Proyecto final

Tecnologías para la construcción de software



Luis Angel Olivo MartÍnez

Roberto Quiñones Cordova

Juego Serpientes y Escaleras

Contenido

[1 Introducción 5](#_Toc26743625)

[1.1 Justificación del proyecto 5](#_Toc26743626)

[2 Especificación de requerimientos: 5](#_Toc26743627)

[2.1 Propósito 5](#_Toc26743628)

[2.2 Alcance 5](#_Toc26743629)

[2.3 Personal involucrado 6](#_Toc26743630)

[2.4 Definiciones, acrónimos y abreviaturas 6](#_Toc26743631)

[2.5 Referencias 6](#_Toc26743632)

[2.6 Resumen 6](#_Toc26743633)

[2.7 Descripción general 7](#_Toc26743634)

[2.7.1 Perspectiva del producto 7](#_Toc26743635)

[2.8 Funcionalidad del producto. 7](#_Toc26743636)

[2.8.1 Modelo de casos de uso 7](#_Toc26743637)

[2.8.2 Descripciones de caso de uso y prototipos 8](#_Toc26743638)

[2.9 Características de los usuarios 22](#_Toc26743639)

[2.10 Restricciones 22](#_Toc26743640)

[2.11 Suposiciones y dependencias 23](#_Toc26743641)

[2.12 Evolución previsible del sistema 23](#_Toc26743642)

[2.13 Requisitos específicos 23](#_Toc26743643)

[2.13.1 Requisitos Funcionales 23](#_Toc26743644)

[2.14 Requisitos comunes de las interfaces 25](#_Toc26743645)

[2.14.1 Interfaces de usuario 25](#_Toc26743646)

[2.14.2 Interfaces de hardware 25](#_Toc26743647)

[2.14.3 Interfaces de software 25](#_Toc26743648)

[2.15 Requisitos funcionales 25](#_Toc26743649)

[2.16 Requisitos no funcionales 26](#_Toc26743650)

[2.16.1 Seguridad 26](#_Toc26743651)

[2.16.2 Usabilidad 26](#_Toc26743652)

[3 Modelo de dominio 27](#_Toc26743653)

[4 Dinámica del juego 27](#_Toc26743654)

[5 Estándar de codificación 28](#_Toc26743655)

[6 Estándar de codificación en C# 28](#_Toc26743656)

[6.1 Convenciones de nomenclatura 28](#_Toc26743657)

[6.2 Convenciones de los comentarios 29](#_Toc26743658)

[6.3 String (Tipo de datos) 29](#_Toc26743659)

[6.4 Tipo de datos sin signo 31](#_Toc26743660)

[6.5 Delegados 31](#_Toc26743661)

[6.6 Operadores && y || 33](#_Toc26743662)

[6.7 New (Operador) 34](#_Toc26743663)

[6.8 Consultas LINQ 36](#_Toc26743664)

[7 Modelos de diseño 38](#_Toc26743665)

[7.1 Diagrama de secuencia 38](#_Toc26743666)

[7.2 Diagrama de despliegue 39](#_Toc26743667)

[8 Justificación de los frameworks y tecnología 40](#_Toc26743668)

[8.1 justificación 40](#_Toc26743669)

[9 Pruebas 40](#_Toc26743670)

[9.1 Pruebas del sistema 40](#_Toc26743671)

[9.2 Pruebas de usabilidad 41](#_Toc26743672)

[9.2.1 Introducción 41](#_Toc26743673)

[9.2.2 Objetivo del documento 41](#_Toc26743674)

[9.2.3 Descripción del prototipo de la interfaz 41](#_Toc26743675)

[9.2.4 Planeación de la prueba de usabilidad 43](#_Toc26743676)

[9.2.5 Características de los participantes 43](#_Toc26743677)

[9.2.6 Describir método tipo de prueba 43](#_Toc26743678)

[9.2.7 Entorno de prueba, equipo y logística 43](#_Toc26743679)

[9.2.8 Datos para recolectar 44](#_Toc26743680)

[9.2.9 Descripción de las herramientas de medición Cuestionario SUS (System Usability Scale) 44](#_Toc26743681)

[9.2.10 Aplicando las pruebas de usabilidad 44](#_Toc26743682)

[9.2.11 Descripción de cómo se llevó a cabo la prueba 44](#_Toc26743683)

[9.2.12 Descripción de los participantes 45](#_Toc26743684)

[9.2.13 Resultados de las pruebas de usabilidad 46](#_Toc26743685)

[9.2.14 Conclusiones 49](#_Toc26743686)

[9.2.15 Sobre resultados 49](#_Toc26743687)

[9.2.16 Modificaciones a futuro 49](#_Toc26743688)

[9.2.17 Experiencia 50](#_Toc26743689)

[10 Análisis estático de código 50](#_Toc26743690)

[11 Conclusiones 50](#_Toc26743691)

[11.1 Lecciones aprendidas por estudiante 50](#_Toc26743692)

[11.1.1 Luis Angel Olivo MartÍnez 50](#_Toc26743693)

[11.1.2 Roberto Quiñones Cordova 50](#_Toc26743694)

[12 Referencias 50](#_Toc26743695)

# Introducción

## Justificación del proyecto

Este proyecto fue desarrollado como proyecto para la experiencia educativa de Tecnologías para la construcción de software y tiene como propósito mejorar nuestras habilidades con las tecnologías para la construcción de software en diferentes ámbitos, por ejemplo:

El desarrollo de aplicaciones con uso de IDE´s con C#

El desarrollo de interfaces graficas con el uso de XAML con ayuda de la integración del IDE

Desarrollo de aplicaciones divididas entre cliente y servidor, desarrollo de aplicaciones distribuidas y desarrollo de aplicaciones con uso de frameworks de persistencia (Entity framework)

El proyecto tiene la intención de permitir el juego de Serpientes y escaleras de diferentes maneras:

Juego con añadidos de Dragones que agrega una funcionalidad aparte de las escaleras y las serpientes, y muerte súbita que cambia a todas las escaleras y serpientes por dragones.

Juego de campaña para un único jugador que permite jugar 5 niveles de historia con una ficha seleccionada por el usuario de las que brinda el sistema.

Juego multijugador para 4 personas en la misma red.

Internacionalización para los idiomas inglés y español.

# Especificación de requerimientos:

Esta especificación de requerimientos es desarrollada para el sistema “Serpientes Escaleras y Dragones” y fue desarrollada basándose en las directrices dadas por el estándar IEEE 830

## Propósito

Esta especificación de requerimientos de software tiene como propósito definir las especificaciones funcionales y no funcionales para el desarrollo de un sistema de Serpientes y escaleras que permitirá El juego del clásico juego de mesa de serpientes y escaleras pero de manera digital tanto para un jugador como en multijugador para 4 personas a la vez.

## Alcance

El sistema está dirigido a jugadores jóvenes que ya tengan conocimiento respecto a la funcionalidad del juego, el sistema permitirá tener una cuenta que se deberá crear, a su vez se desarrollara una función de validar cuenta que mandara un código de verificación de cuenta a el correo electrónico del usuario, el juego contara con un sistema de login para mantener cada cuenta de usuario separada y personalizada, el juego contara con soporte para los idiomas de ingles y de español, permitirá la elección de fichas para el multijugador, la selección de su idioma, la creación de partidas, unirse a una partida, y jugar la campaña que constara de 5 niveles diferentes que presentaran pequeñas historias para el desarrollo de la misma.

## Personal involucrado

|  |  |
| --- | --- |
| **Nombre** | Olivo Martínez Luis Ángel |
| **Rol** | Analista, tester, diseñador y programador |
| **Categoría Profesional** | Estudiante Ingeniería de Software |
| **Responsabilidad** | Análisis de información, diseño, pruebas y programación del SEYD |
| **Información de contacto** | luisolivo340@gmail.com |

|  |  |
| --- | --- |
| **Nombre** | Roberto Quiñones Córdova |
| **Rol** | Analista, tester, diseñador y programador |
| **Categoría Profesional** | Estudiante Ingeniería de Software |
| **Responsabilidad** | Análisis de información, diseño, pruebas y programación del SEYD |
| **Información de contacto** | Quinonesroberto92@gmail.com |

## Definiciones, acrónimos y abreviaturas

|  |  |
| --- | --- |
| ***Nombre*** | ***Descripción*** |
| **Usuario** | Persona que usará el sistema |
| **SEYD** | Serpientes Escaleras y Dragones |
| **ERS** | Especificación de Requisitos Software |
| **RF** | Requerimiento Funcional |
| **RNF** | Requerimiento No Funcional |

## Referencias

|  |  |
| --- | --- |
| **Título del Documento** | **Referencia** |
| Standard IEEE 830 - 1998 | IEEE |

## Resumen

Esta especificación puede ser considerada como dividida en tres apartados. En este primer apartado se realiza una breve introducción al mismo y se proporciona una visión general sobre lo que será el sistema.

En el segundo apartado del documento se realiza una descripción general del sistema para conocer los aspectos principales de las funcionalidades. Abarcando la perspectiva del producto, las funcionalidades, las características de los usuarios, y las restricciones y suposiciones.

Por último, en el tercer apartado del documento se especifican detalladamente tanto los requisitos funcionales y no funcionales que debe cumplir el sistema. Siendo esta la sección más extensa.

## Descripción general

### Perspectiva del producto

El SEYD será un sistema que se conectará en red local y que a través de ella permitirá el juego simultaneo de 4 jugadores a la vez o de uno solo en su modo campaña además de permitir la creación y breve personalización de su perfil de usuario.

## Funcionalidad del producto.

### Modelo de casos de uso



### Descripciones de caso de uso y prototipos

#### CU-01 Registrarse

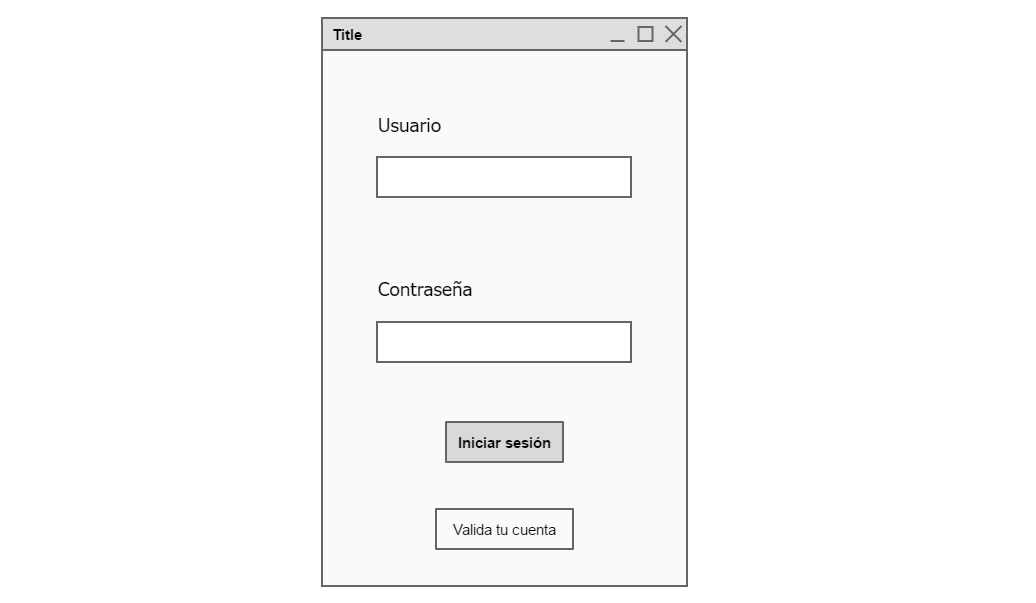
|  |  |
| --- | --- |
| **­­ID:** | CU-01 |
| **Nombre:** | Registrarse |
| **Autor(es):** | Roberto Quiñones Córdova |
| **Fecha de creación:** | 22 de agosto de 2019 |
| **Fecha de actualización:** | 01 de diciembre de 2019 |
| **Actor(es):** | Jugador |
| **Descripción:** | El jugador crea una cuenta en el sistema para poder acceder al mismo |
| **Precondiciones:** |  |
| **Flujo Normal:** | 1. El jugador da clic en el botón “Regístrate”. 2. El sistema muestra un formulario en el que solicita el nombre, correo electrónico, nombre de usuario y contraseña del jugador y los botones “Aceptar” y “Cancelar”. (EX-1) 3. El jugador ingresa los datos en su correspondiente campo y da clic en el botón “Aceptar”. (FA-1) (EX-2) 4. El sistema valida los datos y envía un correo electrónico con un número de confirmación y se extiende al CU-02 Validar cuenta. (FA-2) 5. Finaliza el caso de uso. |
| **Flujos Alternos:** | FA-1.- Datos inválidos   1. El jugador ingresa un dato inválido en algún campo del formulario. 2. El sistema muestra el mensaje “Por favor, revise sus datos” en una ventana emergente. 3. El jugador da clic en “Aceptar”. 4. El sistema vuelve al paso 2 del flujo normal.   FA-2.- Formato de correo invalido   1. El sistema muestra el mensaje “Ocurrió un error al enviar el correo, revise su correo” en una ventana emergente. 2. El jugador da clic en “Aceptar”. 3. El sistema vuelve al paso 2 del flujo normal. |
| **Excepciones:** | EX-1.- No hay conexión a el servidor   1. El sistema muestra el mensaje “Ocurrió un problema al conectar con el servidor” en una ventana emergente. 2. El jugador da clic en “Aceptar”. 3. Finaliza el caso de uso.   EX-2.- Clic en “Cancelar”   1. El jugador da clic en “Cancelar” 2. Finaliza el caso de uso. |
| **Postcondiciones:** | Se crea y se guarda un JUGADOR en la base de datos |
| **Incluye:**  **(relación Include)** |  |
| **Extiende:**  **(relación Extend)** | CU-02 Validar cuenta. |

#### CU-02 Validar cuenta

|  |  |
| --- | --- |
| **­­ID:** | CU-02 |
| **Nombre:** | Validar cuenta extendido de Registrarse e Iniciar sesión |
| **Autor(es):** | Roberto Quiñones Córdova |
| **Fecha de creación:** | 26 de agosto de 2019 |
| **Fecha de actualización:** | 01 de diciembre de 2019 |
| **Actor(es):** | Jugador |
| **Descripción:** | El jugador valida su cuenta mediante el código que le ha enviado el sistema |
| **Precondiciones:** | El jugador ha registrado su cuenta previamente  El sistema ha enviado un correo electrónico al jugador |
| **Flujo Normal:** | 1. El jugador da clic en el botón “Validar cuenta”. 2. El sistema muestra una ventana con el texto “se le mando un código de verificación a su correo, ingréselo aquí”, un textfield y los botones “Aceptar” y “Cancelar”. (FA-1) 3. El jugador ingresa el código que recibió en el correo electrónico. Y presiona el botón “aceptar” 4. El sistema valida el código y muestra en una pantalla emergente el mensaje: “Tu cuenta ha sido validada”. 5. El jugador da clic en “Aceptar”. 6. El sistema muestra la pantalla de inicio de sesión. 7. Finaliza el caso de uso. |
| **Flujos Alternos:** | FA-1.- Código incorrecto   1. El sistema muestra el mensaje “El código no pertenece a ninguna cuenta, revísalo” en una ventana emergente. 2. El jugador da clic en “Aceptar”. 3. El sistema vuelve al paso 2 del flujo normal. |
| **Excepciones:** | EX-1.- No hay conexión a el servidor   1. El sistema muestra el mensaje “Ocurrió un problema al conectar con el servidor” en una ventana emergente. 2. El jugador da clic en “Aceptar”. 3. Finaliza el caso de uso.   EX-2.- Clic en “Cancelar”   1. El jugador da clic en “Cancelar” 2. Finaliza el caso de uso. |
| **Postcondiciones:** | La cuenta quedará activada y lista para usarse en el juego. |
| **Incluye:**  **(relación Include)** |  |
| **Extiende:**  **(relación Extend)** |  |

#### CU-03 Iniciar Sesión

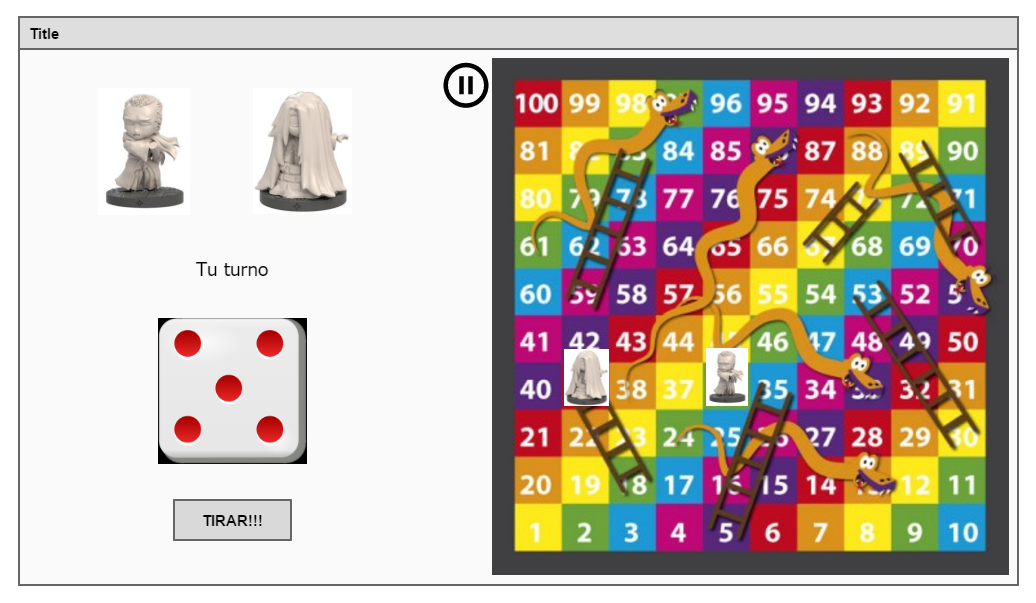
|  |  |
| --- | --- |
| **­­ID:** | CU-03 |
| **Nombre:** | Iniciar sesión |
| **Autor(es):** | Roberto Quiñones Córdova |
| **Fecha de creación:** | 31 de agosto de 2019 |
| **Fecha de actualización:** | 01 de diciembre de 2019 |
| **Actor(es):** | Jugador |
| **Descripción:** | El jugador ingresa al sistema mediante su cuenta de jugador. |
| **Precondiciones:** | El jugador debe tener una cuenta ya registrada y validada en el sistema.  Debe estar conectado al servidor |
| **Flujo Normal:** | 1. El sistema muestra un formulario donde solicita el nombre de usuario y contraseña y los botones “Iniciar sesión”, “Valida tu cuenta” y “Registrarse” (EX-1) 2. El jugador ingresa los datos correspondientes y da clic en “Iniciar sesión”. (FA-4) 3. El sistema valida los datos y muestra la pantalla principal del juego. (FA-1) (FA-2) |
| **Flujos Alternos:** | FA-1.- Contraseña incorrecta   1. El sistema muestra el mensaje “Contraseña incorrecta” en una ventana emergente. 2. El jugador da clic en “Aceptar”. 3. El sistema vuelve al paso 1 del flujo normal.   FA-2.- Usuario inexistente   1. El sistema muestra el mensaje “Usuario inexistente. Por favor registre una cuenta nueva o revise el usuario” en una ventana emergente. 2. El jugador da clic en “Aceptar”. 3. El sistema vuelve al paso 1 del flujo normal.   FA-4.- Clic en “Regístrate”.   1. El jugador da clic en “Registrarse”. 2. El sistema extiende al CU-01 Registrarse. 3. Regresa al paso 1 del flujo normal. |
| **Excepciones:** | EX-1.- No hay conexión a el servidor   1. El sistema muestra el mensaje “Ocurrió un problema al conectar con el servidor” en una ventana emergente. 2. El jugador da clic en “Aceptar”. 3. Finaliza el caso de uso. |
| **Postcondiciones:** | El jugador podrá acceder al juego y sus opciones. |
| **Incluye:**  **(relación Include)** |  |
| **Extiende:**  **(relación Extend)** | CU-02 Registrarse |



#### CU-04 Iniciar Campaña

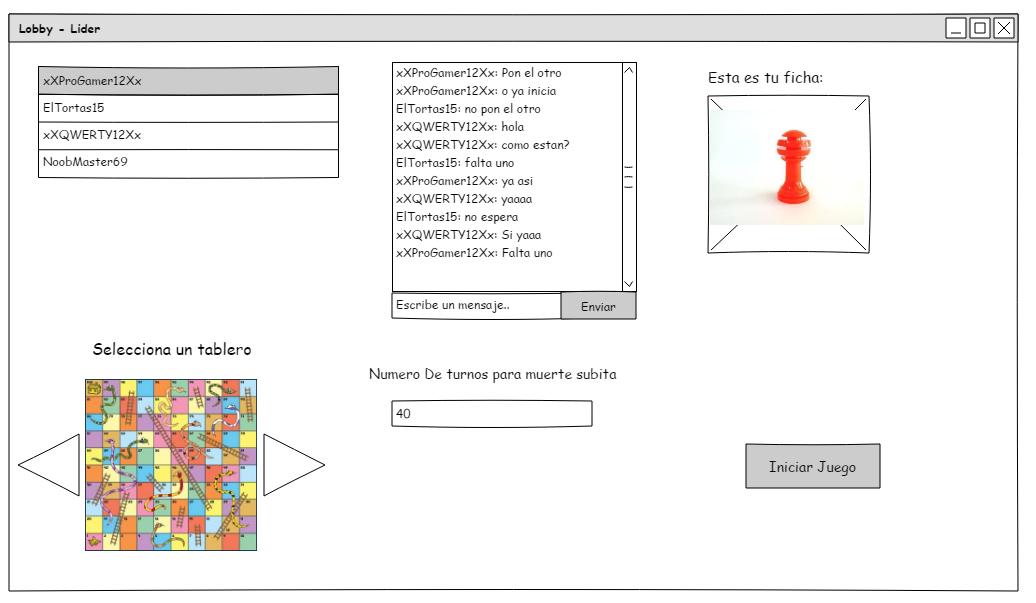
|  |  |
| --- | --- |
| **­­ID:** | CU-04 |
| **Nombre:** | Jugar campaña |
| **Autor(es):** | Roberto Quiñones Córdova |
| **Fecha de creación:** | 31 de agosto de 2019 |
| **Fecha de actualización:** | 1 de diciembre de 2019 |
| **Actor(es):** | Jugador |
| **Descripción:** | El jugador inicia una partida en modo individual |
| **Precondiciones:** | El jugador debe haber iniciado sesión en el sistema |
| **Flujo Normal:** | 1. El jugador da clic en el botón “Campaña” del menú principal. 2. El sistema muestra otra ventana con los botones “Nueva campaña”, “Continuar campaña” y “Regresar”. 3. El jugador da clic en “Nueva campaña”. (FA-1) (EX-1) 4. El sistema muestra la pantalla de historia inicial y un botón de continuar 5. El jugador presiona el botón de continuar. 6. Se repite el proceso hasta que no haya más imágenes de historia 7. El sistema muestra el tablero de juego, un dado, la ficha del jugador, la ficha de la computadora, y el botón “TIRAR!!!”. 8. El jugador da clic en el botón tirar. (EX -2) 9. El sistema muestra un número aleatorio en el dado, mueve la ficha del jugador hasta la posición marcada, vuelve a generar otro numero del dado y mueve la ficha de la computadora hasta la posición marcada. 10. Mientras ninguna ficha toque la casilla final se repiten los pasos 8 y 9. 11. El jugador consigue que su ficha llegue a la casilla final. (FA-3) 12. El sistema muestra el mensaje: “ganaste” y un botón “Aceptar” y actualiza su nivel de avance en la base de datos. 13. El jugador da clic en “Aceptar”. 14. El sistema cambia el tablero y devuelve las fichas a la posición inicial. 15. Se repite desde el paso 6 para los cinco niveles. 16. El sistema muestra la cinemática final y el mensaje “Felicitaciones, has terminado la campaña”. 17. Finaliza el caso de uso. |
| **Flujos Alternos:** | FA-1 Clic en “Continuar campaña”.   1. El jugador da clic en “Continuar campaña” 2. El sistema recupera de la base de datos el nivel en el que va a el jugador. 3. El sistema vuelve al paso 6 del flujo normal a partir del nivel guardado.   FA-2.- La ficha de la computadora llega a la casilla final.   1. El sistema muestra el mensaje “Perdiste, vuelve a intentarlo” y un botón “Aceptar”. 2. El jugador da clic en “Aceptar”. 3. El sistema vuelve al paso 6 del flujo normal en el mismo nivel. |
| **Excepciones:** | EX-1 Clic en “Regresar”   1. El jugador da clic en “Regresar”. 2. Finaliza el caso de uso.   EX-2 Clic en “Guardar y salir”   1. El jugador da clic en “Guardar y salir” 2. El sistema guarda el nivel de la partida y sale del juego. 3. Finaliza el caso de uso. |
| **Postcondiciones:** | Se actualiza el nivel del jugador en la base de datos |
| **Incluye:**  **(relación Include)** |  |
| **Extiende:**  **(relación Extend)** |  |





#### CU-05 Entrar al lobby

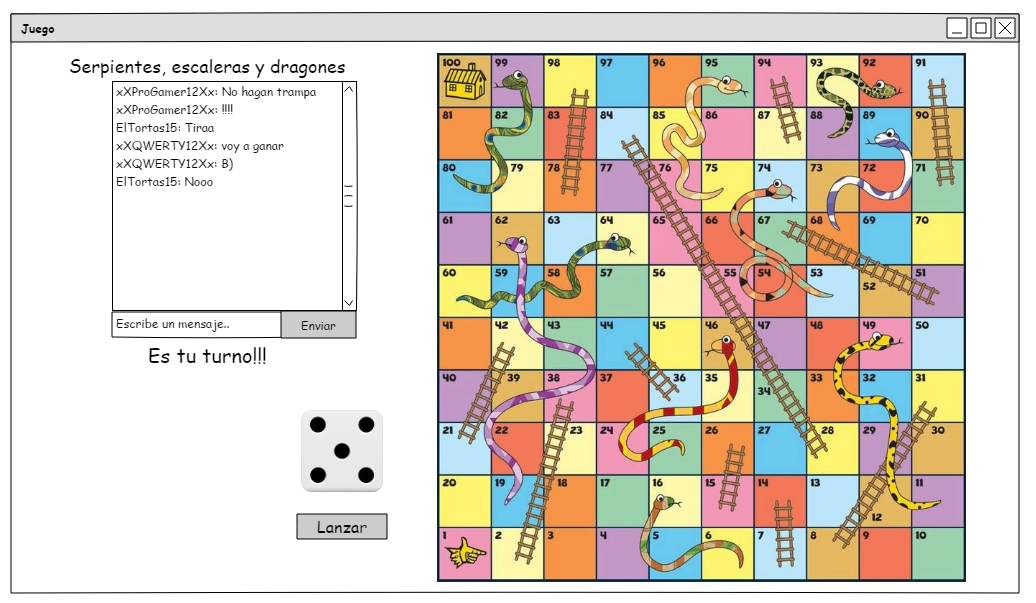
|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | CU-05 Entrar al lobby extendido de Iniciar sesión | | |
| **Versión** | Versión 1 25/08/19 | versión 2 1/12/19 | | |
| **Autores** | Luis Angel Olivo MartÍnez | | |
| **Fuentes** | Presiona el botón “Multijugador” en el menú principal | | |
| **Descripción** | El sistema permitirá a un jugador entrar al lobby previo de la partida e intercambiar mensajes en el chat | | |
| **Precondición** | El jugador debe haber iniciado sesión previamente y debe haber conexión con el servidor | | |
| **Secuencia**  **Normal** | **Paso** | | **Acción** |
| 1 | | El sistema recupera de la base de datos una lista de las partidas que están actualmente activas junto con el numero de jugadores, nombre de la partida y líder de la partida, además de mostrar una etiqueta de crear partida con un campo para el nombre de la partida y un botón de “Crear partida” y uno de “Unirse” |
| 2 | | El jugador Escribe un nombre para la partida y presiona el botón “Crear partida” |
| 3 | | El sistema valida el nombre de la partida y crea en el servidor una nueva partida. |
| 4 | | El sistema abre una nueva ventana con el titulo  de lobby, los nombres de los jugadores que están actualmente en el lado superior izquierdo de la pantalla, un cuadro mostrando la ficha que usara el jugador en la parte superior derecha de la pantalla, en la parte superior central se muestra el chat de texto con un cuadro para escribir un mensaje y se muestra en la parte inferior izquierda una selección de tablero en con botones para cambiarlo, En la parte inferior central se muestra un campo de texto para que se escriba el número de movimientos antes de la muerte súbita y en la parte inferior derecha se muestra el botón “Iniciar partida” bloqueado al no haber 4 jugadores en la sala. |
| 5 | | Se unen más jugadores para alcanzar los 4 |
| 6 | | El sistema desbloquea el botón de “Comenzar juego” |
| 7 | | El jugador selecciona sus preferencias para la partida y presiona el botón “Comenzar juego” (EX2) |
| 8 | | El sistema recoge todas las selecciones del líder y crea un nuevo JUEGO en donde se aplican las configuraciones del líder |
| 9 | | Se extiende al caso de uso CU-05 “Iniciar juego” |
| n | |  |
| **Postcondición** | El sistema termina en el caso de uso “Comenzar juego” | | |
| **Flujos alternos**  “Se pierde la conexión con algún jugador jugadores” | Paso | Acción | |
| 1 | El sistema bloquea el botón de “Comenzar juego” | |
| 2 | Se completan los jugadores nuevamente | |
| 3 | Regresa al paso 4 del flujo normal | |
| n |  | |
| **“Unirse”** | 1 | El jugador selecciona una partida disponible donde hay menos de 4 jugadores actualmente y presiona el botón “Unirse” | |
| 2 | El sistema lo agrega a la partida en el servidor y lo agrega en las ventanas de todos los jugadores que están en el lobby. | |
| 3 | Regresa al paso 4 del flujo normal pero sin mostrar la muerte súbita ni el botón “Comenzar juego” | |
| 4 |  | |
| **Excepciones**  “Se presiona el botón salir” | **Paso** | | **Acción** |
| 1 | | El usuario (jugador) presiona el botón salir |
| 2 | | El sistema quita a el jugador de la partida en el servidor y regresa al menú principal |
| “se pierde la conexión con el servidor” | 1 | | El sistema arroja un mensaje informando que no hay conexión con el servidor y un botón “aceptar” |
| 2 | | El usuario presiona aceptar |
| 3 | | El sistema termina el caso de uso y regresa a la ventana de inicio de sesión |
| 4 | |  |
| n | |  |
| **Frecuencia esperada** | 1 vez/juego | | |
| **Incluye** |  | | |
| **Extiende** | CU-06 “iniciar juego” y CU-08 “Mandar mensaje” | | |
| **Importancia** | Media | | |
| **Urgencia** | Media | | |
| **Comentarios** | El caso de uso puede esperar si hay poco tiempo y se puede pasar a un juego directo sin configurar.  El caso de uso Mandar mensaje se puede extender en cualquier momento que el jugador desee mandar un mensaje. | | |

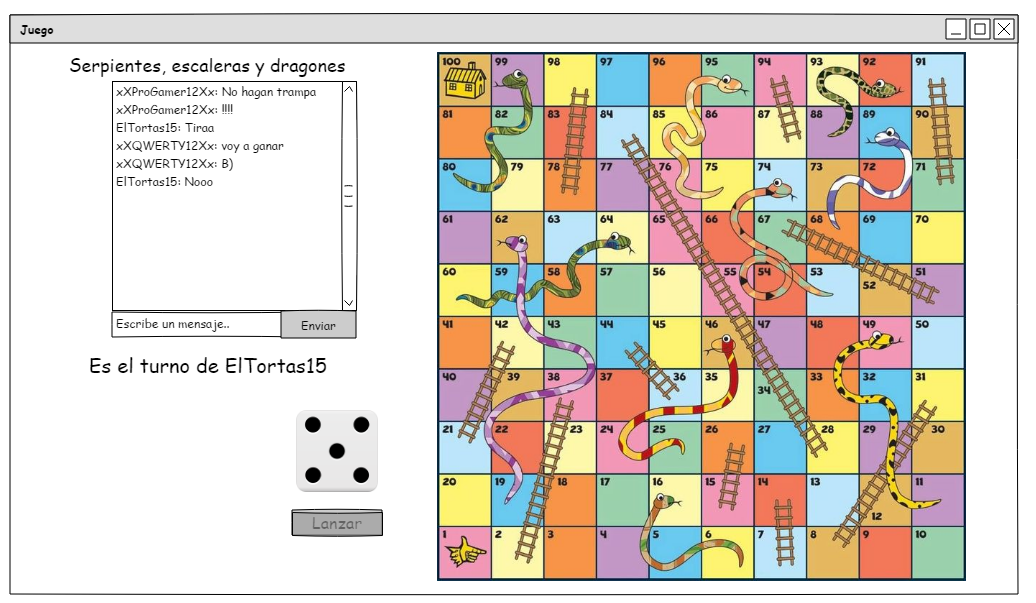




#### CU-06 Iniciar Juego

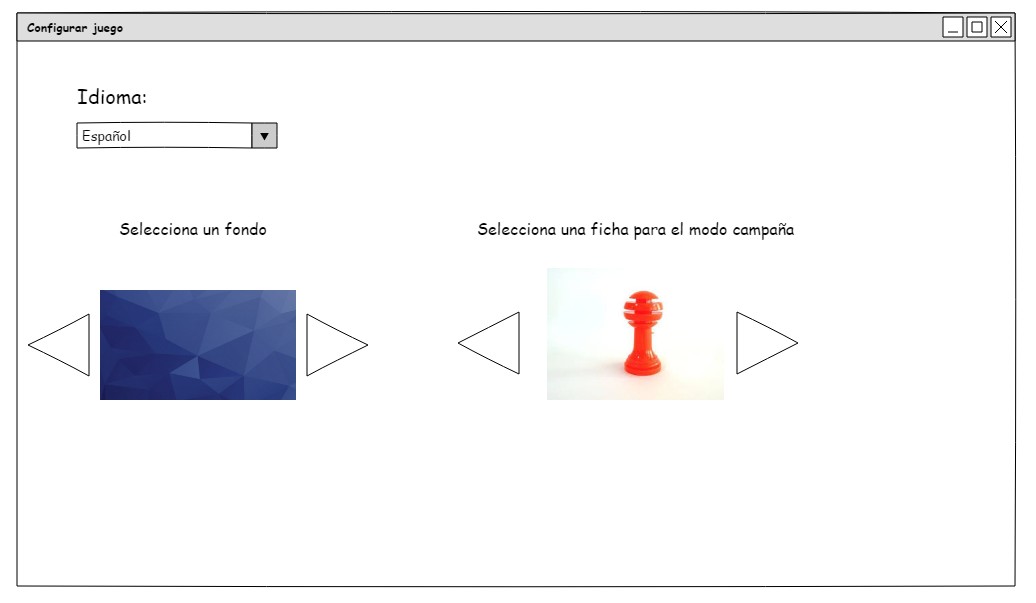
|  |  |  |
| --- | --- | --- |
| **ID** | CU-06 Iniciar juego extendido de Entrar al lobby | |
| **Versión** | Versión 1 25/08/19 | versión 2 01/12/19 | |
| **Autores** | Luis Angel Olivo MartÍnez | |
| **Disparador** | El jugador (líder de grupo) presiona el botón “Comenzar juego” en el lobby | |
| **Descripción** | Este caso de uso permite el desarrollo de un juego en línea de serpientes y escaleras después de que se configure en el caso de uso anterior | |
| **Precondición** | El sistema debe tener 4 jugadores conectados y debe ser configurado el juego previamente en el lobby | |
| **Secuencia**  **Normal** | **Paso** | **Acción** |
| 1 | El sistema muestra una pantalla con el título en la parte superior, un dado y un botón bloqueado de “Lanzar!!”, además de la ventana de chat con el botón para mandar mensaje en la parte izquierda de la ventana, mientras que en la parte central y derecha se muestra el tablero con las cuatro fichas en la posición de inicio (1). |
| 2 | Mientras un jugador no llegue a la última casilla del tablero: |
| 3 | Dependiendo si es o no el turno del jugador: |
| 4 | Caso es su turno: |
| 5 | El sistema habilita el botón “Lanzar” |
| 6 | El jugador presiona el botón “Lanzar” |
| 7 | El sistema crea un numero aleatorio entre 1 y 6 y el numero resultante lo refleja en el dado antes descrito (FA2) |
| 8 | El sistema con base a el número que toca en el dado adelanta la ficha del jugador que tiro el número de casillas que indique el dado. (FA1) |
| 9 | Caso no es su turno: |
| 10 | Se muestra el mensaje “Es el turno de” + “El nickname del JUGADOR de quien es turno” |
| 11 | Una vez que un jugador llegue a la casilla final, se mostrara el mensaje a todos de “El ganador de la partida es” + “nickname del jugador que llego a la última casilla” |
| 12 | El sistema termina el juego y los manda de nuevo al lobby |
| **Postcondición** | El juego regresa al lobby para una nueva partida | |
| **Flujos alternos**  **“Muerte súbita” (FA1)** | **Paso** | **Acción** |
| 1 | El sistema alcanza el número establecido para la “muerte súbita” en los turnos de algún jugador |
| 2 | El sistema cambia el tablero por uno con solo dragones y regresa a el paso 2 del flujo normal |
| 3 |  |
| n |  |
| **Excepciones**  “Se perdió la conexión con el servidor” | **Paso** | **Acción** |
| 1 | El sistema abre un mensaje indicando que se perdió la conexión con el servidor y un botón aceptar |
| 2 | El jugador presiona el botón “Aceptar” |
| 3 | El sistema vuelve a la pantalla de inicio de sesión |
| **Rendimiento** | **Paso** | **Cota de tiempo** |
|  | 1 |  |
|  | 2 |  |
| **Incluye** |  | |
| **Extiende** | CU-08 Mandar mensaje | |
| **Importancia** | Alta | |
| **Urgencia** | Alta | |
| **Comentarios** | El caso de uso mandar mensaje se puede extender en cualquier momento que el jugador quiera mandar un mensaje | |

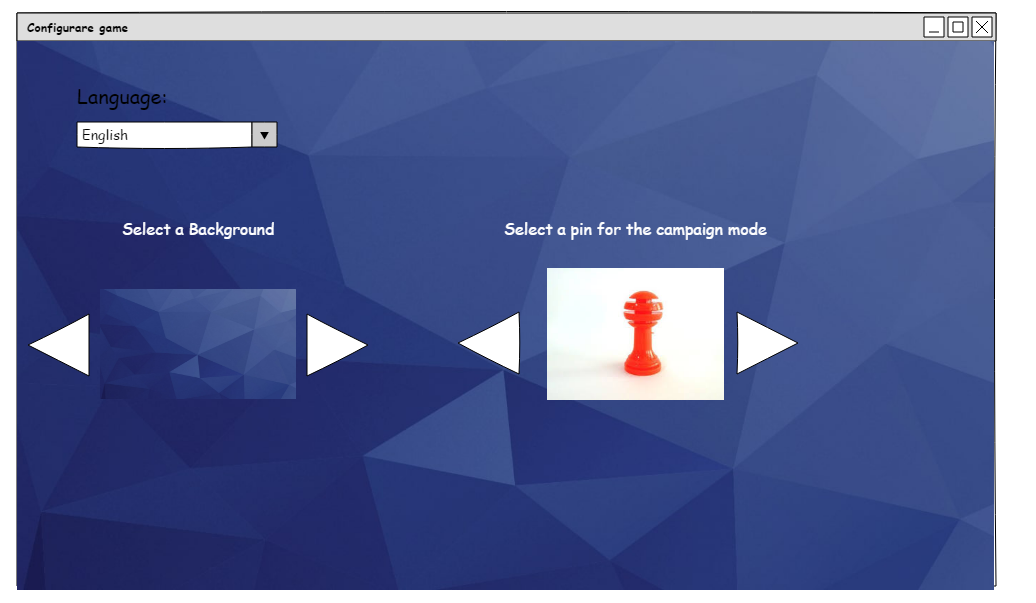




#### CU-07 Configurar Juego

|  |  |  |
| --- | --- | --- |
| **ID** | CU-07 Configurar juego extendido de iniciar sesión | |
| **Versión** | Versión 1 25/08/19 | |
| **Autores** | Luis Angel Olivo MartÍnez | |
| **Disparador** | Se presiona el botón opciones del menú principal | |
| **Descripción** | Este caso de uso le permite al jugador configurar funciones básicas de su juego como el idioma o su ficha de campaña | |
| **Precondición** | Se debe haber iniciado sesión correctamente.  Disparador: Se presiona el botón “configuraciones” en el menú principal | |
| **Secuencia**  **Normal** | **Paso** | **Acción** |
| 1 | El sistema muestra una ventana con el título del juego arriba, un combo box para seleccionar el idioma y un cuadro para selección de ficha en campaña y abajo un botón “Cancelar” y uno “Guardar”. |
| 2 | El jugador configura lo que quiera configurar y presiona el botón aceptar y guardar (EX1) |
| 3 | El sistema recolecta los datos configurados y los guarda en el JUGADOR actual en la base de datos. |
| **Postcondición** | Se alteran los datos de un JUGADOR en la base de datos. | |
| **Excepciones**  “Se presiona el botón salir” | **Paso** | **Acción** |
| 1 | El usuario (jugador) presiona el botón salir |
| 2 | El sistema no guarda los datos y regresa al menú principal. |
| n |  |
| **Rendimiento** | **Paso** | **Cota de tiempo** |
|  | 1 |  |
|  | 2 |  |
| **Frecuencia esperada** | 1 vez cada 10 juegos | |
| **Incluye** |  | |
| **Extiende** |  | |
| **Importancia** | Poco importante | |
| **Urgencia** | Baja | |
| **Comentarios** | El caso de uso puede esperar. | |





#### CU-08 Mandar Mensaje

|  |  |  |
| --- | --- | --- |
| **ID** | CU-08 Mandar mensaje extendido de entrar al lobby e iniciar juego | |
| **Versión** | Versión 1 26/08/19 | |
| **Autores** | Luis Angel Olivo MartÍnez | |
| **Disparador** | Se presiona sobre el campo de texto para escribir mensaje | |
| **Descripción** | Este caso de uso le permite al jugador mandar un mensaje en el chat de texto en el lobby o en el juego | |
| **Precondición** | Se debe haber iniciado sesión correctamente. | |
| **Secuencia**  **Normal** | **Paso** | **Acción** |
| 1 | El jugador presiona sobre el campo de texto del chat que se encuentra en el lobby o en el juego, escribe un mensaje y presiona el botón enviar junto al cuadro de texto |
| 2 | El Sistema toma el mensaje y lo manda a el servidor para que lo escriba en las ventanas de todos los demás usuarios en la misma partida |
| **Postcondición** | Se agrega un mensaje a la zona de chat de los jugadores | |
| **Flujos alternos** | **Paso** | **Acción** |
| 1 |  |
| 2 |  |
| 3 |  |
| n |  |
|  | **Paso** | **Acción** |
| 1 |  |
| 2 |  |
| 3 |  |
| **Rendimiento** | **Paso** | **Cota de tiempo** |
|  | 1 |  |
|  | 2 |  |
| **Frecuencia esperada** | 20 veces/min. | |
| **Incluye** |  | |
| **Extiende** |  | |
| **Importancia** | Medianamente importante | |
| **Urgencia** | Media | |
| **Comentarios** |  | |

## Características de los usuarios

|  |  |
| --- | --- |
| Tipo de usuario | Jugador |
| Formación | Estudiante |
| Habilidades | Manejo intermedio de la tecnología |
| Actividades | Juego y control de opciones de cuenta |

## Restricciones

* Interfaz para ser usada en computadoras personales
* Lenguaje para desarrollo: C#
* Entorno de desarrollo: Visual Studio
* El sistema deberá cuidar los detalles de la usabilidad para permitir la facilidad de aprendizaje de los usuarios hacia las diferentes actividades que permite el sistema.

## Suposiciones y dependencias

* Los requisitos aquí descritos y son los necesarios para el funcionamiento del sistema.
* Los dispositivos en los que se utilizará el sistema cumplen con los requisitos indicados para la ejecución correcta del sistema.
* La interfaz de usuario será intuitiva para su manejo sencillo y eficiente.

## Evolución previsible del sistema

Se desarrollarán versiones posteriores para corregir errores y mejorar en general diferentes aspectos del sistema

## Requisitos específicos

### Requisitos Funcionales

|  |  |
| --- | --- |
| **Identificación del requerimiento:** | RF01 |
| **Nombre del Requerimiento:** | Registrarse |
| **Características:** | Los jugadores deberán de ser capaces de crear una cuenta con su información personal |
| **Descripción del requerimiento:** | Registrarse: permite el registro de un jugador en el sistema creándole una nueva cuenta de usuario que se guardará en la base de datos |
| **Requerimiento NO funcional:** | Usabilidad |
| **Prioridad del requerimiento:**  Alta | |

|  |  |
| --- | --- |
| **Identificación del requerimiento:** | RF02 |
| **Nombre del Requerimiento:** | Validar cuenta |
| **Características:** | Permite validar la cuenta de un usuario a través de un código que se envía a su correo electrónico proporcionado |
| **Descripción del requerimiento:** | Permite la validación de una cuenta de jugador para su posterior uso |
| **Requerimiento NO funcional:** | Usabilidad, seguridad |
| **Prioridad del requerimiento:**  Alta | |

|  |  |
| --- | --- |
| **Identificación del requerimiento:** | RF03 |
| **Nombre del Requerimiento:** | Iniciar sesión |
| **Características:** | Los jugadores deberán poder iniciar su sesión en el sistema |
| **Descripción del requerimiento:** | Permite a un usuario entrar a e el sistema con sus credenciales |
| **Requerimiento NO funcional:** | Usabilidad, seguridad |
| **Prioridad del requerimiento:**  Alta | |

|  |  |
| --- | --- |
| **Identificación del requerimiento:** | RF04 |
| **Nombre del Requerimiento:** | Jugar campaña |
| **Características:** | Los usuarios podrán iniciar un nuevo juego de la campaña o continuar su progreso en donde lo dejaron la ultima vez |
| **Descripción del requerimiento:** | Modificar presupuesto: Permite a un usuario de tipo organizador modificar el plan de gastos (presupuesto) de un evento |
| **Requerimiento NO funcional:** | Usabilidad |
| **Prioridad del requerimiento:**  Alta | |

|  |  |
| --- | --- |
| **Identificación del requerimiento:** | RF05 |
| **Nombre del Requerimiento:** | Configurar opciones |
| **Características:** | Los usuarios deben de poder personalizar sus opciones de idioma y de ficha para la campaña |
| **Descripción del requerimiento:** | Permite que lo usuarios modifiquen el idioma de su interfaz, y que seleccionen una ficha para jugar la campaña |
| **Requerimiento NO funcional:** | Usabilidad |
| **Prioridad del requerimiento:**  Alta | |

|  |  |
| --- | --- |
| **Identificación del requerimiento:** | RF06 |
| **Nombre del Requerimiento:** | Entrar al lobby |
| **Características:** | Los usuarios deberán poder acceder a el lobby de una partida multijugador |
| **Descripción del requerimiento:** | Permite entrar al lobby del juego ya sea creando una nueva partida o uniéndose a una |
| **Requerimiento NO funcional:** | Usabilidad |
| **Prioridad del requerimiento:**  Alta | |

|  |  |
| --- | --- |
| **Identificación del requerimiento:** | RF07 |
| **Nombre del Requerimiento:** | Jugar partida |
| **Características:** | Los usuarios deberán poder entrar a una partida y jugarla |
| **Descripción del requerimiento:** | Permite a un usuario participar en una partida multijugador |
| **Requerimiento NO funcional:** | Usabilidad |
| **Prioridad del requerimiento:**  Alta | |

|  |  |
| --- | --- |
| **Identificación del requerimiento:** | RF08 |
| **Nombre del Requerimiento:** | Mandar mensaje |
| **Características:** | Los usuarios deben poder mandar mensajes durante su estancia en el lobby o durante una partida multijugador |
| **Descripción del requerimiento:** | Ver eventos: permite a un usuario de tipo asistente revisar cuales son los próximos eventos que se realizaran. |
| **Requerimiento NO funcional:** | Usabilidad |
| **Prioridad del requerimiento:**  Alta | |

## Requisitos comunes de las interfaces

### Interfaces de usuario

La interfaz tiene como objetivo ser lo mas amigable con el usuario y ser fácil de usar y de aprender, por lo que se usaran diseños simples y entendibles que permitan la correcta facilidad de uso para los usuarios, la combinación de los colores escogida para el desarrollo del sistema es: Azul, verde y rojo.

### Interfaces de hardware

Será necesario disponer de equipos de cómputos en perfecto estado con las siguientes características:

* Adaptadores de red.
* Procesador de 1.66GHz o superior.
* Memoria mínima de 256Mb.
* Mouse.
* Teclado.

### Interfaces de software

* Windows 10
* SQL Server

## Requisitos funcionales

Registrarse

Validar cuenta

Iniciar sesión

Jugar campaña

Configurar opciones

Entrar al lobby

Jugar partida

Mandar mensaje

## Requisitos no funcionales

### Seguridad

La seguridad en el juego se implementará usando la técnica de cifrado de sha256 para guardar las contraseñas en la base de datos de manera segura, además de que se validaran todas las entradas para evitar inyecciones de código y se manejara un sistema de Inicio de sesión para evitar que se alteren los datos de un usuario desde un agente externo no autorizado.

### Usabilidad

La usabilidad se implementara siguiendo las 10 heurísticas de Jakob Nielsen para garantizar la usabilidad del producto y aparte se desarrollaran revisiones de usabilidad a lo largo del proyecto y para la entrega final, se aplicaran pruebas de usabilidad a usuarios representativos de los usuarios finales, con el objetivo de asegurar la calidad aplicando cuestionarios y la escala de sus para así asegurarse de que la interfaz cumple satisfactoriamente su propósito de ser entendible, fácil de aprender, fácil de recordar, y agradable para el usuario

# Modelo de dominio



# Dinámica del juego

La dinámica del juego “Serpientes Escaleras Y Dragones” se divide en dos tipos: juego en solitario (campaña) y para varios jugadores (Multijugador)

Para el caso de el juego en solitario, se usará un oponente enemigo, se usará un dado para que pueda avanzar de entre 1 a 6 casillas, todos los niveles del juego presentaran una imagen con texto para explicar la historia y contaran con un tablero diferente de 100 casillas,

Además de esto se presentarán los siguientes eventos especiales:

Serpientes: las serpientes se presentarán 3 en cada tablero, y regresarán el número de casillas que indique el final de su cuerpo

Escaleras: las escaleras cumplirán con el rol contrario de las serpientes, también serán tres en cada tablero y subirán el numero de casillas que indique la figura

Dragones: los dragones se presentarán también de 3 en cada tablero y cumplirán ambas funciones antes presentadas, una vez que funcionen como escalera (subiendo al jugador) la próxima vez que se activen, funcionarán como serpiente (bajando al jugador)

Una vez que el jugador gane un nivel, el juego lo llevará al siguiente, Hasta que termine con todos los niveles, en caso de que pierda en alguno, el sistema lo hará comenzar una vez más el nivel.

Para el juego multijugador, se aplicarán casi los mismos criterios, solo que se necesitara 4 jugadores conectados a la vez para poder iniciar una partida, y además se agregara una funcionalidad extra:

Muerte súbita: se activará en el lobby, y se determinará por turnos del jugador, por lo que una vez que algún jugador alcance el numero de turnos establecido, se activar la muerte súbita, que cambiará el tablero por uno lleno de dragones.

# Estándar de codificación

# Estándar de codificación en C#

Las convenciones de codificación tienen los objetivos siguientes:

Crean una apariencia coherente en el código, para que los lectores puedan centrarse en el contenido, no en el diseño.

Permiten a los lectores comprender el código más rápidamente al hacer suposiciones basadas en la experiencia anterior.

Facilitan la copia, el cambio y el mantenimiento del código.

Muestran los procedimientos recomendados de C#.

Microsoft usa las instrucciones de este tema para desarrollar ejemplos y documentación.

## Convenciones de nomenclatura

En ejemplos breves que no incluyen [directivas using](https://docs.microsoft.com/es-es/dotnet/csharp/language-reference/keywords/using-directive), use calificaciones de espacio de nombres. Si sabe que un espacio de nombres se importa en un proyecto de forma predeterminada, no es necesario completar los nombres de ese espacio de nombres. Los nombres completos pueden partirse después de un punto (.) si son demasiado largos para una sola línea, como se muestra en el ejemplo siguiente.

var currentPerformanceCounterCategory = new System.Diagnostics.

PerformanceCounterCategory();

No es necesario cambiar los nombres de objetos que se crearon con las herramientas del diseñador de Visual Studio para que se ajusten a otras directrices.

Convenciones de diseñoLayout Conventions

Un buen diseño utiliza un formato que destaque la estructura del código y haga que el código sea más fácil de leer. Las muestras y ejemplos de Microsoft cumplen las convenciones siguientes:

Utilice la configuración del Editor de código predeterminada (sangría automática, sangrías de 4 caracteres, tabulaciones guardadas como espacios). Para obtener más información, vea [Opciones, editor de texto, C#, formato](https://docs.microsoft.com/es-es/visualstudio/ide/reference/options-text-editor-csharp-formatting).

Escriba solo una instrucción por línea.

Escriba solo una declaración por línea.

Si a las líneas de continuación no se les aplica sangría automáticamente, hágalo con una tabulación (cuatro espacios).

Agregue al menos una línea en blanco entre las definiciones de método y las de propiedad.

Utilice paréntesis para que las cláusulas de una expresión sean evidentes, como se muestra en el código siguiente.

if ((val1 > val2) && (val1 > val3))

{

// Take appropriate action.

}

## Convenciones de los comentarios

Coloque el comentario en una línea independiente, no al final de una línea de código.

Comience el texto del comentario con una letra mayúscula.

Finalice el texto del comentario con un punto.

Inserte un espacio entre el delimitador de comentario (//) y el texto del comentario, como se muestra en el ejemplo siguiente.

// The following declaration creates a query. It does not run

// the query.

No cree bloques con formato de asteriscos alrededor de comentarios.

Convenciones de

En las secciones siguientes se describen las prácticas que sigue el equipo C# para preparar las muestras y ejemplos de código.

## String (Tipo de datos)

Use [interpolación de cadenas](https://docs.microsoft.com/es-es/dotnet/csharp/language-reference/tokens/interpolated) para concatenar cadenas cortas, como se muestra en el código siguiente.

string displayName = $"{nameList[n].LastName}, {nameList[n].FirstName}";

Para anexar cadenas en bucles, especialmente cuando se trabaja con grandes cantidades de texto, utilice un objeto [StringBuilder](https://docs.microsoft.com/es-es/dotnet/api/system.text.stringbuilder).

var phrase = "lalalalalalalalalalalalalalalalalalalalalalalalalalalalalala";

var manyPhrases = new StringBuilder();

for (var i = 0; i < 10000; i++)

{

manyPhrases.Append(phrase);

}

//Console.WriteLine("tra" + manyPhrases);

Variables locales con asignación implícita de tipos

Use [tipos implícitos](https://docs.microsoft.com/es-es/dotnet/csharp/programming-guide/classes-and-structs/implicitly-typed-local-variables) para las variables locales cuando el tipo de la variable sea obvio desde el lado derecho de la asignación, o cuando el tipo exacto no sea importante.

// When the type of a variable is clear from the context, use var

// in the declaration.

var var1 = "This is clearly a string.";

var var2 = 27;

var var3 = Convert.ToInt32(Console.ReadLine());

No use [var](https://docs.microsoft.com/es-es/dotnet/csharp/language-reference/keywords/var) cuando el tipo no sea evidente desde el lado derecho de la asignación.

// When the type of a variable is not clear from the context, use an

// explicit type.

int var4 = ExampleClass.ResultSoFar();

No confíe en el nombre de variable para especificar el tipo de la variable. Puede no ser correcto.

// Naming the following variable inputInt is misleading.

// It is a string.

var inputInt = Console.ReadLine();

Console.WriteLine(inputInt);

Evite el uso de var en lugar de [dynamic](https://docs.microsoft.com/es-es/dotnet/csharp/language-reference/builtin-types/reference-types).

Use tipos implícitos para determinar el tipo de la variable de bucle en bucles [for](https://docs.microsoft.com/es-es/dotnet/csharp/language-reference/keywords/for) y [foreach](https://docs.microsoft.com/es-es/dotnet/csharp/language-reference/keywords/foreach-in).

En el ejemplo siguiente se usan tipos implícitos en una instrucción for.

var phrase = "lalalalalalalalalalalalalalalalalalalalalalalalalalalalalala";

var manyPhrases = new StringBuilder();

for (var i = 0; i < 10000; i++)

{

manyPhrases.Append(phrase);

}

//Console.WriteLine("tra" + manyPhrases);

En el ejemplo siguiente se usan tipos implícitos en una instrucción foreach.

foreach (var ch in laugh)

{

if (ch == 'h')

Console.Write("H");

else

Console.Write(ch);

}

Console.WriteLine();

## Tipo de datos sin signo

En general, utilice int en lugar de tipos sin signo. El uso de int es común en todo C#, y es más fácil interactuar con otras bibliotecas cuando se usa int.

Matrices

Utilice sintaxis concisa para inicializar las matrices en la línea de declaración.

// Preferred syntax. Note that you cannot use var here instead of string[].

string[] vowels1 = { "a", "e", "i", "o", "u" };

// If you use explicit instantiation, you can use var.

var vowels2 = new string[] { "a", "e", "i", "o", "u" };

// If you specify an array size, you must initialize the elements one at a time.

var vowels3 = new string[5];

vowels3[0] = "a";

vowels3[1] = "e";

// And so on.

## Delegados

Utilice sintaxis concisa para crear instancias de un tipo de delegado.

// First, in class Program, define the delegate type and a method that

// has a matching signature.

// Define the type.

public delegate void Del(string message);

// Define a method that has a matching signature.

public static void DelMethod(string str)

{

Console.WriteLine("DelMethod argument: {0}", str);

}

// In the Main method, create an instance of Del.

// Preferred: Create an instance of Del by using condensed syntax.

Del exampleDel2 = DelMethod;

// The following declaration uses the full syntax.

Del exampleDel1 = new Del(DelMethod);

Instrucciones try-catch y using en el control de excepciones.

Use una instrucción [try-catch](https://docs.microsoft.com/es-es/dotnet/csharp/language-reference/keywords/try-catch) en la mayoría de casos de control de excepciones.

static string GetValueFromArray(string[] array, int index)

{

try

{

return array[index];

}

catch (System.IndexOutOfRangeException ex)

{

Console.WriteLine("Index is out of range: {0}", index);

throw;

}

}

Simplifique el código mediante la [instrucción using](https://docs.microsoft.com/es-es/dotnet/csharp/language-reference/keywords/using-statement) de C#. Si tiene una instrucción [try-finally](https://docs.microsoft.com/es-es/dotnet/csharp/language-reference/keywords/try-finally) en la que el único código del bloque finally es una llamada al método [Dispose](https://docs.microsoft.com/es-es/dotnet/api/system.idisposable.dispose), use en su lugar una instrucción using.

// This try-finally statement only calls Dispose in the finally block.

Font font1 = new Font("Arial", 10.0f);

try

{

byte charset = font1.GdiCharSet;

}

finally

{

if (font1 != null)

{

((IDisposable)font1).Dispose();

}

}

// You can do the same thing with a using statement.

using (Font font2 = new Font("Arial", 10.0f))

{

byte charset = font2.GdiCharSet;

}

## Operadores && y ||

Para evitar excepciones y aumentar el rendimiento omitiendo las comparaciones innecesarias, use [&&](https://docs.microsoft.com/es-es/dotnet/csharp/language-reference/operators/boolean-logical-operators#conditional-logical-and-operator-) en lugar de [&](https://docs.microsoft.com/es-es/dotnet/csharp/language-reference/operators/boolean-logical-operators#logical-and-operator-) y [||](https://docs.microsoft.com/es-es/dotnet/csharp/language-reference/operators/boolean-logical-operators#conditional-logical-or-operator-) en lugar de [|](https://docs.microsoft.com/es-es/dotnet/csharp/language-reference/operators/boolean-logical-operators#logical-or-operator-) cuando realice comparaciones, como se muestra en el ejemplo siguiente.

Console.Write("Enter a dividend: ");

var dividend = Convert.ToInt32(Console.ReadLine());

Console.Write("Enter a divisor: ");

var divisor = Convert.ToInt32(Console.ReadLine());

// If the divisor is 0, the second clause in the following condition

// causes a run-time error. The && operator short circuits when the

// first expression is false. That is, it does not evaluate the

// second expression. The & operator evaluates both, and causes

// a run-time error when divisor is 0.

if ((divisor != 0) && (dividend / divisor > 0))

{

Console.WriteLine("Quotient: {0}", dividend / divisor);

}

else

{

Console.WriteLine("Attempted division by 0 ends up here.");

}

## New (Operador)

Utilice la forma concisa de la creación de instancias de objeto con tipos implícitos, como se muestra en la siguiente declaración.

var instance1 = new ExampleClass();

La línea anterior es equivalente a la siguiente declaración.

ExampleClass instance2 = new ExampleClass();

Utilice inicializadores de objeto para simplificar la creación de objetos.

// Object initializer.

var instance3 = new ExampleClass { Name = "Desktop", ID = 37414,

Location = "Redmond", Age = 2.3 };

// Default constructor and assignment statements.

var instance4 = new ExampleClass();

instance4.Name = "Desktop";

instance4.ID = 37414;

instance4.Location = "Redmond";

instance4.Age = 2.3;

Control de eventos

Si va a definir un controlador de eventos que no es necesario quitar más tarde, utilice una expresión lambda.

public Form2()

{

// You can use a lambda expression to define an event handler.

this.Click += (s, e) =>

{

MessageBox.Show(

((MouseEventArgs)e).Location.ToString());

};

}

// Using a lambda expression shortens the following traditional definition.

public Form1()

{

this.Click += new EventHandler(Form1\_Click);

}

void Form1\_Click(object sender, EventArgs e)

{

MessageBox.Show(((MouseEventArgs)e).Location.ToString());

}

Miembros estáticos

Llame a miembros [estáticos](https://docs.microsoft.com/es-es/dotnet/csharp/language-reference/keywords/static) con el nombre de clase: ClassName.StaticMember.. Esta práctica hace que el código sea más legible al clarificar el acceso estático. No califique un miembro estático definido en una clase base con el nombre de una clase derivada. Mientras el código se compila, su legibilidad se presta a confusión, y puede interrumpirse en el futuro si se agrega a un miembro estático con el mismo nombre a la clase derivada.

## Consultas LINQ

Utilice nombres descriptivos para las variables de consulta. En el ejemplo siguiente, se utiliza seattleCustomers para los clientes que se encuentran en Seattle.

var seattleCustomers = from customer in customers

where customer.City == "Seattle"

select customer.Name;

Utilice alias para asegurarse de que los nombres de propiedad de tipos anónimos se escriben correctamente con mayúscula o minúscula, usando para ello la grafía Pascal.

var localDistributors =

from customer in customers

join distributor in distributors on customer.City equals distributor.City

select new { Customer = customer, Distributor = distributor };

Cambie el nombre de las propiedades cuando puedan ser ambiguos en el resultado. Por ejemplo, si la consulta devuelve un nombre de cliente y un identificador de distribuidor, en lugar de dejarlos como Name e ID en el resultado, cambie su nombre para aclarar que Name es el nombre de un cliente e ID es el identificador de un distribuidor.

var localDistributors2 =

from customer in customers

join distributor in distributors on customer.City equals distributor.City

select new { CustomerName = customer.Name, DistributorID = distributor.ID };

Utilice tipos implícitos en la declaración de variables de consulta y variables de intervalo.

var seattleCustomers = from customer in customers

where customer.City == "Seattle"

select customer.Name;

Alinee las cláusulas de consulta bajo la cláusula [from](https://docs.microsoft.com/es-es/dotnet/csharp/language-reference/keywords/from-clause), como se muestra en los ejemplos anteriores.

Use cláusulas [where](https://docs.microsoft.com/es-es/dotnet/csharp/language-reference/keywords/where-clause) antes de otras cláusulas de consulta para asegurarse de que las cláusulas de consulta posteriores operan en un conjunto de datos reducido y filtrado.

var seattleCustomers2 = from customer in customers

where customer.City == "Seattle"

orderby customer.Name

select customer;

Use varias cláusulas from en lugar de una cláusula [join](https://docs.microsoft.com/es-es/dotnet/csharp/language-reference/keywords/join-clause) para obtener acceso a colecciones internas. Por ejemplo, una colección de objetos Student podría contener cada uno un conjunto de resultados de exámenes. Cuando se ejecuta la siguiente consulta, devuelve cada resultado superior a 90, además del apellido del alumno que recibió la puntuación.

// Use a compound from to access the inner sequence within each element.

var scoreQuery = from student in students

from score in student.Scores

where score > 90

select new { Last = student.LastName, score };

# Modelos de diseño

## Diagrama de secuencia



## Diagrama de despliegue





# Justificación de los frameworks y tecnología

Utilizamos el framework de .net para hacer un desarrollo de aplicación rápido y eficiente, ya que .net ofrece una gran variedad de soluciones para el desarrollo de aplicaciones

Framework utilizado durante el desarrollo: Entity framework para el manejo de los datos en la base de datos.

IDE utilizado para el desarrollo del sistema: Visual Studio

Lenguaje de programación utilizado: C#

Lenguaje de formato para la interfaz utilizado: XAML

## justificación

Primero que nada se seleccionó el framwork de .net por que integra muchas opciones y características para el desarrollo de aplicaciones y todas las usa de manera conjunta y sencilla, por lo que nuestras demás selecciones se basaron en tecnologías que fueran implementadas bajo el framework de .net

Para el desarrollo se necesitaba utilizar un IDE que brindara todas las características necesarias para facilitar nuestro proceso de desarrollo, por lo que se escogió visual studio, por que brinda un editor de texto, fácil de aprender y manejar, soporta el lenguaje que seleccionamos para el desarrollo que fue C#, esta integrado con un manejo sencillo de Entity framework que fue el framework para el manejo de datos que utilizamos y además permite la creación de las interfaces graficas por medio de menús controladores y código en XAML,

Para la elección de framwork de persistencia nos basamos en buscar un framework que nos permitiera un aprendizaje sencillo y un manejo efectivo y rápido de los datos, y para las soluciones de .net el mejor es Entity framework que nos pareció una solución sencilla y efectiva de usar para guardar los datos persistentes.

Para elegir un lenguaje de programación, se buscó uno que estuviera integrado en el framework de .net y que fuera relativamente parecido a java, que es el lenguaje que más manejábamos y C# tiene una similitud muy grande con java, además es un lenguaje de programación muy fácil de aprender y muy fácil de leer

# Pruebas

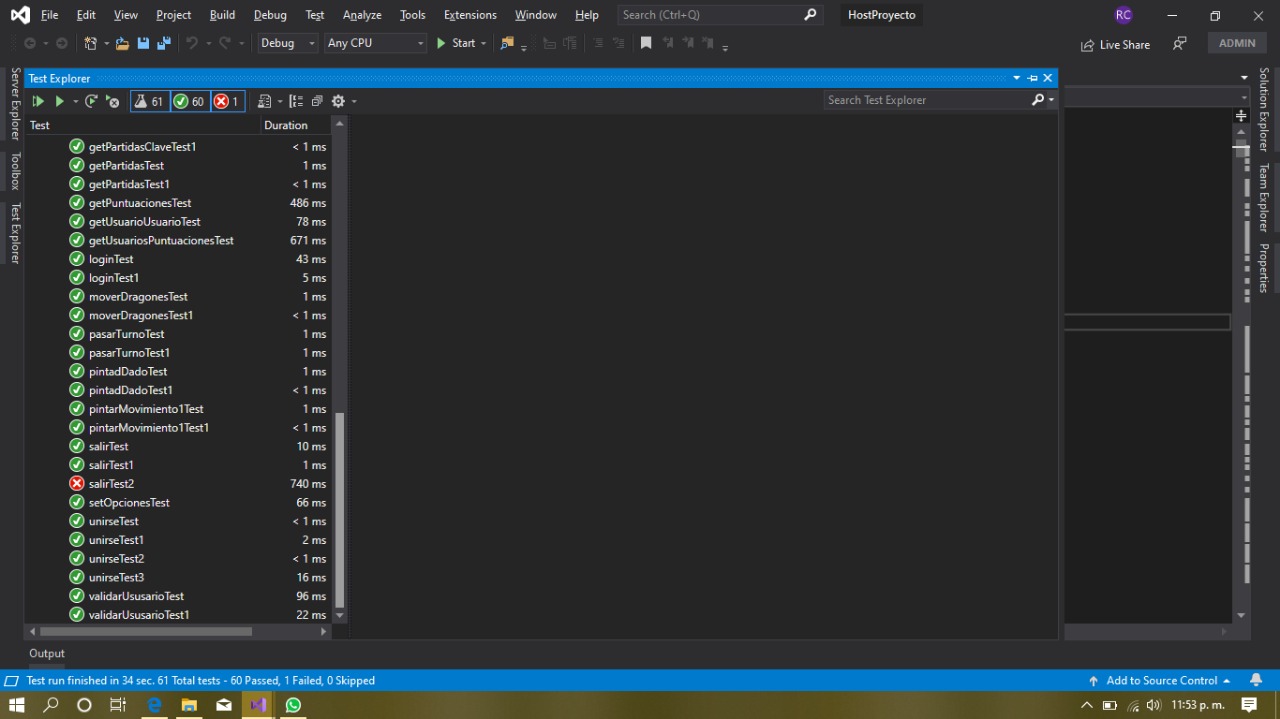
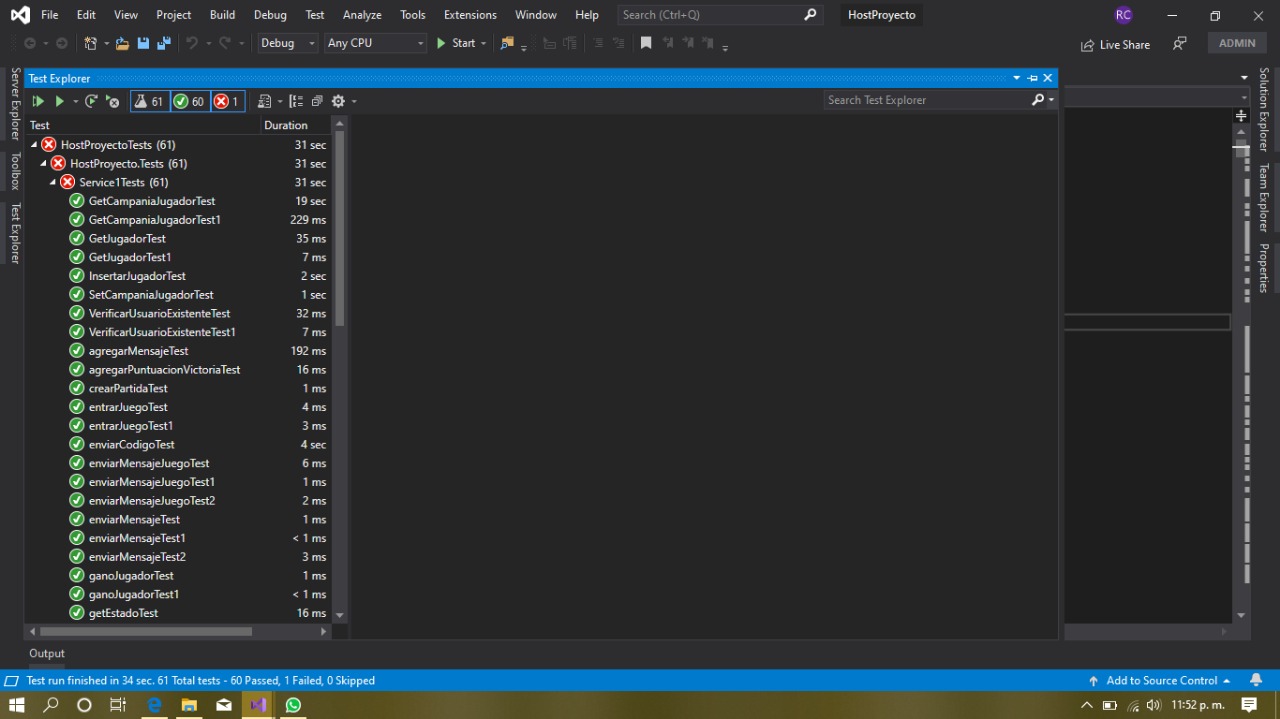
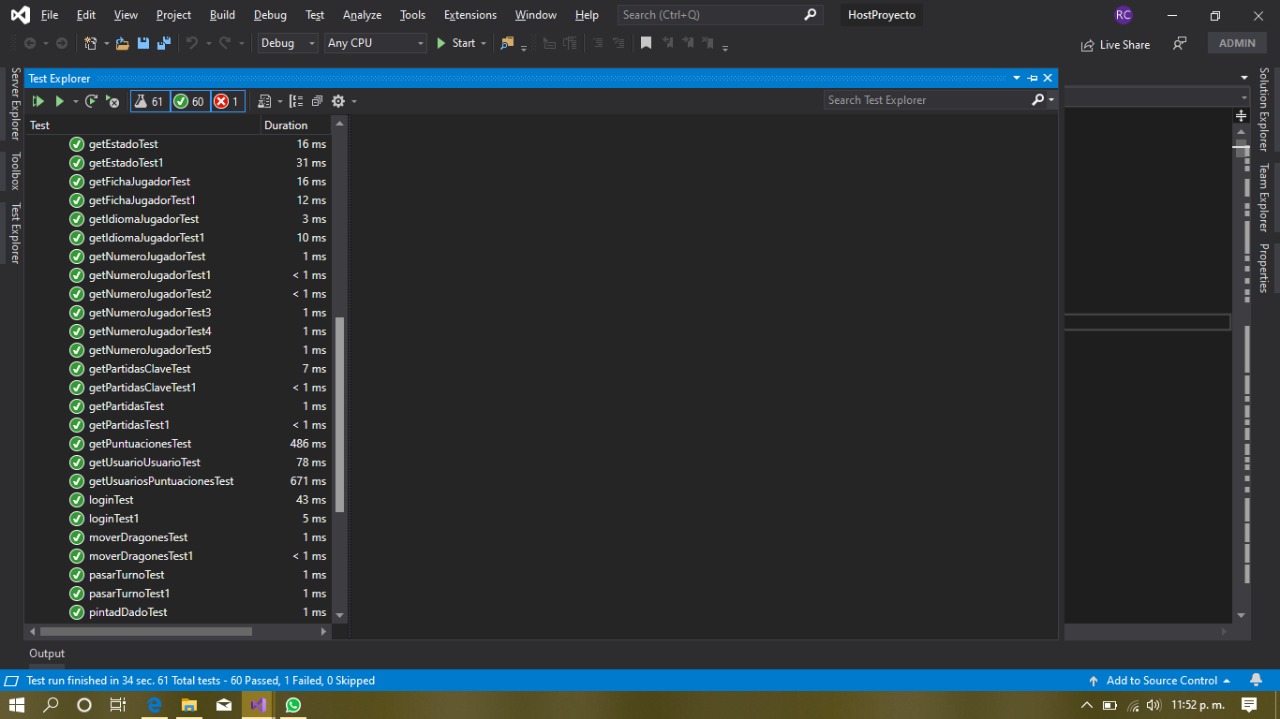
## Pruebas del sistema

Para la realización de las pruebas del sistema decidimos hacer pruebas unitarias para probar por separado cada método del sistema, se probaron un total de 3 flujos para cada método y los resultados fueron los siguientes:

### Resultados

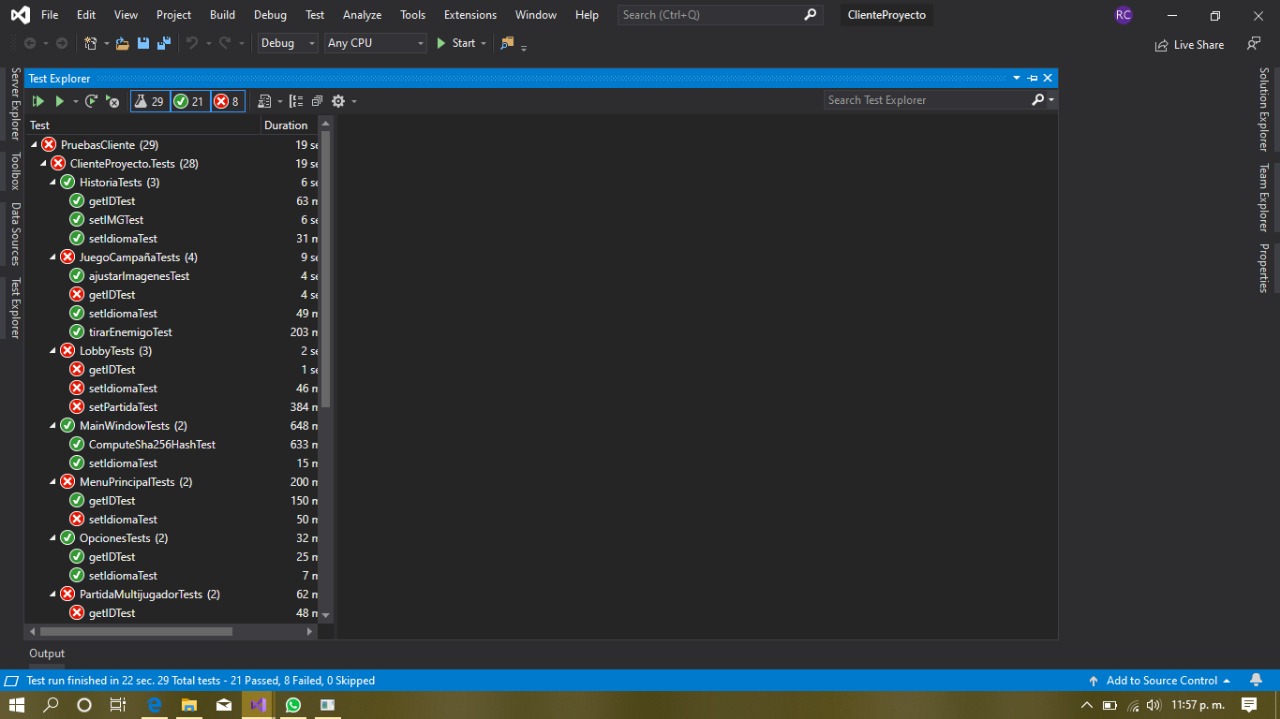
### Resultados en Host

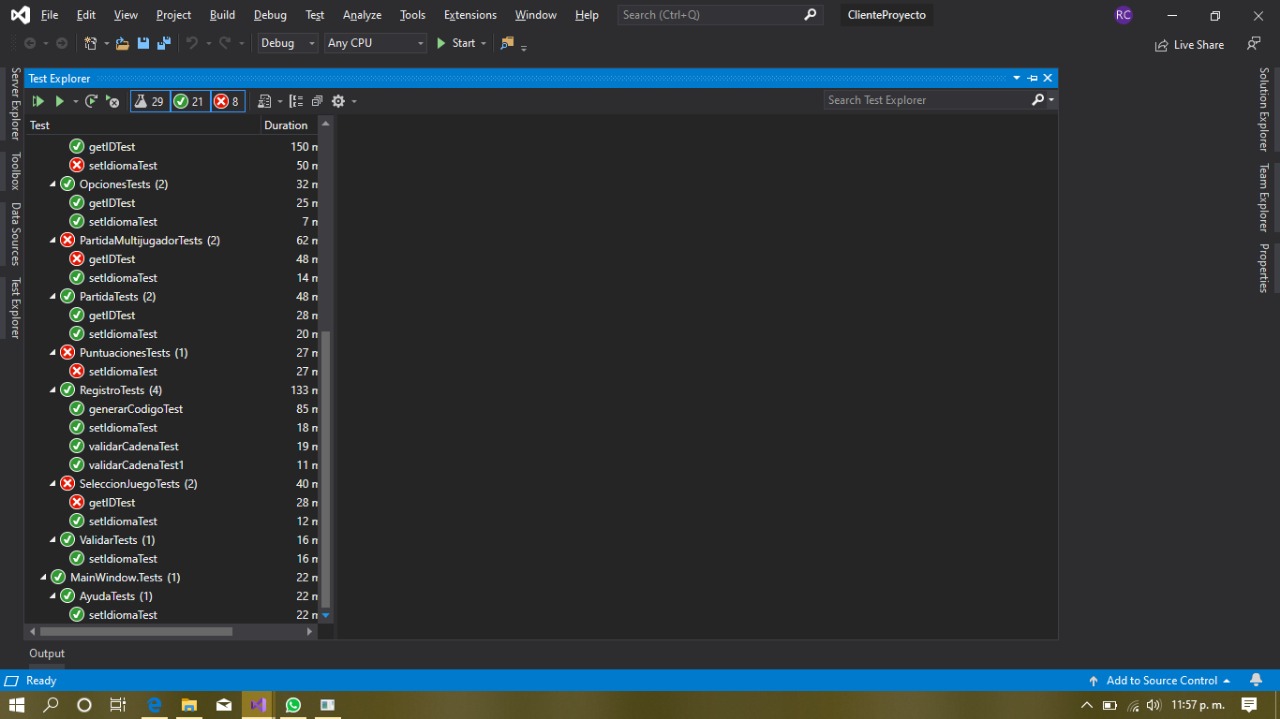
Para las pruebas en el host todas nuestras pruebas pasaron a excepción de una que no supimos por que no pasaba por que en la ejecución del programa si se ejecutaba correctamente

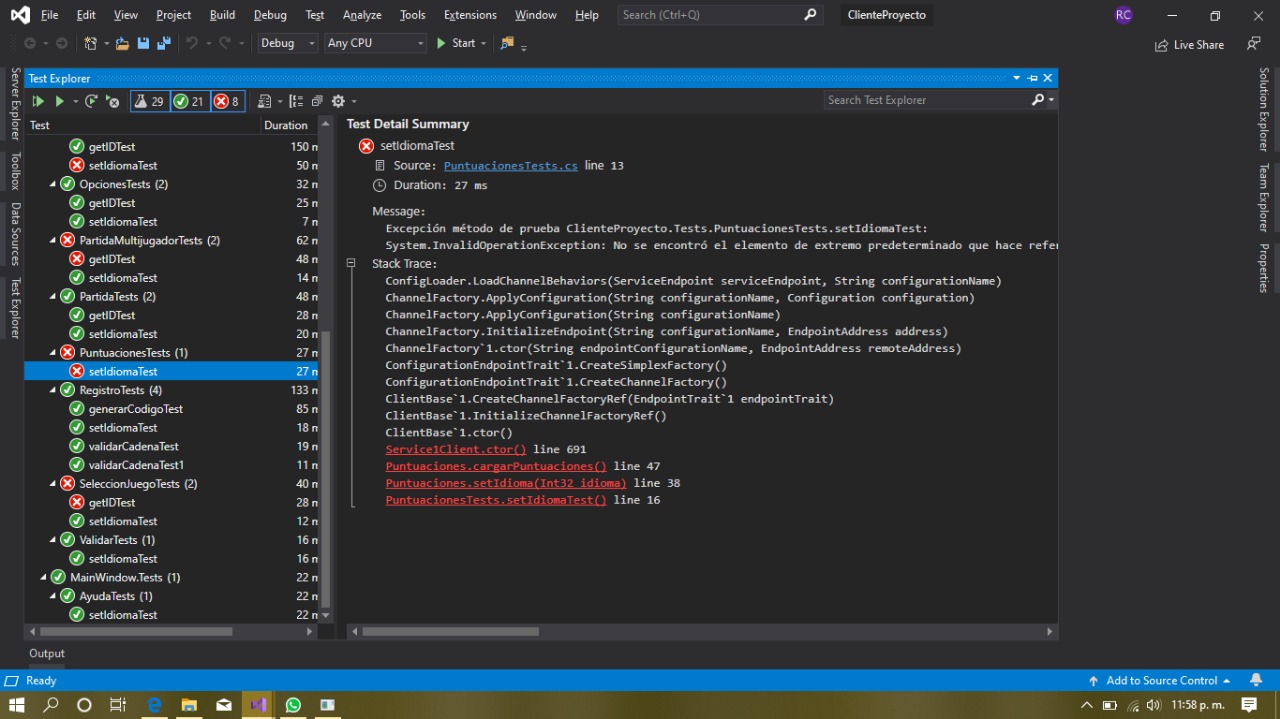


### Pruebas de cliente

En las pruebas del cliente nos fue mal, ya que por alguna razón a la hora de ejecutar las pruebas todas fallan en la conexión, las que no implicaron cambios en los métodos del host, todas pasaron pero a las que se les aplicaron cambios al código en el host, no pasó ninguna





Esta es la excepción que nos salió en todas las pruebas ejecutadas en el cliente que involucraban cambios en el host.

## Pruebas de usabilidad

### Introducción

### Objetivo del documento

Este documento es un reporte de evaluación de usabilidad, por lo que servirá para exponer los resultados de las pruebas aplicadas, permitiendo que se puedan conocer los defectos y virtudes del sistema desarrollado en cuanto a este atributo de calidad, dando pie a futuros cambios y mejoras en partes y temas específicos.

### Descripción del prototipo de la interfaz

La interfaz gráfica de usuario fue desarrollada teniendo en cuenta las heurísticas de diseño de usabilidad, la interfaz está dividida en 9 pantallas las cuales se describen a continuación:

1. *inicio de sesión.*

La pantalla de inicio de sesión contiene los botones de “Iniciar sesión”, “Registrarse” y “Validar cuenta” y un formulario para un usuario y una contraseña que le permitirán al usuario iniciar su sesión.

Para el correcto flujo de esta pantalla, el usuario deberá introducir su usuario y su contraseña y a continuación presionar el botón “Iniciar sesión”

Aparte de esto, la ventana incluye una selección de idioma que permite que la ventana cambie su idioma entre inglés y español

1. *registrarse*

La pantalla de registro incluye los botones de “Aceptar” y “Cancelar” y un formulario para introducir los datos de una cuenta, los cuales incluyen: nombre completo, usuario, contraseña y correo electrónico, en los campos está limitada la entrada de datos solo para su tamaño posible en el sistema.

Para el correcto flujo de esta ventana el usuario tiene que llenar todos los datos solicitados con el formato solicitado y a continuación presionar el botón “Aceptar”

1. *validar cuenta*

La pantalla de validar cuenta contiene un campo para escribir el código que el sistema manda el correo del usuario y un botón “aceptar” y uno de “Cancelar”.

Para el correcto flujo de esta pantalla, el usuario deberá introducir el código y presionar el botón “Aceptar” para que el sistema mande un mensaje de éxito.

1. *menú principal*

Para esta pantalla se le ofrecen a el usuario varios botones para ejecutar diferentes acciones, los botones que se le ofrecen al usuario son: “Campaña”,” Multijugador”,” Opciones”,” Acerca de”,” Ayuda”,” Mejores puntuaciones” y” salir”, en esta pantalla se muestra el logo del juego en la parte superior

Para el correcto flujo de esta ventana el usuario presiona un botón para continuar su actividad y el sistema lo lleva a la ventana del botón que selecciono

1. *selección campaña*

Para esta ventana el sistema despliega una imagen y tres botones, dos principales de

“Nueva campaña” y “Continuar campaña” y uno secundario de “salir”

Para el correcto uso de esta ventana, el usuario selecciona una de las opciones que el prefiera. En caso de que no tenga progreso guardado, el sistema le bloqueara a el usuario el botón de “Continuar campaña”

1. *campaña*

Para esta ventana el sistema despliega el tablero del juego, las fichas del jugador y el enemigo y los botones para “Tirar” y para “Salir”.

Para el correcto funcionamiento de esta ventana, el usuario presionara el botón “Tirar” hasta ganar o perder. Una vez terminado el juego el sistema enviara a el usuario a la siguiente ventana.

1. *selección partida*

Para esta ventana el sistema muestra las partidas que están disponibles actualmente en una lista con un botón “Unirse”. Además de eso, muestra un campo para escribir el nombre de una nueva partida junto con el botón de “Crear partida” además del botón de

“Salir”.

1. *lobby*

Para esta pantalla el sistema muestra a los jugadores actuales en una lista, un cuadro de chat, una imagen con la ficha con la que jugara el jugador y en el caso del líder las opciones para personalizar su partida, como los botones, y turnos ara la muerte súbita, así como el botón de “Comenzar partida”.

Para el flujo correcto de esta pantalla el usuario entra a la partida, espera a los demás jugadores, selecciona las opciones de su partida y presiona el botón “Comenzar partida”

1. *partida multijugador*

Para esta ventana el sistema muestra el tablero con las fichas de los jugadores, junto con el chat y con el botón de “Tirar” que se activa para el jugador de quien es turno

Para el correcto funcionamiento de esta ventana. Los usuarios presionaran el botón “Tirar” hasta que alguien gane y el sistema los mande a la siguiente ventana

1. *opciones*

En esta pantalla el sistema muestra una opción para cambiar su ficha de campaña y una para cambiar su idioma, además de un botón de “Guardar Cambios” y uno de “Cancelar”.

Para el correcto flujo de esta pantalla, el usuario selecciona las opciones de su preferencia y presiona el botón de “Guardar Cambios”

### Planeación de la prueba de usabilidad

#### Objetivo de la prueba

El objetivo de la prueba de usabilidad es determinar métricas para estudiar los resultados de pruebas aplicadas a usuarios finales para así permitirnos conocer cuáles son las deficiencias del diseño del sistema y así poder aplicar mejoras y correcciones en futuras actualizaciones del sistema. Como objetivo final y más importante. Las pruebas tienen el objetivo de determinar si el producto cumple con la calidad suficiente respecto a la usabilidad para el despliegue final.

### Características de los participantes

Los participantes elegidos para las pruebas, deberán ser personas entre las siguientes características:

-Edad entre 16 y 28 años

-Gusto por los videojuegos

-Habilidades intermedias en computación

-gusto por los juegos de mesa

-conocimiento del juego “Serpientes y escaleras”

-tendencia competitiva

### Describir método tipo de prueba

Las técnicas que se utilizarán para la prueba serán: Observando a los usuarios y Preguntando a los usuarios.

La técnica observando a los usuarios se utilizará mientras está completando la lista de tareas proporcionada. Los observadores serán los integrantes del equipo durante toda la iteración.

La técnica preguntando a los usuarios se realizará después de completada la lista de tareas mediante un cuestionario de satisfacción.

### Entorno de prueba, equipo y logística

La prueba de usabilidad se realizará en un salón de clases cuidando de que no interfiera mucho el ruido externo u otros factores que puedan ser distracciones para el usuario que ayudará con la evaluación. El lugar debe estar suficientemente iluminado, cuidando de no exceder en este caso.

La prueba se realizará en dos equipos de cómputo marca HP, los cuales debe tener el juego Serpientes, Escaleras y Dragones previamente instalado; las características de los equipos son las siguientes:

Sistema operativo: Windows 10 de 64 bits.

Memoria RAM: AMD Radeon con 12 gb e Intel inside con 4 gb.}

Ambos deberán de contar con una cámara web y un programa grabador de pantalla para el registro de las acciones en el sistema.

La logística que se seguirá para la realización de estas pruebas será la siguiente:

1. La participación de los usuarios invitados será de máximo dos a la vez.
2. Se les dará la bienvenida y se les explicará en lo que consiste la prueba, las acciones que realizará y los métodos de evaluación.
3. Se les mostrará y pedirá que firmen un formato de consentimiento (en el caso que ellos se acepten las condiciones), después se les solicitará sus datos mediante un formulario.
4. Se le dará al usuario un tiempo máximo de 5 minutos para que explore la interfaz del juego.
5. Se le dará la lista de tareas y comenzará a realizarlas mientras el observador toma las notas pertinentes.
6. Se aplicará el cuestionario SUS.
7. El equipo agradece por su participación y despide al usuario.

### Datos para recolectar

Mediante la aplicación de la evaluación, se busca obtener información sobre los siguientes puntos.

* Nivel de satisfacción del usuario con la utilización del sistema.
* Puntos de mejora sobre las interfaces.
* El tiempo tomado para la realización de cada tarea.
* El comportamiento del usuario frente a la interfaz.
* La cantidad de errores que pueda llegar a cometer en un tiempo determinado.

### Descripción de las herramientas de medición Cuestionario SUS (System Usability Scale)

Elaborado por Brooke en 1986 con 10 ítems, de los cuales 5 son positivos y 5 son negativos. Fue de las primeras que surgieron para evaluar la usabilidad de una interfaz. La creación de sus ítems surgió a partir de la definición de usabilidad en el ISO 9241-11:

“el grado en el que un producto puede ser utilizado por usuarios específicos para lograr metas determinadas con efectividad, eficiencia y satisfacción en un contexto particular de uso”

### Aplicando las pruebas de usabilidad

A continuación, se describe los pasos seguidos para la realización de las pruebas

### Descripción de cómo se llevó a cabo la prueba

Cada sesión de prueba se realizó individualmente, se puso al usuario frente al sistema a interactuar con el de forma que las distracciones no fueran demasiadas para el uso del sistema. Se realizaron tres sesiones las cuales duraron alrededor de 30 minutos cada una. Además, el orden de participación de los usuarios se realizaría conforme al orden de llegada al punto de la prueba.

Antes de iniciar las sesiones, nos aseguramos de que contáramos con todo lo necesario para iniciar con las pruebas: el pc con “Serpientes, escaleras y dragones” y un capturador de pantalla previamente instalados, disponer del espacio apropiadamente puesto que se tuvieron que mover algunas cosas para poder ocuparlo y tener los presentes de agradecimiento listos para los usuarios; una vez hecho eso, se esperó y recibió a los usuarios. Solo hubo un contratiempo: los usuarios llegaron aproximadamente 10 minutos más tarde de lo acordado, aunque esto causó un impacto muy menor.

Una vez que inició la prueba se dio lugar a las actividades descritas en el programa: Se les dio la bienvenida, se les explicó el propósito de esa sesión y se les entregó los formatos de consentimiento y datos de usuario para su llenado.

Una vez llenados estos formularios, se les explicó la interfaz del juego y se les otorgó un tiempo de familiarización con la interfaz. Pasados esos 5 minutos, se les entregó el listado de tareas a los usuarios y se les explicó en qué consistían, de manera que quedaran más claras, también se resolvieron algunas dudas que surgieron de los usuarios. A la par de esto, se preparaba el capturador de pantalla para iniciar la grabación de la interacción.

Durante la sesión, el usuario en turno realizó las tareas descritas con anterioridad mientras uno de nosotros, que fungió como moderador, estaba al pendiente de las dudas o preguntas que el usuario pudiera tener. Al mismo tiempo, el otro fungía como observador y realizaba anotaciones sobre las posturas que observaba en el usuario. Cuando el usuario finalizaba se le entregaba el cuestionario SUS para que lo respondiera. Una vez hecho esto, se les agradeció su participación y se les entregó un presente como agradecimiento.

### Descripción de los participantes

Los usuarios que nos apoyaron tenían las siguientes características:

* Edad entre 18 y 22 años
* Habilidades altas en computación
* Gusto por los juegos de mesa
* Conocimiento básico del juego “Serpientes y escaleras”

Las pruebas se realizaron con 3 usuarios con estas características. Los tres, estudiantes de licenciatura en ingeniería de software en la Facultado de Estadística e Informática de la Universidad Veracruzana.

### Resultados de las pruebas de usabilidad

Los resultados obtenidos en esta prueba de usabilidad por parte de los usuarios son los siguientes:

* *Efectividad*

En el caso de la evaluación para la prueba de “Serpientes, escaleras y dragones” los objetivos eran las 5 tareas que se les pidió a los usuarios realizar en el sistema. Los observadores fueron los responsables de hacer este registro, en el que finalmente se obtuvo el siguiente resultado: los tres usuarios lograron completar el 100% de las tareas solicitadas de manera precisa.

* *Eficiencia*

En cuanto a la eficiencia, el equipo de evaluación se basó en la rapidez con la que los objetivos de los usuarios pueden ser completados con exactitud y completitud. Estos resultados comúnmente se miden en términos de tiempo, por lo que el equipo de evaluación estableció un tiempo esperado para completar cada tarea, basándose en el tiempo que a ellos les había tomado cubrir cada una, y dejando 20 segundos de tolerancia, considerando que los usuarios no tendrían tanta experiencia en el sistema como los evaluadores (responsables de la evaluación).}

Se tiene que considerar que la parte de juego campaña (Tarea 4) es difícil de medir ya que depende de cómo se desarrolla el juego ya que se basa en aleatoriedad.

A continuación, en la tabla 3 se muestra el tiempo que le tomó a los usuarios realizar cada tarea, así como el promedio final. Esta información se recopiló del checklist de observación llenado por los observadores.

*Tabla 1 Tiempos de realización de tareas por usuario*

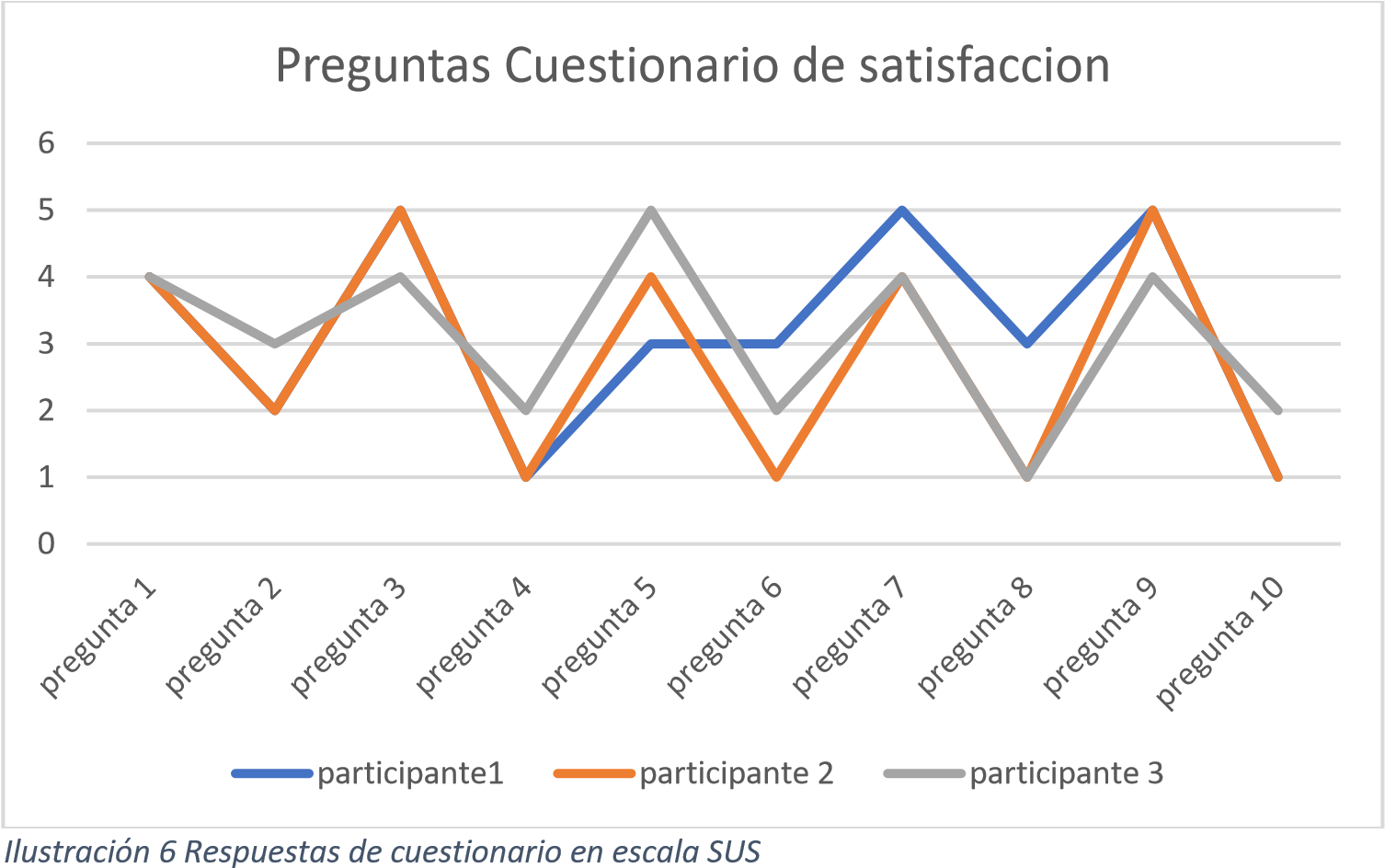
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Tarea | **Tiempo por usuario (segundos)** | | |  | **Tiempo esperado**  **(Segundos)** | **Tiempo promedio final**  **(segundos**  **)** |
| Usuario 1 | Usuario 2 | Usuario 3 |  |
| 1 | 81.44 | 77.21 | 82.16 | 93 | 80.27 |
| 2 | 19.9 | 15.66 | 16.05 | 33 | 17.20 |
| 3 | 35.36 | 32.03 | 30.17 | 58 | 32.52 |
| 4 | 190.48 | 270.58 | 201.51 | 140 | 220.86 |
| 5 | 15.40 | 10.48 | 16.39 |  | 30 | 14.09 |

Los observadores también hicieron anotaciones respecto a cada tarea, por ejemplo, errores y complicaciones. Estas anotaciones y la observación de los videos de interacción apoyaron a entender mejor las razones de porqué le tomó cierto tiempo a cada usuario realizar cada tarea. Por lo que a continuación se describe lo que se mostró en la tabla 3, agregando las observaciones obtenidas.

1. Registrarse en el juego. Todos los usuarios pudieron completar la tarea en un menor tiempo al esperado.
2. Loguearse en el sistema. Todos los usuarios pudieron completar la tarea en un menor tiempo al esperado.
3. Explorar opciones del menú principal. Todos los usuarios pudieron completar la tarea en un menor tiempo al esperado.
4. Iniciar una campaña y jugar. Debido a que se trata de un juego aleatorio, es difícil calcular el tiempo de prueba que se llevaría este punto por lo que los tiempos por cada usuario fueron muy diferentes entre sí y respecto al esperado.
5. Modificar las opciones del sistema. Todos los usuarios pudieron completar la tarea en un menor tiempo al esperado.

* + *Satisfacción*

La satisfacción se refiere a la percepción del usuario, sentimientos y opiniones del producto. Para la medición de la satisfacción se aplicó el cuestionario SUS y se obtuvieron los siguientes resultados:



*Ilustración*

*7*

*Respuestas*

*a*

*preg*

*untas*

*positivas en escala S*

*US*

0

1

2

3

4

5

6

pregunta 1

pregunta 3

pregunta 5

pregunta 7

pregunta 9

Preguntas impares

0

0.5

1

1.5

2

2.5

3

3.5

pregunta 2

pregunta 4

pregunta 6

pregunta 8

pregunta 10

Preguntas pares

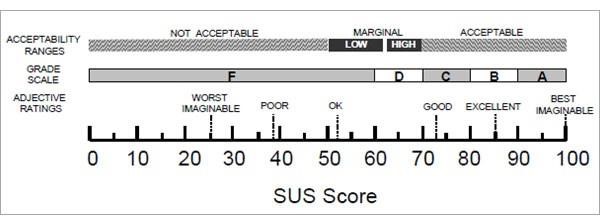
*Ilustración 8 Respuestas a preguntas negativas en escala SUS*

Para el primer participante el resultado obtenido fue de 80% en la escala de SUS.

Para el segundo participante, el resultado fue de 90% según la escala de SUS.

Para el tercero, el resultado obtenido fue de 77.5% en la escala de SUS.

Por lo tanto, el resultado general de las pruebas aplicadas fue de 82.5% que, según la escala de SUS, indica que el sistema se encuentra en el nivel B, Sistema Excelente.



*Ilustración 9 Escala SUS*

* + Aspectos negativos:
  + Contiene leves inconsistencias entre ventanas
  + Ligera dificultad para recorrer - Aspectos positivos:
  + Fácil de utilizar
  + Buen nivel de confianza brindada al usuario

### Conclusiones

### Sobre resultados

Consideramos que los resultados fueron positivos ya que, excluyendo la parte del juego en modo campaña, por ser aleatorio, ninguna tarea tomó a los usuarios más tiempo del que esperábamos. Además, durante las pruebas, ningún usuario mostró aversión por el diseño y organización del contenido. Por último, los resultados de satisfacción obtenidos por medio de la escala de SUS nos arrojaron resultados positivos respecto a los participantes después de utilizar las funcionalidades del juego.

### Modificaciones a futuro

Indudablemente, el juego no está libre de errores y consideramos que los comentarios más recurrentes con respecto al sistema, tanto dentro del procesos de las pruebas como durante todo el desarrollo, fueron:

* Errores de ortografía
* Errores de traducción
* Inconsistencias
* Diseño de ventanas emergentes

Por lo que las mejoras que se aplicarán a futuro serán directamente basadas en estos pontos. Por ejemplo, la revisión de la ortografía y traducciones en el juego, cambios en la organización de los botones en algunas ventanas, posibilidad de cambio en los colores de la interfaz y rediseño de las ventanas emergentes.

### Experiencia

Consideramos que la experiencia durante la aplicación de pruebas de usabilidad fue muy importante para nuestro desarrollo profesional, ya que, dichas pruebas, constituyen una parte fundamental para el proceso de desarrollo de cualquier sistema. Además, aprendimos de los errores que descubrimos al aplicarlas, principalmente de la organización. Por ello esperamos mejorar nuestros procesos para proyectos futuros.

En general, creemos que este proyecto nos ayudó a descubrir formas de utilizar el desarrollo de sistemas centrado en usuarios.

# Análisis estático de código

# Conclusiones

## Lecciones aprendidas por estudiante

### Luis Angel Olivo MartÍnez

Creo que este proyecto fue muy importante para mi formación, ya que aprendí muchísimo de comunicación en red desde aplicaciones y de desarrollo en general, aparte de esto, mejoré mucho mis habilidades para ser autodidacta a la hora de aprender cosas que no sabía cómo hacer y aprendí la importancia que puede tener dedicar una parte del tiempo de desarrollo a seguir buenas practicas y mantener el código lo más entendible posible.

Otra lección aprendida para mi fue la importancia de el uso de una aplicación colaborativa como GitHub, ya que no lo usamos como debíamos y por lo tanto, se nos complicaron algunas cuestiones de comunicación y colaboración en el equipo

### Roberto Quiñones Cordova

Fueron muchas las habilidades que hemos adquirido durante este proyecto, por lo que destacó la importancia que éste ha tenido en nuestra formación. Toda la investigación que conllevo nos ayudó a reflejar habilidades como la investigación y el autoaprendizaje. El conocimiento adquirido sobre la parte técnica sobre conexiones, latencia, seguridad han enriquecido bastante nuestra experiencia de aprendizaje. Nos ayudó a conocer, aprender y comprender nuevas tecnologías desconocidas para mí hasta entonces. No hizo aprender el su uso y combinarlas, a la vez con prácticas, patrones y con entre tecnologías para mejorar el proceso de desarrollo de software.

Además, este proyecto fortaleció nuestro desempeño al trabajar en equipo, desde la organización y distribución de tiempo. Mejoró nuestra habilidad para atravesar situaciones adversas al desarrollo del proyecto. Todo esto nos ayudó positivamente en muchos otros aspectos más.

# Referencias

https://visualstudio.microsoft.com/es/vs/

https://docs.microsoft.com/es-es/dotnet/framework/data/adonet/ef/overview

https://docs.microsoft.com/es-es/dotnet/framework/data/adonet/ef/security-considerations

https://docs.microsoft.com/es-es/dotnet/framework/wcf/whats-wcf

https://docs.microsoft.com/es-es/visualstudio/designers/getting-started-with-wpf?view=vs-2019

http://www.essi.upc.edu/~gomariz/index\_archivos/IntroduccionSD-EnricMartinez.pdf

<https://cursos.mejorcodigo.net/article/que-es-y-como-prevenir-inyeccion-de-sql-php-69>

https://docs.microsoft.com/es-es/visualstudio/test/unit-testing-visual-csharp-code-in-a-store-app?view=vs-2019

https://docs.microsoft.com/es-es/dotnet/standard/class-library-overview

https://docs.microsoft.com/es-es/dotnet/standard/class-libraries