Requirement Analysis Document (RAD)

Project: Plight of Pip: Wrath of the Riper

Team: Rascally Saints
Purpose of this Document

The purpose of this document is to clarify to the Project Team, Mentor, and overseeing Instructor what the requirements for this system are, as well as various constraints that the Project will be placed under. It formally explains what we will be producing as an end product.
# Table of Contents

**Purpose of this Document** 1

**Table of Contents** 2

**Introduction** 3

  - Scope of the system 3
    - What this project is: 3
    - What this project is not: 3
  - Project Deliverables: 4
  - Purpose of the System 4
  - Objective and Success Criteria for the Project 5
  - Definitions, acronyms, and abbreviations 5

**Proposed System** 6

  - Overview 6
  - Functional Requirements 6
  - Non-Functional Requirements 10
  - System Models 12
    - Use Cases 12

**Object Model** 25

  - Class Diagram 25

**Dynamic Model** 26

  - Sequence Diagrams 26
  - User Interface Mockups 36

*Rascally Saints* 2
Introduction

This project ambitiously seeks to develop a base demo for a video game, capable of being released to personal computer marketing platforms like Steam, and later console platforms like the Xbox, Playstation, and Switch’s online platforms. There is a supported market for side-scroller/top-down hybrid games, as proved by the success of established franchises like Legend of Zelda in their accredited title: A Link Between Worlds. This project seeks to capitalize on this and bring a new game to this once thought niche market. While the boom in the game industry has brought an oversaturation of products to the market, a truly unique product such as this has a true chance of success in the field.

Purpose of the System

The purpose of this system is simply put: Generate a unique environment for players to explore and enjoy. This is a purpose that is hard to define further, but research into the medium of video games have proven insightful in getting a better idea of how to achieve this goal. This experience will be generated by challenging the user in a variety of ways, both in terms of narrative storytelling and physical challenges posed within the system. Mastery of skills developed to achieve narrative goals set out within the system will provide the user a sense of accomplishment, feelings of joy, and entertainment.
Scope of the system

What this project is:

This project covers the development and implementation of a side-scroller/top-down dungeon crawler hybrid game. Included in this is the creation and gathering of art assets to use, development of main and side plot-lines, planning of level designs and maps, and the development of code that pertains to the inclusion and presentation of the previously stated items.

What this project is not:

This project covers almost every aspect of game design. The most notable exception is the creation of a game engine itself. This is provided by the Godot engine. The team plans to only add the implementations of the game and mechanics on top of the existing engine. Should it become necessary for performance, they will write their own scripts in C++ and recompile the engine, but Godot will remain the primary component of the engine.

Project Deliverables:

- Project Inception Report
- Requirement Specification
- Project Design Document
- Software Package Deliverable to Steam
Objective and Success Criteria for the Project

This project is a success when enough work has been produced to create adequate content for 10 ingame islands. Each island will host its own main questline, side quests, community, and other potential activities. There will also be dungeons that will provide the user with the main challenge of each island. Each island will be designed and built with an extensive amount of animations and sounds integrated into a cohesive experience. Islands will also feature unique gameplay mechanics. If done properly, a completed island will encourage the user to engage in continued gameplay.

Definitions, acronyms, and abbreviations

Dungeon: a labyrinthian environment with various enemies, obstacles, puzzles and treasures.

RPG: role-playing game

Side-Scroller: a style of game where the camera shows the world in profile.

Top-down: a style of game where the camera shows the world from a bird’s eye view.

Island: A level in the game.

NPC: non-player character.

Scene: an editable environment within Godot.

Actor: User
Proposed System

Overview

The environment is a hybrid gameplay which will require the user to engage in both top-down and side-scrolling views during different areas of gameplay. There will be a fully fleshed out story accompanying the game with a grand quest, side quests, in-depth story, unique game mechanics for different islands, interactive communities, innovative items/weapons system, unique species, and various side activities to give the game a role-playing feel.

Functional Requirements

FR1. Scene-to-Scene Transition

1.1. The system will be able to display a transition scene while the system loads the content for the incoming scene as the previous scene’s content is cleared from the active data.

1.2. System will activate the Island selection process when the user is in a selected area.

1.3. User will select next scene to load by scrolling down a list and selecting a location.

FR2. User Controls

2.1. User will be able to move up, down, left, and right.
2.2. User will be able to jump to higher locations.

2.3. User will be able to attack entities. This will cause the entities to lose health.

2.4. User will be able to interact with collectable objects. This will cause some objects to be added to the user’s inventory.

2.5. User will be able to interact with friendly NPCs, causing a dialogue box to appear on the screen.

2.6. User will be able to interact with event objects, causing a scripted event attached to the object to occur.

2.7. User will be able to launch their Player’s scarf towards a point of contact. The user will then be transitioned from their origin point to the point of contact.

2.8. User will be able to open the inventory screen with the press of a key.

2.9. User will be able to open the pause screen with the press of a key.

FR3. Items/Companions

3.1. User will have the ability to obtain NPC’s that will be added to the inventory to be utilized as items. Each NPC will add a new mechanic for the user to activate.

3.1.1. User will be able to unveil previously unavailable areas and content.

3.1.2. User will be able to add game objects to screen that can be walked upon to allow access to previously unavailable areas.

3.1.3. User will be able to latch onto climbable textured tiles to reach new areas.

3.1.4. User will be able throw item at a game object to interact with it as if it were the player.
3.1.5. User will be able to interact with a new event object that is a cuttable, destructible object.

3.1.6. User will be able to create a small lighting system in the scene to illuminate more of the scene.

3.1.7. User will be able to interact with new event tiles that are cracked destructible objects.

3.1.8. User will be able to add game objects to screen that connects two previously unconnected paths.

3.1.9. User will be able to throw object to hinder the hostile NPC’s capabilities.

3.2. User will be able use a health item in the inventory to gain twenty-five percent of their overall health back.

3.3. User will be able to use a hint item in the inventory, to gain a hint toward completing the challenge of the island. The hint will be displayed on the inventory screen.

3.4. User will be able to select items from inventory screen to set as active.

3.5. User will be able to use active items with one of two key presses. (Two active item keys)

4.1. NPCs will be able to move up, down, left, and right without user input.

4.2. Hostile NPCs will be able to attack the user and cause damage reflected in their health bar.
4.3. NPC will transition between the decision to move or to interact with the user through artificial intelligence.

FR5. Settings Management

5.1. User will be able to adjust the volume of the background music.

5.2. User will be able to adjust the volume of the sound effects.

5.3. User will be able to adjust the master volume.

5.4. User will be able to adjust the screen resolution.

FR6. Saving and Loading

6.1. User will be able to create new game file with three sub-save files.

6.2. User will be able to save data to the sub-save files.

6.3. User will be able to override old save data with new save data.

6.4. User will be able to load from a sub-save slot to load in previously saved data to continue gameplay.

FR7. Title Screen

7.1. User will be able to transition to game data creation scene by selecting new game.

7.2. User will be able to transition to load game data scene by selecting load game.

7.3. User will be able to transition to the options scene by selecting options.

FR8. Menu

8.1. User will be able to save file during gameplay from pause menu.

8.2. User will be able to load file during gameplay from pause menu.

8.3. User will be able to open in-game options menu during gameplay from pause menu.
8.4. User will be able to exit the current game playthrough from pause menu.

FR9. Views

9.1. System will transition between a top-down and side-scrolling view.

FR10. Sound System

10.1. System will play loaded background music when scene is loaded.

10.2. System will play sound clip attached to an event when event is triggered.

FR11. Animation

11.1. System will transition between a series of images to simulate animation.

FR12. Game End

12.1. System will include a timer that when it reaches zero will end the user’s current game play through.

12.2. System will be able to end user’s current game play through if player’s health falls to zero.

12.3. System will be able to end user’s current game play though when the player reaches the end of the main quest storyline.

Non-Functional Requirements

NFR1: Godot Development

1.1 Game will be developed using Godot as a development tool.

1.2 Game will be programmed in GDScript.

1.3 Godot will be used because it is free and open source. This helps with the cost and the ability to change what we need.
NFR2: Cohesive Audio

2.1 The individual music tracks and sound effects should sound similar enough to each other that it is obvious to a casual listener that they were created by the same person.

NFR3: Quality of Gameplay

3.1 The game must be playable and an enjoyable experience for the user. This will be measured by user feedback during BETA testing phase.

NFR4: Engine Performance

4.1 The Project Team will maintain quality of performance (measured in FPS) and ensure a general user’s machine will be able to utilize the Project.

4.1.1. General user’s machine will have approximately 6GB RAM and a 2.30GHz processor speed

NFR5: Documentation

5.1 The Project will be documented to support project knowledge and collaboration.

NFR6: Game Save Protection

6.1 Game save data will be encapsulated within internal game files to prevent user from manipulating the game in ways not intended by the developers.

NFR7: Accessibility of Controls

7.1 System will support standard game controllers.

NFR8: Object-Oriented Programming Approach to Design

8.1 The Project will use an Object-Oriented Programming (OOP) approach to designing components.
NFR9:  
Ten Island Structure

9.1  
The gameplay will be structured on ten distinct islands that vary in content, structure, and theme.
## System Models

### Use Cases

<table>
<thead>
<tr>
<th>Use Case Number</th>
<th>UC-01</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use Case Name</td>
<td>Sound Controls</td>
</tr>
<tr>
<td>Participating Actor</td>
<td>User</td>
</tr>
<tr>
<td>Entry Condition</td>
<td>User is within the Gameplay loop.</td>
</tr>
</tbody>
</table>
| Flow of Event   | 1. User presses the Pause key and is presented with the Pause Screen.  
|                 | 2. User uses direction keys to highlight options.  
|                 | 3. User presses the confirm key.  
|                 | 4. User uses direction keys to highlight the master volume option.  
|                 | 5. User uses the left and right direction keys to increase or decrease the slider and the master volume.  
|                 | 6. User presses the Cancel button to return to the Pause Menu.  
|                 | 7. User presses the Pause button to return to the Gameplay loop.  
| Alternative flow| 4.1 User uses direction keys to highlight the sound effects option.  
|                 | 5.1 User uses the left and right direction keys to increase or decrease the slider and the sound effects volumes.  
|                 | 6.1 Return to step 6.  
| Alternative flow| 4.1 User uses direction keys to highlight the background music option.  
|                 | 5.1 User uses the left and right direction keys to increase or decrease the slider and the background music volume.  
<p>|                 | 6.1 Return to step 6. |</p>
<table>
<thead>
<tr>
<th>Use Case Number</th>
<th>UC-02</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use Case Name</td>
<td>Screen Controls</td>
</tr>
<tr>
<td>Participating Actor</td>
<td>User</td>
</tr>
<tr>
<td>Entry Condition</td>
<td>User is within the Gameplay loop.</td>
</tr>
</tbody>
</table>

**Flow of Event**

1. User presses the Pause key and is presented with the Pause Screen.
2. User uses direction keys to highlight options.
3. User presses confirm key.
4. User uses direction keys to highlight full screen option.
5. User presses the Confirm key to toggle the checkbox activating fullscreen.
6. User presses the Cancel key to return to the Pause Menu.
7. User presses the Pause key to return to the Gameplay loop.

**Alternative Flow**

5.1 User presses the Confirm key to toggle the checkbox deactivating fullscreen.
6.1 Return to step 6.
## Use Case UC-03: Exiting Game

**Participating Actor:** User  
**Entry Condition:** User is within the Gameplay loop.

**Flow of Event:**
1. User presses the Pause key and is presented with the Pause Screen.  
2. User uses direction keys to highlight Exit option.  
3. User presses confirm key.  
4. User is prompted to ensure they want to exit.  
5. User selects No.  
6. User is taken back to the Pause Menu.

**Alternative Flow:**
5.1 User selects Yes.  
6. User is taken to the Main Menu.

## Use Case UC-04: Save Game

**Participating Actor:** User  
**Entry Condition:** User is within the Gameplay loop.

**Flow of Event:**
1. User presses the Pause key and is presented with the Pause Screen.  
2. User uses direction keys to highlight Save Game.  
3. User presses confirm key.  
4. User is prompted with three sub-save slots.  
5. User uses direction keys to highlight desired sub-save slot.  
6. User presses confirm key.  
7. System saves the game data to the game file and stores it.  
8. User returns to the Pause Menu.
<table>
<thead>
<tr>
<th>Use Case Number</th>
<th>UC-05</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use Case Name</td>
<td>Load Game (In-Game)</td>
</tr>
<tr>
<td>Participating Actor</td>
<td>User</td>
</tr>
<tr>
<td>Entry Condition</td>
<td>User is within the Gameplay loop.</td>
</tr>
</tbody>
</table>
| Flow of Event   | 1. User presses the Pause key and is presented with the Pause Screen.  
2. User uses direction keys to highlight load game.  
3. User presses the confirm key.  
4. User is prompted with three sub-save slots.  
5. User uses direction keys to highlight desired sub-save slot.  
6. User presses confirm key.  
7. System will load in selected data from file.  
8. System will load transition scene.  

*Rascally Saints 16*
<table>
<thead>
<tr>
<th>Use Case Number</th>
<th>UC-06</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use Case Name</td>
<td>Load Game (Main Menu)</td>
</tr>
<tr>
<td>Participating Actor</td>
<td>User</td>
</tr>
<tr>
<td>Entry Condition</td>
<td>User is at the Main Menu.</td>
</tr>
</tbody>
</table>
| Flow of Event   | 1. User uses direction keys to highlight Load Game.  
|                 | 2. User presses the confirm key.  
|                 | 3. User is taken to Game data scene.  
|                 | 4. User uses direction keys to highlight desired main save slot  
|                 | 5. User presses the confirm key causing three sub-save slots to appear.  
|                 | 6. User uses direction keys to highlight desired sub-save slot.  
|                 | 7. User presses the confirm key.  
|                 | 8. System loads the game data from file.  
|                 | 10. User enters the Gameplay loop. |

<table>
<thead>
<tr>
<th>Use Case Number</th>
<th>UC-07</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use Case Name</td>
<td>New Game (Main Menu)</td>
</tr>
<tr>
<td>Participating Actor</td>
<td>User</td>
</tr>
<tr>
<td>Entry Condition</td>
<td>User is at the Main Menu.</td>
</tr>
</tbody>
</table>
|                 | 2. User presses the confirm key.  
|                 | 3. User is taken to the Game data scene.  
|                 | 4. User uses direction keys to highlight desired main game slot.  
|                 | 5. System loads a fresh instance of the game.  
<p>|                 | 6. User enters the Gameplay loop. |</p>
<table>
<thead>
<tr>
<th>Use Case Number</th>
<th>UC-08</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use Case Name</td>
<td>Select an Island</td>
</tr>
<tr>
<td>Participating Actor</td>
<td>User</td>
</tr>
<tr>
<td>Entry Condition</td>
<td>User is within the Gameplay loop and on a selected area of the map.</td>
</tr>
</tbody>
</table>
|                 | 2. User is taken to the Map interface.  
|                 | 3. User uses direction keys to highlight desired island from the island list.  
|                 | 4. User presses the confirm key.  
|                 | 5. System Game state is changed, and the scene transition is activated. Proceed to UC-09.  
|                 | 6. User enters the Gameplay loop. |

<table>
<thead>
<tr>
<th>Use Case Number</th>
<th>UC-09</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use Case Name</td>
<td>Scene Transition</td>
</tr>
<tr>
<td>Participating Actor</td>
<td>User</td>
</tr>
<tr>
<td>Entry Condition</td>
<td>User is within the Gameplay loop and has triggered a scene transition event.</td>
</tr>
</tbody>
</table>
|                 | 2. System queries data for incoming scene data.  
|                 | 3. System loads in new scene data.  
<p>|                 | 4. User enters the gameplay loop. |</p>
<table>
<thead>
<tr>
<th>Use Case Number</th>
<th>UC-10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use Case Name</td>
<td>Player Action Listener</td>
</tr>
<tr>
<td>Participating Actor</td>
<td>User</td>
</tr>
<tr>
<td>Entry Condition</td>
<td>User is within the Gameplay loop.</td>
</tr>
</tbody>
</table>

**Flow of Event**

1. User presses a key mapped to either Jumping, Item use one, Item use two, Movement (Any Direction), Attack, Interaction, Pausing, Inventory, or their grappling scarf.
2. System will catch user input.
3. Player will perform the jumping action.
4. After resolving the input, the Player Action Listener will listen to any more user mapped buttons.

**Alternative Flow**

3.1 Item assigned to key will be activated.
4.1 Return to step 4.

**Alternative flow**

3.1 Player will be moved in direction assigned to the key pressed.
4.1 Return to step 4.

**Alternative flow**

3.1 System will add weapon game object to the scene checking its collision boundaries.
4.1 If it collides with another game object, it causes damage.
5.1 The weapon game object is removed from the scene.
6.1 Return to step 4.

**Alternative flow**

3.1 System will check if there is an interactable entity within range.
4.1 System will activate the entities scripted event.
5.1 Return to step 4.

**Alternative flow**

3.1 System will fire the player’s scarf and transition player if there is a valid tile in the path.
<table>
<thead>
<tr>
<th>Use Case Number</th>
<th>UC-11</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use Case Name</td>
<td>Hostile Non Player Character</td>
</tr>
<tr>
<td>Participating Actor</td>
<td>User</td>
</tr>
<tr>
<td>Entry Condition</td>
<td>User is within the Gameplay loop.</td>
</tr>
</tbody>
</table>
| Flow of Event   | 1. System gathers data about the scene. If the user is within range of the NPC it will activate the NPC’s attacking routine.  
2. System detects if there is a collision between the NPC and the player. If there is a collision, the system causes the player’s health to drop. |
<p>| Alternative Flow | 1.1 System gathers data about the scene. If the user is not within range of the NPC it will activate the NPC’s basic patrol routine. |</p>
<table>
<thead>
<tr>
<th>Use Case Number</th>
<th>UC-12</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use Case Name</td>
<td>Non-Hostile Non Player Character</td>
</tr>
<tr>
<td>Participating Actor</td>
<td>User</td>
</tr>
<tr>
<td>Entry Condition</td>
<td>User is within the Gameplay loop.</td>
</tr>
<tr>
<td>Flow of Event</td>
<td>1. System gathers data about the scene. If the user is interacting with the NPC, it will activate the interaction event.</td>
</tr>
<tr>
<td>Alternative Flow</td>
<td>1.1 System gathers data about the scene. If the user is not interacting with the NPC, it will activate the basic partol routine.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Use Case Number</th>
<th>UC-13</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use Case Name</td>
<td>Selecting an Item</td>
</tr>
<tr>
<td>Participating Actor</td>
<td>User</td>
</tr>
<tr>
<td>Entry Condition</td>
<td>User is within the Gameplay loop.</td>
</tr>
</tbody>
</table>
| Flow of Event   | 1. User presses the inventory button and is taken to the Inventory screen.  
2. User is placed in the inventory item area.  
3. User uses direction keys to transition to item selection area.  
4. User uses direction keys to highlight wanted item.  
5. User presses assignment key.  
6. System assigns item to the key.  
7. System closes the inventory screen.  
8. User is returned to the gameplay loop. |
### Use Case Number
UC-14

### Use Case Name
Using an inventory item

### Participating Actor
User

### Entry Condition
User is within the Gameplay loop.

### Flow of Event
1. User presses the inventory button and is taken to the Inventory screen.
2. User is placed in the inventory item area.
3. User uses direction keys to highlight wanted item.
4. User presses confirm key to activate item.
5. User’s health is increased by 25%.
6. User returns to step three until they are satisfied.
7. User presses the inventory key to return to gameplay loop.

### Alternative Flow
5.1 System displays a hint in the hint box.
6.1 Return to step 6.

---

### Use Case Number
UC-15

### Use Case Name
Animation

### Participating Actor

### Entry Condition
User is within the Gameplay loop.

### Flow of Event
2. System attaches these images to an entity.
3. System cycles through the images within the specified time limit.
4. System returns to the first image in the cycle when it reaches the end of a series.
<table>
<thead>
<tr>
<th>Use Case Number</th>
<th>UC-16</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use Case Name</td>
<td>Game over</td>
</tr>
<tr>
<td>Participating Actor</td>
<td></td>
</tr>
<tr>
<td>Entry Condition</td>
<td>User is within the Gameplay loop.</td>
</tr>
</tbody>
</table>
| Flow of Event   | 1. System loads the timer data.  
|                 | 2. System decrements the timer as long as game is running.  
|                 | 3. System displays end game scene if timer reaches 0.  
|                 | 4. User is taken back to the main menu screen. |
| Alternative Flow| 1.1 System loads the player health data.  
|                 | 2.1 System monitors health data.  
|                 | 3.1 System displays end game scene if player health reaches 0.  
|                 | 4.1 Return to step 4. |
Object Model
Rascally Saints 25
Dynamic Model

Sequence Diagrams
Screen Controls

User

Game Menu

Presses the Pause Button

Gameplay Loop is paused

Displays Pause Menu

Pause Menu

Selects Options from Pause Menu

Displays Options Menu

Options Menu

Queries whether the user is in fullscreen

Returns fullscreen mode settings

Screen & Sound Options are displayed

Loop

[Turn on Full Screen]

User checks the FullScreen checkbox

Requests fullscreen view

Requests full-screen view

Sets view to fullscreen

[Turn off Full Screen]

User unchecks the FullScreen checkbox

Requests windowed view

Requests windowed view

Sets view to realizable window

User leaves the Screen & Sound Options Menu

Returns control to Pause Menu

Presses the Pause Button

Returns control to Gameplay Loop

Gameplay Loop is unpaused

User

Game Menu

End

Game Engine

www.ansequence.com
**Player Action Listener**

1. **[Jump (Side Scrolling Only)]**
   - Presses the Jump Button
     - User's character is elevated

2. **[Items]**
   - Presses an Item Button
     - The selected item is utilized, performing its specific effect

3. **[Movement]**
   - Presses a Movement Button
     - The User character is moved in the direction of the movement button

4. **[Attack]**
   - Presses the Attack Button
     - The User character will swing their weapon in the direction they are currently facing

5. **[Interaction]**
   - Presses the Interaction Button
     - The User will attempt interact an object in front of them, triggering an event

6. **[Grappling Scarf]**
   - Presses the Scarf Button
     - The User character will launch their scarf in the direction they are facing
     - If the scarf collides with a valid point, the player will move from their origin point to the point of contact

---

Rascally Saints 32
Hostile Non Player Character

loop
- Query Environment data
  - Returns player location, map data, etc.

opt [NPC and User Collision]
- Sends damage to player's health

alt [User is in range]
- Attack state is activated
  - [User not in range (anymore)]
  - Resume patrol state

NPC (Non Player Character)
Game State
User

www.websequencediagrams.com
User Interface Mockups

Title Screen
Game Data Screen (Where user is taken after selecting save or load on main menu).

Game Data Screen 2 (If user is loading then after selecting a main file they will select from sub-file to load).
Inventory Screen (The other two screens are greyed out to show you are not in those areas).

Item Selection Screen
Inventory lore area

Pause Screen
Options Menu (Also where user comes if select options on Title screen).

Top-down Gameplay view
Side-scrolling Gameplay view
Island Selection Screen