |
| Green | Indicative of movement of element associated |
| Red | Informative user-feedback visual element eg. selection bar |
| Other Colours | Refer to individual storyboards for meanings |

|  |
| --- |
| Main Menu |
|  |
| Main menu of program.  Buttons work as follows:  ‘NEW GAME’ leads to Team Select Menu  ‘OPTIONS’ leads to Options Menu  ‘CREDITS’ leads to Credits Menu  ‘QUIT’ closes program  Title / logo maintains a static position on the menu. |
| Mouse cursor hovers over button to select it. When hovering over a button, a wider black bar will move behind the button text with a fixed tween time of 0.1 seconds.  The background city will scroll across very slowly to the left.  As a considered option, a parallax effect may be applied on the city buildings to create depth and increased polish. |
| Audio  On hovering over button, ‘tick’ sound will play.  On clicking button, ‘beep’ sound will play.  Background music: Illusions - Thomas Bergersen (Two Steps From Hell) |

|  |
| --- |
| Team Select Menu |
|  |
| Team selection determines which agents are available in battle for the player.  Bottom left arrow button redirects to the Main Menu Screen.  Clicking on each character ‘card’ selects it. Tab at the bottom of each card turns green to indicate selection.  When two characters are selected, bottom right ‘DEPLOY’ button becomes visible. Clicking the button redirects to the Level Preparation Screen.  Depicts character information such as name, type and DEF, ATK, and SPD statistics.  Depicts character type. Circle = rock, triangle = scissors, square = paper. |
| Transition between Main Menu Screen and Team Select Menu involves background zooming out to shown perspective and the black stylised border converging into view as shown, along with the character ‘cards’ dropping from above. This transition happens in unison and should take 1 second.  Background movement remains consistent with main menu. |
| Audio  On hovering over button, ‘tick’ sound will play.  On clicking button, ‘beep’ sound will play.  Background music: Continuous from Main Menu Screen |

|  |
| --- |
| Options |
|  |
| Options menu. More options may become available in the final version.  Menu title.  Clicking and dragging the white box allows the user to adjust the volume of either music and/or sound effects. The ‘BACK’ button returns the user to the main menu screen.  The translucent white bar will represent the orientation and nature of the slider bar. |
| The transition between the main menu screen and the options screen will consist of the translucent grey box with all its contents fading in.  Background movement remains consistent with main menu. |
| Audio  On hovering over button, ‘tick’ sound will play.  On clicking button, ‘beep’ sound will play.  Background music: Continuous from Main Menu Screen |

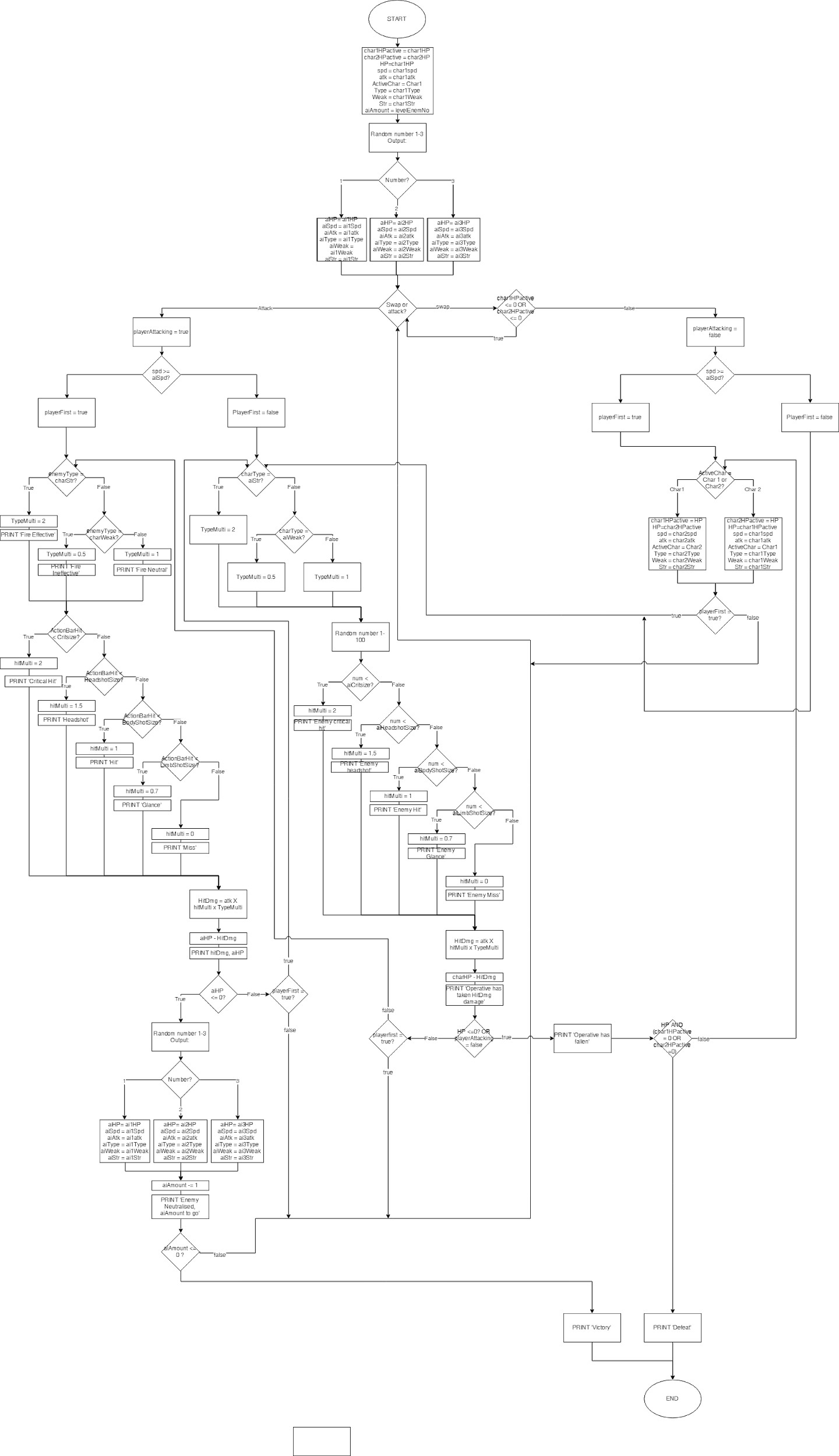
|  |
| --- |
| Credits |
|  |
| Credits screen.  The text in the middle will fade out to fade in the next credit after a few seconds, until all the credits are done.  Pressing any key will return the user to the main menu. |
| Credit categories are displayed individually, using a cross fade transition.  Background movement remains consistent with main menu. |
| Audio  On hovering over button, ‘tick’ sound will play.  On clicking button, ‘beep’ sound will play.  Background music: Red Sky - Thrice |

|  |
| --- |
| Level Preparation |
|  |
| Level select screen.  The solid blue dots represent cleared levels. The glowing dot represents the next level. The grey dots represent unbeaten levels. The blue line lengthens to follow the glowing dot. These are not interactive, and only serve to illustrate progress.  The ‘ADVANCE’ button will initiate the next level. Level 1 will consist of 1 enemy, level 2 of two enemies, and so forth, until the 5th level, which will consist of a boss.  Title. |
| Hovering over the ‘ADVANCE’ button will create the glow. Not hovering over it will remove the glow. |
| Audio  On hovering over button, ‘tick’ sound will play.  On clicking button, ‘beep’ sound will play.  Background music: Raxez 19 - Two Steps From Hell |

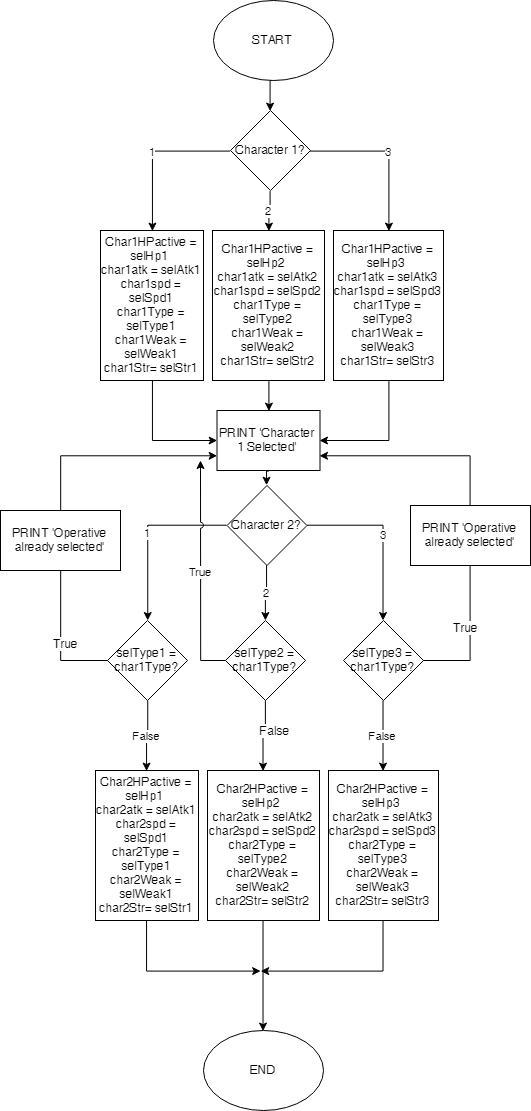
|  |
| --- |
| Battle Screen |
|  |
| The combat sequence screen.  The current selected character.  Represents various status effects. Status effects will be considered an optional addition and may not be present in the final product. Status effects are intended to enhance or hinder a characters performance, such as damage-over-time, lowered accuracy, or increased defence.  A display of the selected characters ‘type’.  A dynamic description of the events of the combat sequence, describing an attacks effectiveness, imparted status effects, swaps, etc.  This section displays information linked to the current character. It displays type, status of backup characters, active status effects, and a visual hit-point bar with the coloured bar representing the hit-point percentage.  Graphical representations of the 3 main buttons. The ‘ACTION’ Button will transform the 3 buttons into several buttons, consisting of various attacks and a back button. The ‘ITEM’ Button will transform the buttons into a list of consumable items. This is an optional addition and may not be present in the final product. The ‘SWAP’ Button will swap the current active character with the secondary character.  Dynamic visual representation of the unit count. Each visible bar represents 1 unit. Each coloured bar represents an alive unit, while a grey bar represents an eliminated unit. |
| The background will be dependent on the level chosen.  The character will consist of 4 sprites, which will represent idle, action, special action, and taking damage. An brief animation of the current character sliding backwards and the new character sliding forwards will also play to signify a swap. |
| Audio  On hovering over button, ‘tick’ sound will play.  On clicking button, ‘beep’ sound will play.  For each action, sounds will play for feedback.   * Gunfire will have gunfire sounds * Hits will have hit sounds. Different types of hits will use different sounds   Background music cycles randomly: By Two Steps From Hell   * A Hole in the Sun * Global Waste * She Sees the Future * Walking on Air * You Will Count Your Dead * The Frail * Bourne in the Future * Hot Cargo * Timelapse * Dark Ages * Low G Mission   Final boss background music: Corruption |

# Design Report

## Combat Algorithm Diagram



## Character Select Algorithm



## Data Dictionary

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Data Format** | **Field Size** | **Description** | **Example** |
| char(x)HPactive | integer |  | 3 | Current dynamic HP retained while idle for particular mon | 68 |
| char(x)HP | integer |  | 3 | Max Hit points for particular mon | 100 |
| HP | integer |  | 3 | Current dynamic HP for equipped mon | 45 |
| spd | integer |  | 2 | Speed value of equipped mon | 15 |
| char(x)spd | integer |  | 2 | Speed value of a particular player mon | 25 |
| spd | integer |  | 2 | Attack value of equipped mon | 35 |
| char(x)atk | integer |  | 2 | Attack value of a particular player mon | 20 |
| ActiveChar | text |  |  | Stores name of currently equipped mon | Nomad |
| char(x)Type | text | T | 1 | First letter of a mon type: eg. R = rock, P = paper, S = scissors | S |
| Weak | text | T | 1 | The type the equipped mon is weak to | R |
| Str | text | T | 1 | The type the equipped mon is strong against | P |
| char(x)Weak | text | T | 1 | The type a particular player mon is weak to | S |
| char(x)Str | text | T | 1 | The type a particular player mon is strong against | P |
| aiAmount | integer | T | 1 | The amount of enemies left alive on current level | 3 |
| levelEnemNo | integer | T | 1 | The amount of enemies a particular level begins with | 4 |
| aiHP | integer |  | 3 | Current HP for equipped enemy mon | 45 |
| aispd | integer |  | 2 | Speed value of equipped enemy mon | 15 |
| ai(x)spd | integer |  | 2 | Speed value of a particular enemy mon | 25 |
| aiSpd | integer |  | 2 | Attack value of equipped enemy mon | 35 |
| ai(x)atk | integer |  | 2 | Attack value of a particular enemy mon | 20 |
| ai(x)Type | text | T | 1 | First letter of a mon type: eg. R = rock, P = paper, S = scissors | S |
| aiWeak | text | T | 1 | The type the equipped enemy mon is weak to | R |
| aiStr | text | T | 1 | The type the equipped enemy mon is strong against | P |
| ai(x)Weak | text | T | 1 | The type a particular enemy mon is weak to | S |
| ai(x)Str | text | T | 1 | The type a particular enemy mon is strong against | P |
| playerAttack | boolean | T | 1 | Indicator of the nature of a player turn, whether it is an attack or a swap | true |
| hitMulti | Floating point | N.N |  | Multiplier regarding the ‘action bar’ mechanic | 2 |
| typeMulti | Floating point | N.N |  | Multiplier regarding the ‘type’ mechanic | .5 |
| hitDmg | Floating point |  | 3 | Final calculation of damage inflicted | 75 |
| playerFirst | boolean | T | 1 | Indicator of the order of a player turn, whether the player or an enemy attacks first | F |
| ActionBarHit | integer |  | 3 | Indicator of the zone of the hit on the action bar | 34 |
| Critsize | integer |  | 3 | Size of the critical hit range. 2x damage multiplier | 3 |
| Headshotsize | integer |  | 3 | Size of the headshot hit range. 1.5x damage multiplier | 8 |
| bodyshotsize | integer |  | 3 | Size of the body hit range. 1x damage multiplier | 30 |
| limbsize | integer |  | 3 | Size of the limb hit range. 0.7x damage multiplier | 60 |
| selHP(x) | integer |  | 3 | Base HP of that character in character selection | 300 |
| selAtk(x) | integer |  | 3 | Base atk of that character in character selection | 50 |
| selSpd(x) | integer |  | 3 | Base spd of that character in character selection | 20 |
| selType(x) | text |  | 1 | Type of that character in character selection | R |
| selWeak(x) | text |  | 1 | Weakness type of that character in character selection | P |
| selStr(x) | text |  | 1 | Strength type of that character in character selection | S |

## 

## IPO Diagram

|  |  |  |
| --- | --- | --- |
| **Input** | **Process** | **Output** |
| ActionBarHit | Compare the number to the hit range values to get hit multiplyer |  |
|  | Multiply hit multiplier with type multiplyer with base damage |  |
|  | Reduce enemy hit points by resulting value | Return damage value, current enemy hit points |
|  |  |  |
| Select Character 1 | Character 1 variables put in place |  |
| Select Character 2 | Character 2 variables put in place |  |
|  |  | Continue button becomes available |