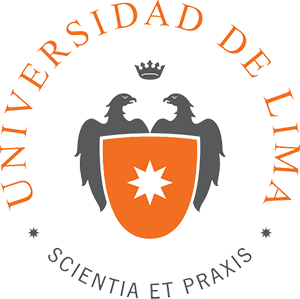
**Universidad de Lima**

**Carrera de Ingeniería de Sistemas**



**Proyecto de Ingeniería de Software 2:**

**“Abajo Los corruptos”**

**Sección: 801**

**Integrantes:**

* BISSO BERNAL, CARLOS MANUEL
* DONAYRE MEDINA, CARLO STEPHANO
* MELGAREJO PAUCAR, RICARDO ANTONIO

**Lima, Febrero de 2019**

**HISTORIAL DE CAMBIOS**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **FECHA** | **VERSIÓN** | **DESCRIPCIÓN** |  | **RESPONSABLE** | |  |  |
|  | **10-01-19** | 1.0 | Organización | del |  |  |  |  |
|  |  |  | Documento. |  | Bisso Melgarejo | |  |  |
|  | **17-01-19** | 1.1 | Sprites y Login base |  | Bisso | Melgarejo |  | Donayre |
|  |  |  |  |  |  | |  |  |
|  | **24-01-19** | 1.2 | Fondos base y Login prueba |  | Bisso | Melgarejo |  | Donayre |
|  |  |  |  |  |  | |  |  |
|  | **31-01-19** | 1.3 | Sprites incompleto |  | Bisso | Melgarejo |  | Donayre |
|  |  |  |  |  |  | |  |  |
|  | **04-02-19** | 1.4 | Fondos y Sprites completo | | Bisso | Melgarejo |  | Donayre |
|  | **07-02-19** | 1.5 | Sonidos y Efectos Base, Login mejora | | Bisso | Melgarejo |  | Donayre |
|  |  |  |  |  |  | |  |  |
|  | **11-02-19** | 1.6 | Login actualizado, Fondos, Sprites, Efectos y Sonido completos, y Documentación Parcial | | Bisso | Melgarejo |  | Donayre |

**Tabla 1: Historial de cambios**

**TABLA DE CONTENIDO**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| HISTORIAL DE CAMBIOS ................................................................................................................................. | | | 2 | | 2 |  | |  |  |
| TABLA DE CONTENIDO ................................................................................................................................. | | | 3 | |  |  | |  |  |
| ÍNDICE DE TABLAS ................................................................................................................................. | | | 5 | |  |  | |  |  |
| 1 | INTRODUCCIÓN .............................................................................................................................. | | 6 | |  |  | |  |  |
|  | 1.1. | Objetivos ..................................................................................................................... | 6 | |  |  | |  |  |
|  | 1.2. | Estrategia de Pruebas  ..................................................................................................................... | 6 | |  |  | |  |  |
|  | 1.3. | Alcance  ..................................................................................................................... | 6 | |  |  | |  |  |
|  | 1.4. | Referencias ..................................................................................................................... | 7 | |  |  | |  |  |
|  | 1.5. | Definiciones, abreviaciones y acrónimos ..................................................................................................................... | 8 | |  |  | |  |  |
| 2 | ARTEFACTOS DE PRUEBA  ............................................................................................................................. | | 9 |  |  | |  | |  |
|  | 2.1 | Módulos del Programa  .................................................................................................................... | 9 | |  |  | |  |  |
|  | 2.2 | Procedimientos de Usuario  .................................................................................................................... | 10 | |  |  | |  |  |
| 3 | CARACTERÍSTICAS A SER PROBADAS  .............................................................................................................................. | | 11 |  |  | |  | |  |
| 4 | CARACTERÍSTICAS QUE NO SERÁN PROBADAS  .............................................................................................................................. | | 13 |  |  | |  | |  |
| 5 | APROXIMACION  .............................................................................................................................. | | 14 |  |  | |  | |  |
|  | 5.1 | Pruebas Unitarias …………...................................................................................................... | 14 | |  |  | |  |  |
|  | 5.2 | Pruebas de Frontera  .................................................................................................................... | 14 | |  |  | |  |  |
|  | 5.3 | Pruebas de Integración  ..................................................................................................................... | 15 | |  |  | |  |  |
|  | 5.4 | Pruebas de Sistema ..................................................................................................................... | 15 | |  |  | |  |  |
| 6 | PROCESO DE PRUEBAS  .............................................................................................................................. | | 17 |  |  | |  | |  |
| 7 | CASOS DE PRUEBA DEL JUEGO  .............................................................................................................................. | | 24 |  |  | |  | |  |
|  |  | |  |  |  | |  | |  |

**ÍNDICE DE TABLAS**

TABLA 1: HISTORIAL DE CAMBIOS 2

TABLA 2: DEFINICIONES, ACRÓNIMOS Y ABREVIACIONES 9

TABLA 3: MÓDULOS A PROBAR EN EL SISTEMA 11

TABLA 4. CARACTERÍSTICAS A SER PROBADAS 13

TABLA 5. CARACTERÍSTICAS QUE NO SERÁN PROBADAS. 14

TABLA 6. PRUEBAS UNITARIAS. 15

TABLA 7. PRUEBAS DE FRONTERA 15

TABLA 8. ENTREGABLE DE LA PRUEBA. 16

TABLA 9. PRUEBA DE INTEGRACIÓN. 16

TABLA 10. RESULTADOS PRUEBA INTEGRACIÓN 16

TABLA 11. PRUEBAS DE SISTEMA 16

TABLA 12. RESULTADOS PRUEBAS DE SISTEMA. 17

TABLA 13. CASO DE PRUEBA 1 18

TABLA 14. CASO DE PRUEBA 2 19

TABLA 15. CASO DE PRUEBA 3 20

TABLA 16. CASO DE PRUEBA 4 20

TABLA 17. CASO DE PRUEBA 5 21

TABLA 18. CASO DE PRUEBA 6 21

TABLA 19. CASO DE PRUEBA 7 21

TABLA 20. CASO DE PRUEBA 8 22

TABLA 21. CASO DE PRUEBA 9 22

TABLA 22. CASO DE PRUEBA 10 23

TABLA 23. CASO DE PRUEBA 11 24

TABLA 24. CASO DE PRUEBA 12 25

TABLA 25. CASO DE PRUEBA 13 26

TABLA 26. CASO DE PRUEBA 14 27

TABLA 26. CASO DE PRUEBA 14 27

**1 INTRODUCCIÓN**

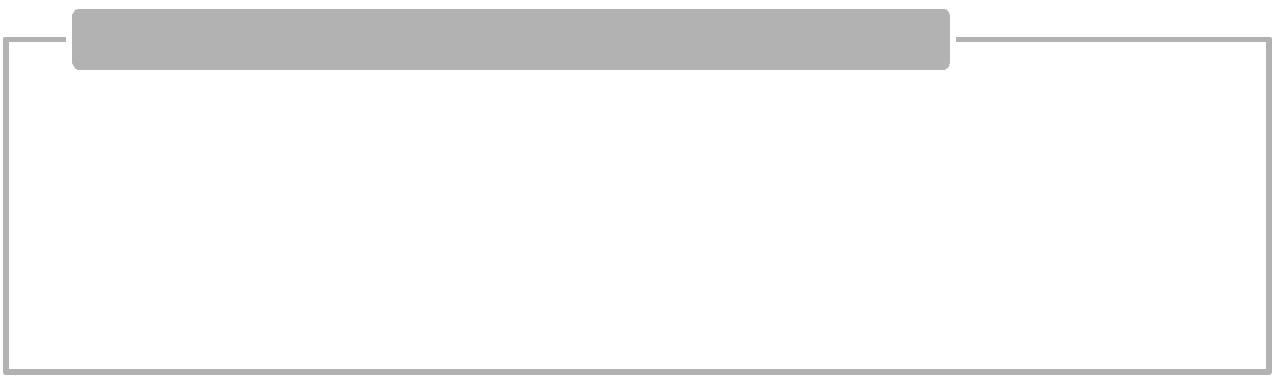
**1.1. Objetivos**

El plan de pruebas de Software se elabora con el fin de especificar qué elementos o componentes se van a probar para que el equipo de trabajo pueda realizar el proceso de Validación y Verificación de los requerimientos funcionales y no funcionales que se necesiten para crear el videojuego “Abajo Los Corruptos”. Además, a través del plan de pruebas se puede continuar con la trazabilidad de los requerimientos, con lo cual el grupo, identifica el porcentaje de avance que se ha logrado por periodos.

Al desarrollar el plan de pruebas, se puede obtener información sobre los errores, defectos o fallas que tiene el prototipo, así se realizan las correcciones pertinentes, según el caso y se asegura la calidad del videojuego que se realizará en el curso. El plan de pruebas se aplica sobre el videojuego, es decir, todo el código necesario para su implementación, los resultados de las pruebas son registrados en un formato que se encuentra en el Anexo 1: Reportes de Pruebas. Las pruebas a implementar son básicas, y esto incluye las pruebas unitarias y de integración que son vitales para la validación del producto.

**1.2. Estrategia de Pruebas**

A través de los diferentes documentos que se han realizado, se pretende retomar información directamente relacionada con las pruebas, para asegurar la calidad en general. Además, le permite al responsable de las pruebas saber exactamente los criterios que se deben tener en cuenta para probar cada elemento del sistema. A continuación, se explica con mayor detalle, el aporte de cada documento con respecto al plan de pruebas.



•Priorización: se escogen los requerimientos de mayor priorización para poder aplicar las pruebas correspondientes.

•Grafo de dependencias: permite una visualización clara sobre los diferentes grupos de requerimientos que deben ser evaluados dentro de las pruebas de integración.

•Trazabilidad: le permite al responsable de la prueba saber el estado de requerimiento.

SRS



•Diagrama de Componentes: La separación del sistema por componentes permite la clasificación de las pruebas según la funcionalidad del módulo.

•Diagrama de CU: Permite una mejor visualización de los diferentes escenarios para realizar las pruebas de sistema.

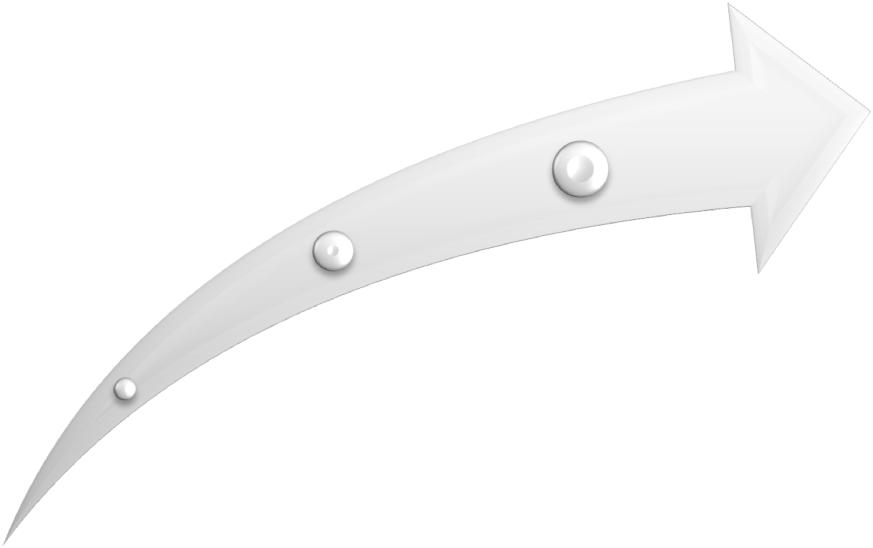
•Vista Física: Prueba de los componentes de Hardware que tiene la aplicación.

**Ilustración 1: Estrategia del plan de pruebas**

Con esta estrategia se asegura llevar el seguimiento de la trazabilidad basada en los requerimientos del documento SRS.

**1.3.** **Alcance**

Teniendo en cuenta los documentos hechos anteriormente, se pretende realizar las pruebas, de manera incremental, por módulo. Para una mejor comprensión, ver la Ilustración 2: Alcance del plan de pruebas, la cual muestra el alcance y el orden en que se realizarán.



Pruebas de

Sistema

Pruebas de

Integración

Pruebas

unitarias

(Pruebas

Frontera)

**Ilustración 2: Alcance del plan de pruebas**

Las pruebas unitarias se realizarán inmediatamente después de haber implementado la aplicación, esto quiere decir que, el orden corresponde al descrito en la prioridad de requerimientos y por grafo de dependencias construidos desde el documento SRS.

**1.5. Definiciones, abreviaciones y acrónimos**

|  |  |  |
| --- | --- | --- |
|  | **CONCEPTO** | **DESCRIPCIÓN** |
|  | AS | Arquitectura de Software |
|  | ANS | Asignación Numérica Simple (Método de Priorización) |
|  | ERMT | Hace referencia a las iniciales del Nombre designado para la |
|  |  | Herramienta: Easy Requirement Management Tool. |
|  | IS | Ingeniería de Software |
|  | STP | Software Tester Plan |

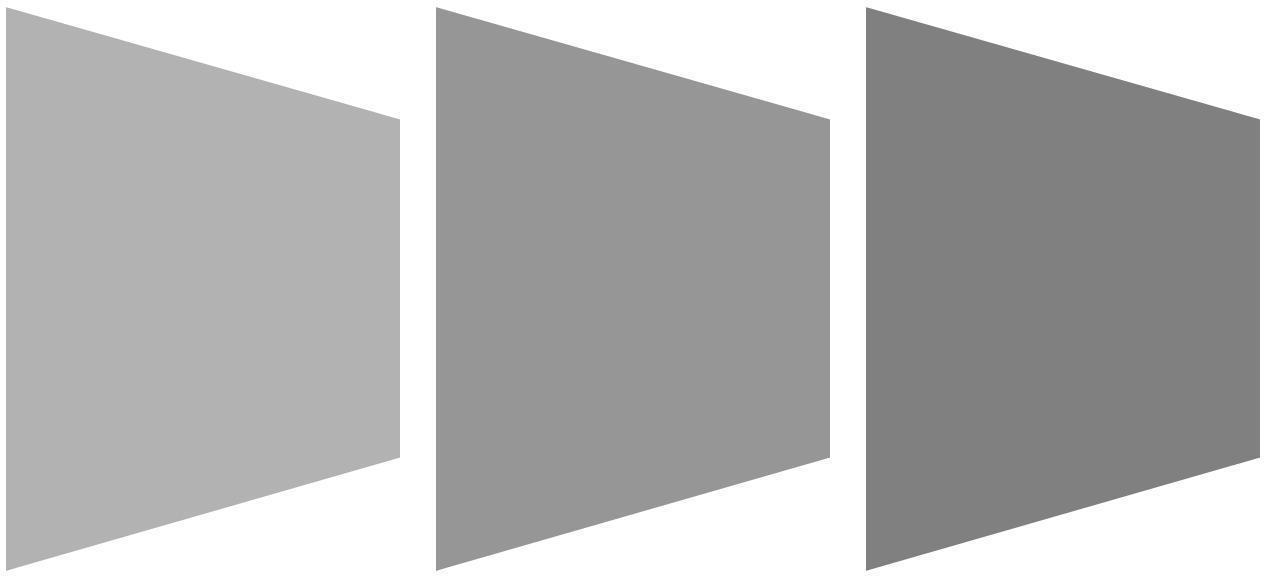
Método Wiegers Método de Priorización de Wiegers

**Tabla 2: Definiciones, acrónimos y abreviaciones**

**2 ARTEFACTOS DE PRUEBA**

**2.1 Módulos del Programa**

En esta sección se muestran los módulos que se pretenden probar, además de las especificaciones de las pruebas a realizar en cada uno. Cada módulo representa un componente del sistema.



|  |  |  |
| --- | --- | --- |
| Módulo | Pruebas | Descripción |
| •Es un conjunto | •Tipo de pruebas | •Breve |
| de elementos, | que serán | explicación |
| el cual tienen | utilizadas | sobre la prueba |
| en común la | dentro del | a realizar. |
| finalidad. | componente. |  |

**Ilustración 3: Módulos del programa**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | **Modulo** | **Pruebas** | **Descripción** |
|  |  |  |  |  |
|  |  | **GUI** | Facilidad de uso | La facilidad de uso consiste en que siempre tengan el conocimiento sobre qué hacer desde que se inicia el juego hasta que este finalice, como también saber qué y cómo hacerlo. |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  | Look & Feel | Es el diseño y arte que brinda la apariencia que se proporciona al usuario. |
|  |  |  |  |  |
|  |  | **Lógicas** |  |  |
|  |  | Funcionalidad | El sistema debe poder realizar todos los requerimientos establecidos con el cliente. Este módulo será guiado por los diferentes tipos de requerimientos que se han manejado durante el proyecto. |
|  |  |  |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | **DAO** |  |  |  |
|  |  |  | Persistencia | El sistema debe ser capaz de guardar datos, como al realizar el “Login” de los jugadores y los puntajes que se obtengan durante este para ser usados en otro momento. Además, de tener acceso a ellos sin tener ningún inconveniente de consistencia e integridad. |
|  |  |  | No funcionales | El sistema debe cumplir con los requerimientos no funcionales que se han especificado en el SRS teniendo en cuenta el diseño. |

**Tabla 3: Módulos a probar en el sistema**

**2.2 Procedimientos de Usuario**

Para utilizar la herramienta de manera adecuada se necesitan guías o manuales que sean claros, correctos, completos y coherentes, para que el usuario pueda interactuar con el juego de forma correcta y pueda comprender los conceptos tras la funcionalidad. A continuación, se muestran los diferentes atributos de calidad de estos procedimientos:

**Clara**: las instrucciones proporcionadas en el documento, deben ser lo suficientementeexplicitas para que el usuario pueda desenvolverse dentro del juego

**Correcta**: No existen errores en general en el producto final que se le entrega al jugador

**Completa**: la información debe estar completa, desde la parte técnica hasta la partefuncional.

**Coherente**: no existen ambigüedades, ni incongruencias en el documento ni durante el juego.

**3 CARACTERISTICAS A SER PROBADAS**

En esta sección se encuentran las características de la herramienta a ser probadas con un caso de estudio específico.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | **Característica** |  |  | **Descripción** | |  |  |  | **Módulo** |
|  |  |  | |  |  |  |  |  |  |  |
|  |  | **Requerimientos Funcionales** | |  | Se debe | tener | en | cuenta | el | Los módulos donde se puede |
|  |  |  |  |  | criterio | de | aceptación | | y | probar esta característica son: |
|  |  |  |  |  | dependencias, | | para realizar | | | Lógicas |
|  |  |  |  |  | pruebas | en | los | módulos. | |  |
|  |  |  |  |  | Además, se debe utilizar el | | | | |  |
|  |  |  |  |  | documento de casos de uso | | | | | DAO |
|  |  |  |  |  | para tener claro los casos de | | | | |  |
|  |  |  |  |  | éxito y fallo, y si el juego | | | | |  |
|  |  |  |  |  | cumple con esto. | | | | |  |
|  |  | **Requerimientos NO** |  |  |  |  |  |  |  |  |
|  |  |  | Se debe | tener | en | cuenta | el | El módulo donde se puede |
|  |  | **Funcionales** |  |  | criterio de aceptación y lo que | | | | | probar esta característica es:  NO Funcionales. |
|  |  |  |  |  | exige el requerimiento para su | | | | |  |
|  |  |  |  |  | cumplimiento, especificado en | | | | |  |
|  |  |  |  |  | el documento SRS. | | |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |

**Tabla 4. Características a ser Probadas**

**4 CARACTERÍSTICAS QUE NO SERAN PROBADAS**

En esta sección se encuentran las características del juego a ser probadas con un caso de estudio específico.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | **Característica** | | **Descripción** | |  |  | **Módulo** |
|  |  | **Procedimientos de** |  | No serán probados, debido a que no se | | | | Procedimientos de Usuario |
|  |  | **Usuario** |  | cuenta | con | el tiempo suficiente para | |  |
|  |  |  |  | probar | los | diferente | criterios, |  |
|  |  |  |  | relacionados con los procedimientos, ya | | | |  |
|  |  |  |  | que se necesitan demasiadas horas de juego para poder encontrar todos los posibles Bugs del juego. | | | |  |
|  |  |  |  |  | | | |  |
|  |  |  |  |  | | | |  |
|  |  | **RED NO LOCAL** |  |  | | |  |  |
|  |  |  | Debido a que el tiempo es insuficiente, y | | | | El módulo donde se puede |
|  |  |  |  | no está dentro del alcance del proyecto, | | | | probar esta característica son: |
|  |  |  |  | las pruebas de éste módulo no se | | | | RED |
|  |  |  |  | realizaran. | |  |  |  |

**Tabla 5. Características que no serán probadas.**

**5 APROXIMACION**

En esta sección se exponen los tipos de pruebas a utilizar para el juego, cada una de ellas presenta un formato, el cual se va registrar los resultados.

**5.1 Pruebas Unitarias**

Pruebas por cada unidad, en este caso una unidad es equivalente a un requerimiento. El requerimiento es aprobado y aprobado si este cumple con lo que está escrito en la especificación de requerimientos.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| NOMBRE | Pruebas Unitarias | | ID: | UT01 |
| ACTIVIDADES |  | Análisis de requerimientos del sistema | | |
| TIEMPO ESTIMADO | | 30 – 60 minutos por unidad | | |
| MÉTODOS O |  | Python, VsCode , librerías |  |  |
| HERRAMIENTAS | |  |  |  |
| ENTREGABLE |  | Lista de chequeo sobre el cumplimiento del requerimiento, ¿realiza lo que se describe? | | |
|  |  |  | | |
|  |  |  | **Tabla 6. Pruebas Unitarias.** | |
| **5.2 Pruebas de Frontera** | | |  |  |

Pruebas fronteras, son las que toman en cuenta valores límite, para verificar el comportamiento del juego en esos casos.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| NOMBRE | |  | Pruebas de Frontera | | |  | ID: | | |  | LT01 | |
|  |  |  |
| ACTIVIDADES | | |  | Se realizaran los distintas pruebas con los valores límites y mínimos de usuarios que pueda soportar el juego en red local. | | | | | | | | |
|  |  |  |  |  | |  |  |  |  |  |  |  |
| TIEMPO ESTIMADO | | | | 20 minutos por prueba | |  |  |  |  |  |  |  |
| MÉTODOS | | | O | VsCode, CMD, Python, Git | |  |  |  |  |  |  |  |
| HERRAMIENTAS | | | |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  | **Tabla 7. Pruebas de Frontera** | | |  |  |  |  |  |
|  |  |  |  |  |  | | |  |  |  |  |  |
| Respecto a los entregables, para la prueba se llenara la siguiente tabla: | | | | | | | | | |  |  |  |
|  |  | | |  |  |  | |  |  | | |  |
|  | **Nombre** | | |  |  | **Identificador** | | **T01** | | | |  |
|  | **Valor máximo** | | |  |  | **Valor mínimo** | |  |  |  |  |  |
|  | **Resultado esperado** | | | | |  |  |  |  |  |  |  |
|  | **Resultados obtenidos** | | | | |  |  |  |  |  |  |  |
|  | **Estado** | | |  |  | Funciona: | |  |  | No funciona: | |  |

**Comentarios**

**Tabla 8. Entregable de la prueba.**

**5.3 Pruebas de Integración**

Las pruebas de integración, como su nombre lo indica, son pruebas hechas a un conjunto de requerimientos, en este caso se distribuyen en los tipos de requerimientos que se definieron en el documento SRS.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| NOMBRE |  | Pruebas de Integración | | |  | IDENTIFICADOR | IT01 |
|  |  |
| ACTIVIDADES | |  |  | Validación de requerimientos | |  |  |
|  | | |  |  | |  |  |
| TIEMPO ESTIMADO | | |  | 15 minutos por pruebas | |  |  |
| MÉTODOS | |  | O | Git | |  |  |
| HERRAMIENTAS | | |  |  |  |  |  |
| ENTREGABLES | |  |  | Informe generado en donde se indica si se tiene un correcto funcionamiento o | | | |
|  |  |  |  | no. | |  |  |

**Tabla 9. Prueba de Integración.**

|  |  |  |
| --- | --- | --- |
| **ID** | **GRUPO DE REQUERIMIENTOS** | **RESULTADOS DE LA PRUEBA** |
|
|  | Grupo de requerimientos que | Resultado de la prueba. |
|  | están relacionados dentro del |  |
|  | grafo de dependencias |  |
|  | **Tabla 10.Resultados** | **Prueba Integración** |

**5.4 Pruebas de Sistema**

Las pruebas de sistema son pruebas realizadas al juego como un conjunto, que casos de uso cumple en su totalidad, con rutas de éxito y fallo, que han sido definidas en el documento de Casos de Uso.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| NOMBRE | Pruebas sistemas | | IDENTIFICADOR | ST01 |
| ACTIVIDADES |  | Se realizaran pruebas funcionales y No funcionales | | |
| TIEMPO ESTIMADO | | 15 minutos por prueba |  |  |
| MÉTODOS |  | O Git |  |  |
| HERRAMIENTAS | |  |  |  |
| ENTREGABLES |  | Informe generado por el responsable de esta prueba el cual informara si se | | |
|  |  | tiene un correcto funcionamiento o no. | | |

**Tabla 11. Pruebas de Sistema**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **ID** | **CASO DE USO** | **RESULTADOS DE LA PRUEBA** |
|  |  | Grupo de requerimientos que | Resultado de la prueba. |
|  |  | están relacionados dentro del |  |
|  |  | grafo de dependencias |  |

**Tabla 12. Resultados Pruebas de Sistema.**

**6 PROCESO DE PRUEBAS**

En esta sección se presentan los casos de pruebas generales para usarlos con el juego. Cada cuadro está asociado a un caso de Uso, desde ahí se desglosa en los diferentes módulos involucrados para el funcionamiento y se evalúa el resultado obtenido. En las siguientes tablas, se muestran los casos de pruebas a realizar:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **NOMBRE** | **Actualizar Control de Cambios** | | | |  | PRUEBAS | **P1** |
| **PROPÓSITO** | Verificar si al hacer un cambio en los | | | | requerimientos, | | se pide el registro de |
|  | cambios y actualiza el historial de cambios en la base de datos. | | | | | | |
| **PRERREQUISITOS** |  | Haber solicitado manejar historial de cambios | | | | | |
|  |  | Tener requerimientos en la base de datos | | | | |  |
|  |  | Realizar cambios significativos | | | | |  |
|  |  | o | Atributos del requerimientos | | | |  |
|  |  | o | Relaciones del requerimiento | | | |  |
|  |  | o | Eliminación del requerimiento | | | |  |
| **UBICACIÓN** | Base de datos MySQL | | | | | |  |
| **ENTRADA** |  | Creación o Modificación de algún requerimiento | | | | | |
|  |  | Datos solicitados del historial | | | |  |  |
|  |  | o | Responsable | | |  |  |
|  |  | o | Justificación | | |  |  |
| **ORÁCULO** | El historial de cambios ha sido actualizado | | | | | |  |
| **PASOS** | 1. | Visitar las Relaciones de Logins guardados. | | | | | |
|  | 2. | Realizar uno o más cambios en los requerimientos( como borrar). | | | | | |
|  | 3. | Ingresar el id del jugador, es decir el Login o nombre de usuario | | | |  |  |
|  | 4. | Clic en “borrar usuario”. | | | | | |
|  |  |  | |  |  |  |  |
|  | 5. | Clic en“Aceptar”. | |  |  |  |  |
|  |  |  | | | |  |  |
| **Módulos Asociados** |  | Atributos y clasificación | | | |  |  |
|  |  | Historial de cambio | | | |  |  |
|  |  | Grafo y reportes (relaciones) | | | |  |  |
|  |  | |  |  | |  |  |
| NOMBRE | **Actualizar** | | **relaciones** | **entre** |  | PRUEBAS | **P2** |
|  | **requerimientos en la** | | | **Base de** |  |  |  |
|  | **datos** |  |  |  |  |  |  |
| PROPÓSITO | Verificar que si es posible crear, eliminar y modificar relaciones entre los jugadores | | | | | | |
|  |  | | |  |  |  |  |
| PRERREQUISITOS |  | Existen requerimientos en la base de datos. | | | | | |
| UBICACIÓN | Base de datos MySQL, | | | | | | |
|  | creación y modificación-eliminación. | | | | |  |  |
| ENTRADA |  | Seleccionar un requerimiento | | | |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | Selección de la acción que desea hacer | |
|  |  |  | o | Crear |
|  |  |  | o | Modificar |
|  |  |  | o | Eliminar |
|  |  | Selección del segundo requerimiento | |
| ORÁCULO |  | Cambio en el atributo “Requerimientos asociados” del requerimiento(s) | | |
|  |  | involucrados en la relación | | |
|  |  |  | (PARA CREAR) La base de datos se encuentra actualizada con la | |
|  |  |  | nueva relación, entre un par de requerimientos | |
|  |  | (PARA MODIFICAR)La base de datos se encuentra actualizada con | |
|  |  |  | la eliminación de una relación, entre un par de requerimientos | |
|  |  | (PARA ELIMINAR)La base de datos se encuentra actualizada con el | |
|  |  |  | cambios de relación, entre un par de requerimientos | |
| PASOS |  | **PARA CREAR** | |  |
|  | 1. | | Visitar la pantalla de modificación | |
|  | 2. | | Seleccionar un requerimiento. | |
|  | 3. | | Seleccionar “crear”. | |
|  | 4. | | Seleccionar la acción | |
|  | 5. | | Seleccionar el segundo requerimiento. | |
|  | 6. | | Clic en crear. | |
|  |  | **PARA MODIFICAR** | | |
|  | 1. | | Visitar la pantalla de relaciones. | |
|  | 2. | | Seleccionar un requerimiento. | |
|  | 3. | | Seleccionar el segundo requerimiento. | |
|  | 4. | | Clic en “modificar”. | |
|  |  | **PARA ELIMINAR** | | |
|  | 1. | | Visitar la pantalla de jugadores. | |
|  | 2. | | Seleccionar un requerimiento. | |
|  | 3. | | Seleccionar el segundo requerimiento. | |
|  | 4. | | Clic en “eliminar”. | |
|  |  |  |  | |
| **Módulos Asociados** |  |  | Grafo y reportes (relaciones) | |
|  |  |  |  |  |

**Tabla 14: Caso de Prueba 2**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| NOMBRE | **Actualizar requerimiento** | |  | | |  | PRUEBAS | | **P3** |  |  |  |  |
|  |  | | | | | |  | |  | | | | |
| PROPÓSITO | Verificar el cambio en los atributos | | | | | | del requerimiento, cuando el usuario | | | | | | |
|  | lo realice | |  | |  |  |  |  |  |  |  |  |  |
| PRERREQUISITOS | El proyecto debe tener atributos asociados para la plantilla de | | | | | | | | | | | | |
|  |  | especificación |  | |  |  |  |  |  |  |  |  |  |
|  | El requerimiento | debe | | | estar | | clasificado | | en | un | tipo | de |
|  |  | requerimiento |  | |  |  |  |  |  |  |  |  |  |
|  | El requerimiento debe estar creado en la base de datos | | | | | | | | | | |  |  |
| UBICACIÓN | Base de datos MySql | | | | | | |  |  |  |  |  |  |
| ENTRADA |  | Modificación |  | | Uno | o más | | atributos | | de | uno | o | mas |
|  |  | requerimientos |  | | |  |  |  |  |  |  |  |  |
| ORÁCULO | El requerimiento ha | | | sido | modificado exitosamente. Ahora | | | | | | | se puede | |

|  |  |  |
| --- | --- | --- |
|  | observar los nuevos valores modificados | |
| PASOS | 1. | Visitar la página BD |
|  | 2. | Modificar información de los atributos del requerimiento. |
|  | 3. | Clic en modificar. |
| **Módulos Asociados** |  | Atributos y clasificación |
|  |  | Localización y Trazabilidad |

**Tabla 15: Caso de Prueba 3**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| NOMBRE | **Actualizar listas de jugadores** | |  | PRUEBAS | |  | **P4** | |
| PROPÓSITO | Verificar que al registrar | | un cambio | en las los | criterios de la lista de | | |
|  | verificación y validación, la base de datos puede actualizar sus valores en | | | | | | |
|  | cada requerimiento o el proyecto, dependiendo del tipo de lista que este | | | | | | |
|  | manejando. | |  |  | |  |  | |
| PRERREQUISITOS |  | Debe haber requerimientos en la base de datos. | | | | | |
|  |  | Debe haber seleccionado una lista de verificación y validación | | | | | |
| UBICACIÓN | Base de datos de MySQL y Pantalla de VerificarValidar | | | | | | |
| ENTRADA | **PARA VALIDACIÓN EN REQUERIMIENTOS** | | | | |  |  | |
|  | 1. | Selección o no selección del criterio de la lista de jugadores de uno o más requerimientos | | | | | |
|  |  |  | | | |  |  | |
|  | **PARA VALIDACIÓN EN EL PROYECTO** | | | | |  |  | |
|  | 2. | Selección o no selección del criterio de la lista de jugadores | | | | | |
| ORÁCULO | Los criterios de la lista de jugadores ha sido actualizada en la base de datos. | | | | | | |
| PASOS | **PARA VALIDAR EN REQUERIMIENTOS** | | | | |  |  | |
|  | 1. | Visitar la página de VerificarValidar. | | | |  |  | |
|  | 2. | Seleccione la lista requerida. | | | |  |  | |
|  | 3. | Seleccionar el requerimiento a Verificar. | | | |  |  | |
|  | 4. | Modificar los atributos | | | | | |
|  |  | . | | | |  |  | |
|  | 5. | Clic en “Modificar”. | | | |  |  | |
|  | **PARA VALIDAR EN PROYECTO** | |  |  | |  |  | |
|  | 1. | Visitar la página de VerificarValidar. | | | |  |  | |
|  | 2. | Seleccione la lista que desea diligenciar. | | | |  |  | |
|  | 3. | Modificar los “Check” de la lista de validación en los criterios definidos. | | | | | |
|  |  | para esa lista de jugadores. | | | |  |  | |
|  | 4. | Clic en “Modificar”. | | | |  |  | |
| **Módulos Asociados** |  | ListaJugadores |  |  | |  |  | |
|  |  |  |  |  | |  |  | |
|  |  | **Tabla 16: Caso de Prueba 4** | | | |  |  | |
|  |  | |  |  | |  |  | |
| NOMBRE | **Actualización Priorización** | |  | PRUEBAS | |  | **P5** | |
| PROPÓSITO | Verificar que en la base | | de datos | se han actualizado los valores de | | | |
|  | priorización, modificados por el usuario. | | | | |  |  | |
| PRERREQUISITOS |  | Debe haber requerimientos en la base de datos. | | | | | |
|  | 3. | Debe haber seleccionado un método de Priorización | | | | | |
| UBICACIÓN | Base de datos MySQL | |  |  | |  |  | |
| ENTRADA | Modificación de valores | |  |  | |  |  | |
| ORÁCULO | La base de datos se encuentra actualizada | | | | | con los nuevos valores | |
|  | modificados y el atributo prioridad también se encuentra actualizado. | | | | | | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| PASOS | 1. | Visitar la página de Priorización. | | | | |  |
|  | 2. | Modificación de valores de priorización (dentro de los límites, según la “Frontera”) | | | | | |
|  |  |  | | | | | |
|  | 3. | Hacer clic en “Modificar”. | | |  |  |  |
|  |  |  | |  |  |  |  |
| **Módulos Asociados** |  | Priorización | | |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  | **Tabla 17: Caso de Prueba 5** | |  |  |  |
|  |  | | |  |  |  |  |

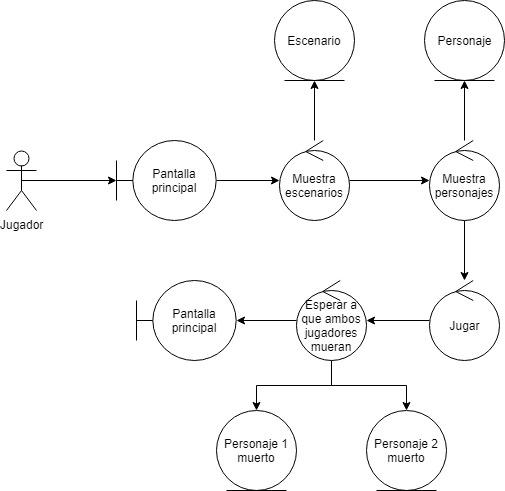
|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | |  | | | | |  |
| NOMBRE | **METODOS GENERALES** | | |  | PRUEBAS | |  | **P6** |
| PROPÓSITO | Verificar que el comportamiento de | | | cada uno de los métodos generales de | | | | |
|  | la lógica del negocio. | |  | | | |  |  |
| PRERREQUISITOS |  |  | |  | |  |  |  |
| UBICACIÓN | Base de datos MySQL | |  | | | |  |  |
| ENTRADA | Información requerida en cada uno de los métodos | | | | | | |  |
| ORÁCULO | Verificación de los datos en la base de datos o en las excepciones | | | | | | | |
|  | capturadas | |  | | | |  |  |
| PASOS | 1. Construr una clase para probar cada uno de los métodos | | | | | | | |
|  | 2. Registrarlos en el archivo de “Reporte de Pruebas”. | | | | | | | |
| **Módulos Asociados** | TODOS |  | |  | |  |  |  |
|  |  |  | |  | |  |  |  |

**7 Casos de Prueba del juego**

|  |  |
| --- | --- |
| **Formato de Caso de Prueba** | |
| **Identificador** | *CP\_01* |
| **Objetivo del Caso de Prueba** | Validar que el “login” sea exitoso. |
| **Caso de Prueba** | **Criterios de Aceptación** |
| 1. Ingresar nombre de usuario 2. Dar click en el botón “Login” | El sistema pasa a la siguiente fase de selección de personajes. |
| **Identificador** | *CP\_02* |
| **Objetivo del Caso de Prueba** | Verificar “login” incorrecto. |
| **Caso de Prueba** | **Criterios de Aceptación** |
| 1. Ingresar nombre de usuario 2. Comprobar que el nombre de usuario ya se haya utilizado anteriormente | El sistema manda un mensaje de “login incorrecto”. |
| **Identificador** | *CP\_03* |
| **Objetivo del Caso de Prueba** | Selección de personaje correcto. |
| **Caso de Prueba** | **Criterios de Aceptación** |
| 1. Ingresar nombre de usuario 2. Acceder a la pantalla de selección de personajes 3. Seleccionar el botón “Ready” | El sistema ha seleccionado el mismo personaje elegido por el usuario para su posterior uso durante el juego. |
| **Identificador** | *CP\_04* |
| **Objetivo del Caso de Prueba** | Validar que se habilite correctamente la opción de seleccionar un personaje al azar cuando se dé click en el botón “Random”. |
| **Caso de Prueba** | **Criterios de Aceptación** |
| 1. Ingresar nombre de usuario 2. Acceder a la pantalla de selección de personajes 3. Seleccionar el botón “Random” | El sistema ha seleccionado un personaje aleatorio para que utilice el usuario durante el juego. |
| **Identificador** | *CP\_05* |
| **Objetivo del Caso de Prueba** | Verificar que se salga satisfactoriamente del juego al darle click en el botón “Salir”. |
| **Caso de Prueba** | **Criterios de Aceptación** |
| 1. Ingresar nombre de usuario 2. Dar click en botón “Salir” 3. Dar click en botón “Si” en la decisión “Abandonar el juego: Si/No” | El sistema muestra en la pantalla un mensaje de “Se ha salido del juego con éxito”. |
| **Identificador** | *CP\_06* |
| **Objetivo del Caso de Prueba** | Selección del mapa exitoso. |
| **Caso de Prueba** | **Criterios de Aceptación** |
| 1. Ingresar nombre de usuario 2. Seleccionar personaje 3. Hacer click en el botón “Ready” 4. Acceder a la pantalla de elección de mapa 5. Seleccionar un mapa para su uso durante el juego 6. Hacer click en el botón “Start” | El sistema muestra correctamente el mapa del juego seleccionado por el usuario. |
| **Identificador** | *CP\_07* |
| **Objetivo del Caso de Prueba** | Selección correcta del mapa aleatorio. |
| **Caso de Prueba** | **Criterios de Aceptación** |
| 1. Ingresar nombre de usuario 2. Seleccionar personaje 3. Hacer click en el botón “Ready” 4. Acceder a la pantalla de elección de mapa 5. Hacer click en el botón “Random” para seleccionar aleatoriamente un mapa del juego | El sistema ha seleccionado un mapa del juego de modo aleatorio para su uso posterior por el usuario. |
| **Identificador** | *CP\_08* |
| **Objetivo del Caso de Prueba** | Movimiento correcto del personaje durante el juego. |
| **Caso de Prueba** | **Criterios de Aceptación** |
| 1. Ingresar nombre de usuario 2. Seleccionar personaje 3. Hacer click en el botón “Ready” 4. Acceder a la pantalla de elección de mapa 5. Seleccionar un mapa para su uso durante el juego 6. Hacer click en el botón “Start” 7. Acceder al juego 8. Mover el personaje con el teclado | El personaje seleccionado para el juego debe moverse de acuerdo a las teclas presionadas por el usuario. Por ejemplo, si el usuario presiona la flecha derecha del teclado, el personaje se mueve hacia el lado derecho del mapa según el movimiento deseado por parte del usuario. |
| **Identificador** | *CP\_09* |
| **Objetivo del Caso de Prueba** | Mostrar adecuadamente los créditos del juego. |
| **Caso de Prueba** | **Criterios de Aceptación** |
| 1. Ingresar a la pantalla principal del juego 2. Hacer click en el botón “Créditos” | La pantalla muestra información respecto a los integrantes del equipo de trabajo que conformaron para el desarrollo del juego. |
| **Identificador** | *CP\_10* |
| **Objetivo del Caso de Prueba** | Verificar que jugadores mueran |
| **Caso de Prueba** | **Criterios de Aceptación** |
| 1. Jugar 2. Esperar a que mueran | Los personajes deben morir una vez que se les acabe la vida |
| **Identificador** | *CP\_11* |
| **Objetivo del Caso de Prueba** | Verificar que el juego culmine con todos los jugadores muertos |
| **Caso de Prueba** | **Criterios de Aceptación** |
| 1. Jugar 2. Esperar a que mueran | La partida debe culminar una vez que ambos jugadores estén muertos |
| **Identificador** | *CP\_12* |
| **Objetivo del Caso de Prueba** | Determinar ganador |
| **Caso de Prueba** | **Criterios de Aceptación** |
| 1. Jugar 2. Esperar a que mueran | El ganador debe ser aquel jugador que permaneció más tiempo vivo. |
| **Identificador** | *CP\_13* |
| **Objetivo del Caso de Prueba** | Verificar la validación de eventos por el usuario |
| **Caso de Prueba** | **Criterios de Aceptación** |
| 1. Ingresar nombre de usuario 2. Seleccionar personaje 3. Hacer click en el botón “Ready” 4. Acceder a la pantalla de elección de mapa 5. Seleccionar un mapa para su uso durante el juego 6. Hacer click en el botón “Start” 7. Acceder al juego   Mover el personaje con el teclado | El juego debe identificar correctamente los eventos realizados por el usuario tanto con el mouse y el teclado. |
| **Identificador** | *CP\_14* |
| **Objetivo del Caso de Prueba** | Validar registro de nombre |
| **Caso de Prueba** | **Criterios de Aceptación** |
| 1. Ingresar nombre de usuario | Validar que el nombre del jugador sea recepcionado correctamente en el código. |
| **Identificador** | *CP\_15* |
| **Objetivo del Caso de Prueba** | Ingreso al sistema |
| **Caso de Prueba** | **Criterios de Aceptación** |
| 1.Acceder al sistema | Validar que el juego arranque sin errores ni bugs. |
| **Identificador** | *CP\_16* |
| **Objetivo del Caso de Prueba** | Ingreso de los valores para los atributos del sistema |
| **Caso de Prueba** | **Criterios de Aceptación** |
| 1.Acceder al sistema | Validar que la base de datos contenga los valores enviados desde las lógicas. |
| **Identificador** | *CP\_17* |
| **Objetivo del Caso de Prueba** | Validar conexión a la base de datos |
| **Caso de Prueba** | **Criterios de Aceptación** |
| 1.Acceder al sistema | Validar que el motor logre conectarse con la base de datos. |
| **Identificador** | *CP\_18* |
| **Objetivo del Caso de Prueba** | Validar las dimensiones del personaje durante todo el juego |
| **Caso de Prueba** | **Criterios de Aceptación** |
| 1.Acceder al sistema  2. Escoger escenario  3. Escoger personaje  4. Jugar | Validar que las dimensiones del personaje se mantengan iguales a lo largo del juego. |
| **Identificador** | *CP\_19* |
| **Objetivo del Caso de Prueba** | Validar animación de los personajes |
| **Caso de Prueba** | **Criterios de Aceptación** |
| 1.Acceder al sistema  2. Escoger escenario  3. Escoger personaje  4. Jugar | Validar que el personaje este animado a lo largo del juego. |
| **Identificador** | *CP\_20* |
| **Objetivo del Caso de Prueba** | Subir archivos a github |
| **Caso de Prueba** | **Criterios de Aceptación** |
| 1.Seleccionar carpeta de archivos  2.Enviar archivo a git  3.Validar carga de archivo al repositorio | Validar carga de archivo al repositorio |

**8 DIAGRAMA DE ROBUSTEZ**

El diagrama de robustez servirá de guía para conocer cual es el flujo del juego programado. El cual arranca en una pantalla principal, desde la que se elegirá primero el escenario y luego el personaje. Luego, arrancará el juego multijugador en el que ambos jugadores haber muerto para que el juego culmine. Esto se ve reflejado en el siguiente diagrama:



**9 POST MORTEM**

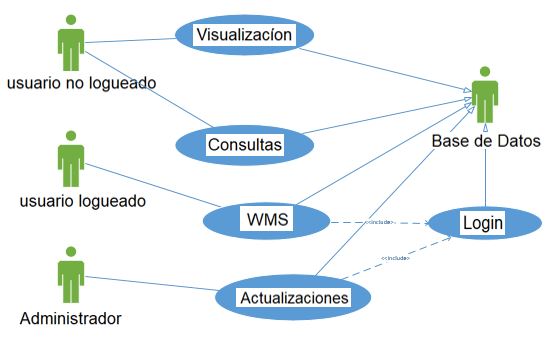
En esta última sección se mostrarán aquellos errores o inconvenientes a los que nos hemos enfrentado a lo largo de este proyecto. Aquí, se identifiacarán los eventos, su descripción, causa y solución.

En la siguiente tabla se muestran los errores identificados. Así como su descripción, causa, solución y opcionalmente recomendación.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **TABLA POSTMORTEM** | | | | | | |
| ID\_ACTIVIDAD | Grupo | ERROR O INCONVENIENTE | DESCRIPCION DEL ERROR | Análisis de la causa del problema | Solución realizada | Recomendación (Opcional) |
| 1 | Arte | Imágenes en .jpg | Las imágenes de los personajes que se deben utilizar deben ser en .png, debido a que sin este formato el fondo de los personajes se sobrepondrá sobre el fondo | Formato de imagen | Quitar el fondo de las imágenes con photoshop y guardarlo en .png | Aplicar formato .png para todos los personajes |
| 2 | Arte | Imágenes de selección de escenarios | Las imágenes de selección de escenarios deben ser del mismo tamaño. Ya que, de ser de diferentes dimensiones esto afectará las lógicas del juego. | Dimensiones de escenarios en la imagen | Redimensionar imágenes | Redimensionar imagen |
| 3 | Motor | Falta de comunicación entre equipos | Debido a que desde el inicio no hubo un consenso, entre los equipos habían discrepancias en cuanto a la visión del producto terminado | Falta de comunicación | Reunión y votación para llegar a una sola visión |  |
| 4 | Motor | Delimitación de funciones entre motor y lógica | Durante la parte inicial de la codificación, no teniamos claras las funciones del equipo de lógica, lo que nos llevó a hacer parte del código que les correspondía | Falta de conocimientos | Definir un backlog para determinar limites claros entre las funciones de cada equipo |  |
| 5 | Motor |  |  |  |  |  |
| 6 | Logicas | Imágenes a usar en el juego | El grupo de Arte compartió las imágenes a usar a lo largo del juego | Mala comunicación con el grupo de Arte | Comunicar el problema al grupo de arte para que solucionen el problema | Mejorar comunicación con el grupo de Arte para obtener los sprites adecuados siempre |
| 7 | Logicas | Fondo de selección de escenarios | Las imágenes en miniatura en la pantalla de selección de escenario o mapa tenían tamaños diferentes | Mala comunicación con el grupo de Arte | Comunicar el problema al grupo de arte para que solucionen el problema | Mejorar comunicación con el grupo de Arte para obtener los sprites adecuados siempre |
| 8 | Networking | Conexion entre clientes |  |  |  |  |

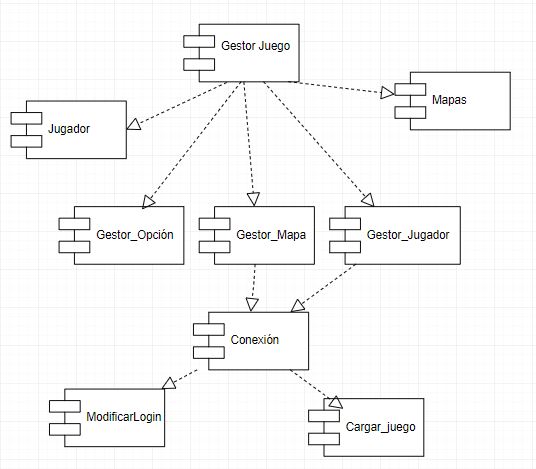
**10 CASOS DE USO**

En esta sección se detallarán los procesos que se realizaron en el juego. Son la interacción entre actores y el sistema que produce un resultado observable de valor para un actor.

****

**11 CASOS DE USO**

En esta sección se detallarán los componentes, interfaces y relaciones entre ellos.

****