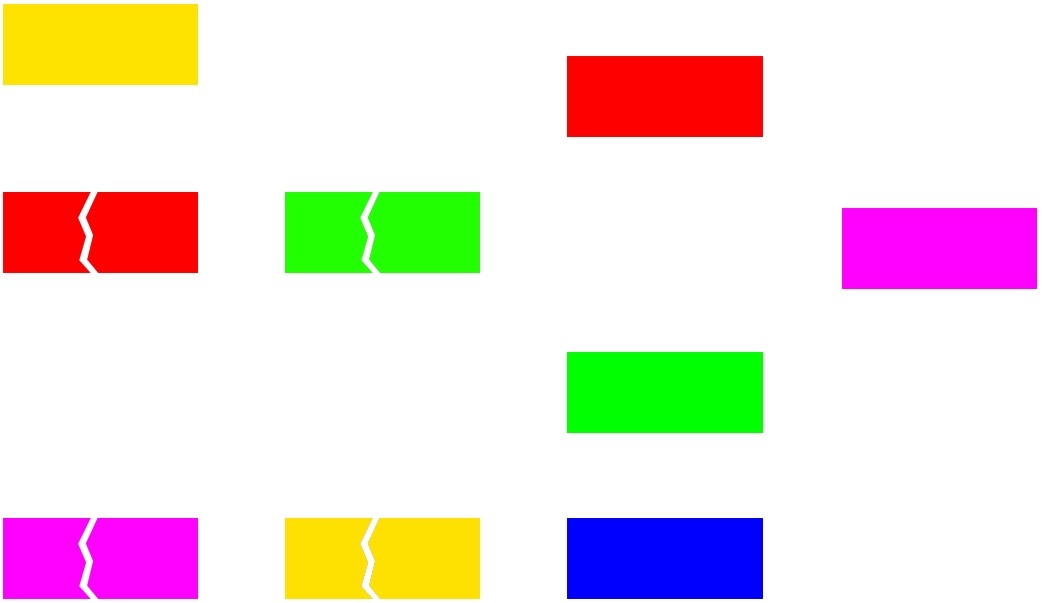
MAGNATES ARKANOID



Realizado por:

José Daniel Mauricio Guerrero 00089719
Rodrigo Efraín Figueroa Aguirre 00157219
Félix Gerardo Guevara Palacios 00055419
Rodrigo Alejandro Fernández Leiva 00041619

Fecha de entrega: 23/06/2020

Índice:

Aspectos generales

Objetivo del documento

- El objetivo de este documento es explicar el funcionamiento del proyecto y las herramientas utilizadas.

Descripción del proyecto

- Para el desarrollo del videojuego se hizo uso del modelo-vista-controlador y una conexión a una base de datos junto con conceptos programación orientada a objetos. El programa está desarrollado para que el usuario disfrute de una buena jugabilidad.

Software utilizado

- Para el desarrollo del videojuego utilizamos JetBrains Rider 2019, junto con PgAdmin 4 para la creación de una base de datos. Se usó como complemento Npgsql para tener la conexión a la base de datos en el programa.

Modelos utilizados

Diagrama UML de casos de uso:

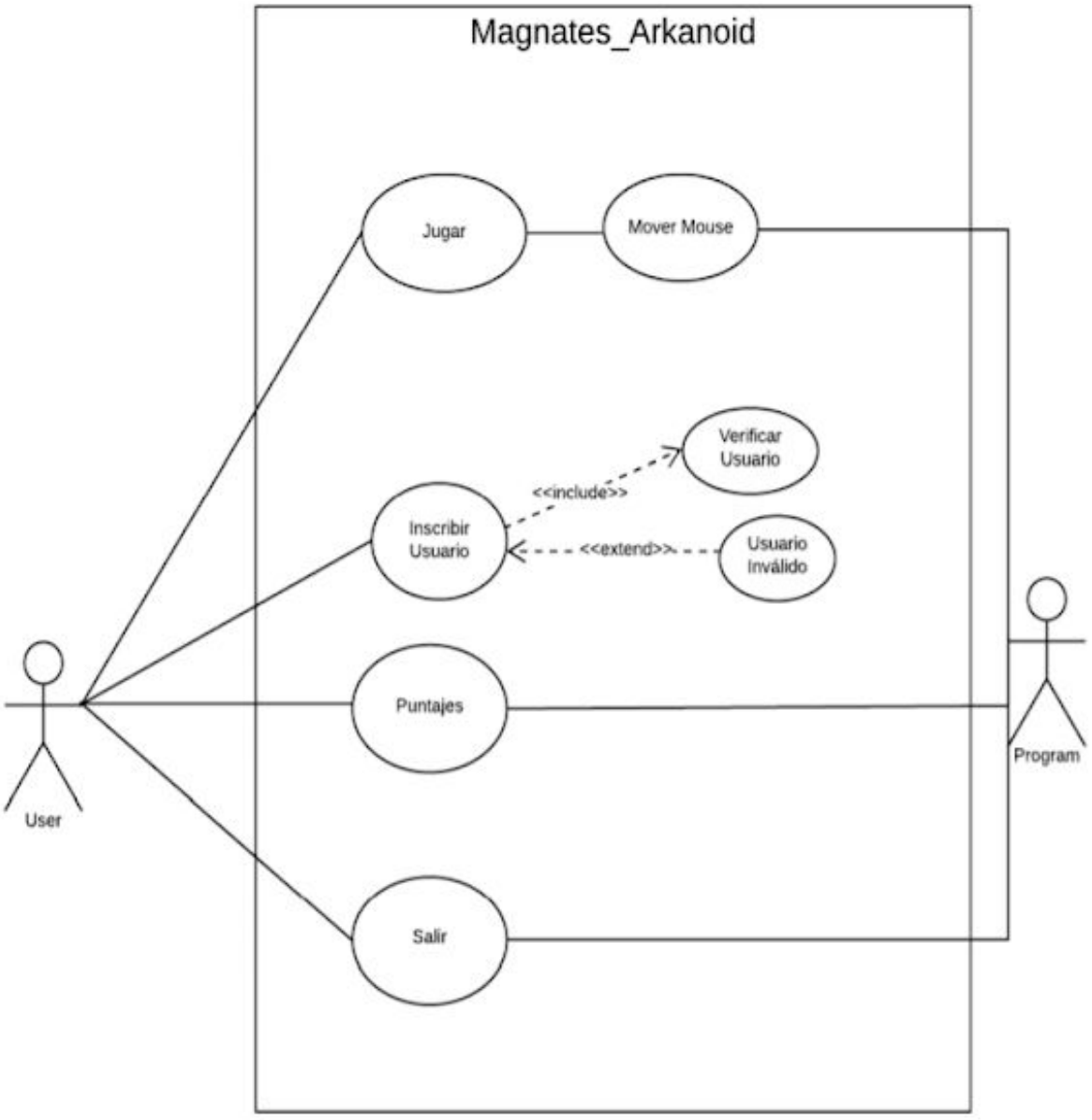
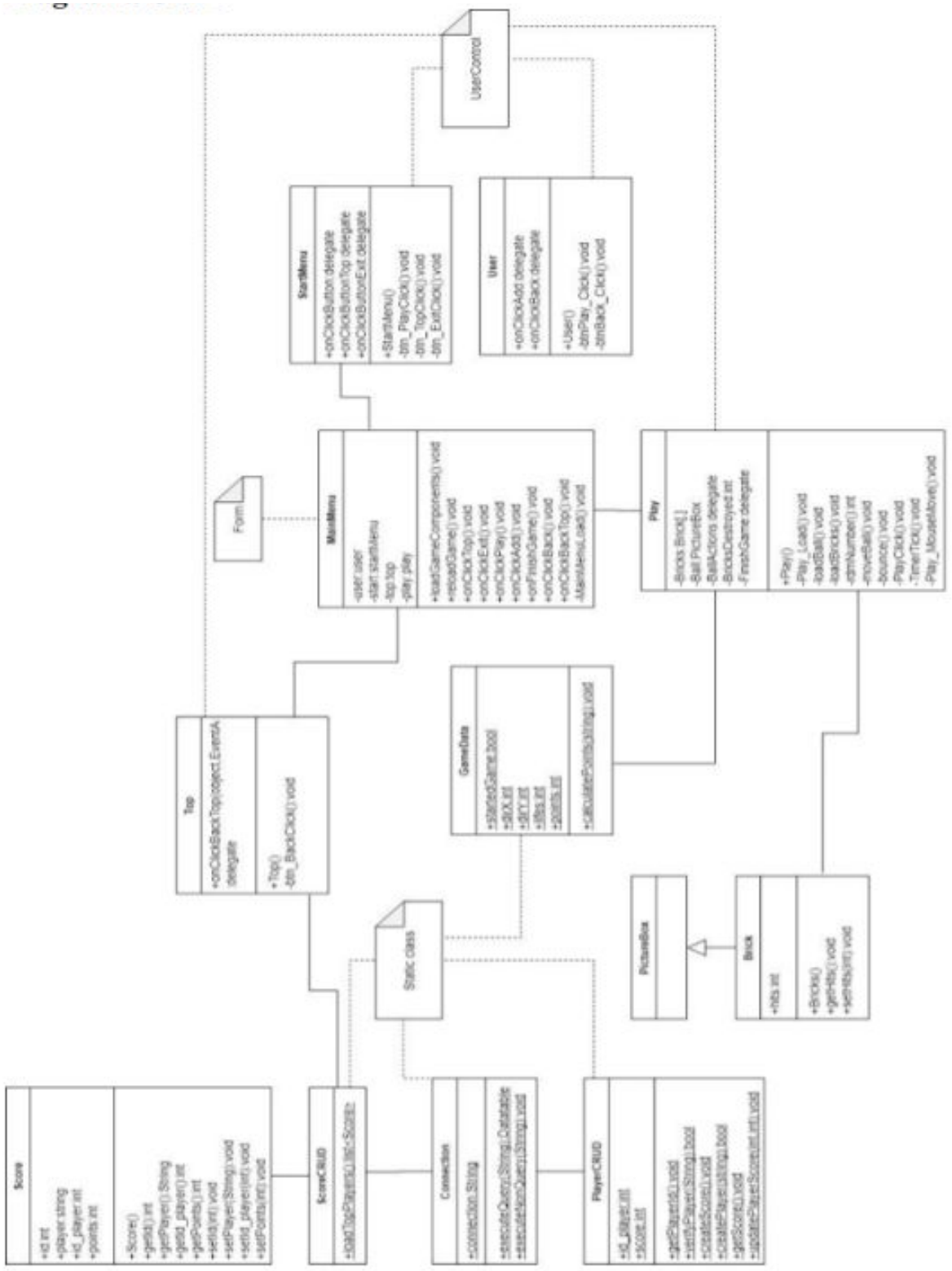


Diagrama UML:



Explicación del Diagrama de clases

En todo el juego se pueden agregar cualquier cantidad de jugadores, o ingresar al juego con un nickname previamente registrado.

Existen 4 clases estáticas en el programa:

1. ConnectionDataBase: Se encarga de realizar la conexión a la base y la ejecución de las consultas.
2. ScoreCRUD: Su único método se encarga de obtener el top 10 de jugadores y sus puntajes.
3. PlayerCRUD: Se encarga de realizar las principales acciones en torno al jugador tales como obtener su id, actualizar su puntaje, crear nuevos jugadores, jugar con uno previamente registrado.
4. GameData: Se encarga de manejar los datos tales como el puntaje, las vidas, las posiciones de la bola, verificar si el jugador ha iniciado la partida y además su único método se encarga de calcular el puntaje que el jugador logra a lo largo del juego.

La clase Score se utiliza para manejar los datos obtenidos en la clase estática ScoreCRUD.

Además se cuenta utiliza la clase picturebox como clase base para ejercer un concepto de herencia con la clase implementada Brick, la cual solamente almacena la cantidad de golpes de cada bloque en pantalla.

El programa cuenta con 4 user controls:

1. StartMenu: se encarga de mostrar la pantalla de inicio y de relacionar los 3 principales botones (salir, top, jugar) con el form MainMenu.
2. User: Se encarga de solicitar el nickname al usuario para iniciar la partida, y verificar que el nickname cumpla con algunos requisitos tales como que no sea nulo y tenga menos de 15 caracteres.
3. Top: Se encarga de mostrar el top 10 de jugadores con base a los datos obtenidos en la clase ScoreCRUD.
4. Play: Se encarga de mostrarle al jugador todos los elementos para que pueda jugar tales como los bloques, la bola, las vidas que posee, su puntaje y la tabla, además se encarga básicamente de implementar la jugabilidad ya que sus métodos se encargan de la física de la pelota, la destrucción de los bloques, los movimientos de la tabla.

El form MainMenu se encarga de manejar los 4 user controls que posee el programa y enlazarlos entre sí a través del uso de delegates y de recargar algunos componentes al finalizar la partida o al seleccionar la opción de regresar en el user Top o en el user User.

Diagrama entidad- relación extendido:

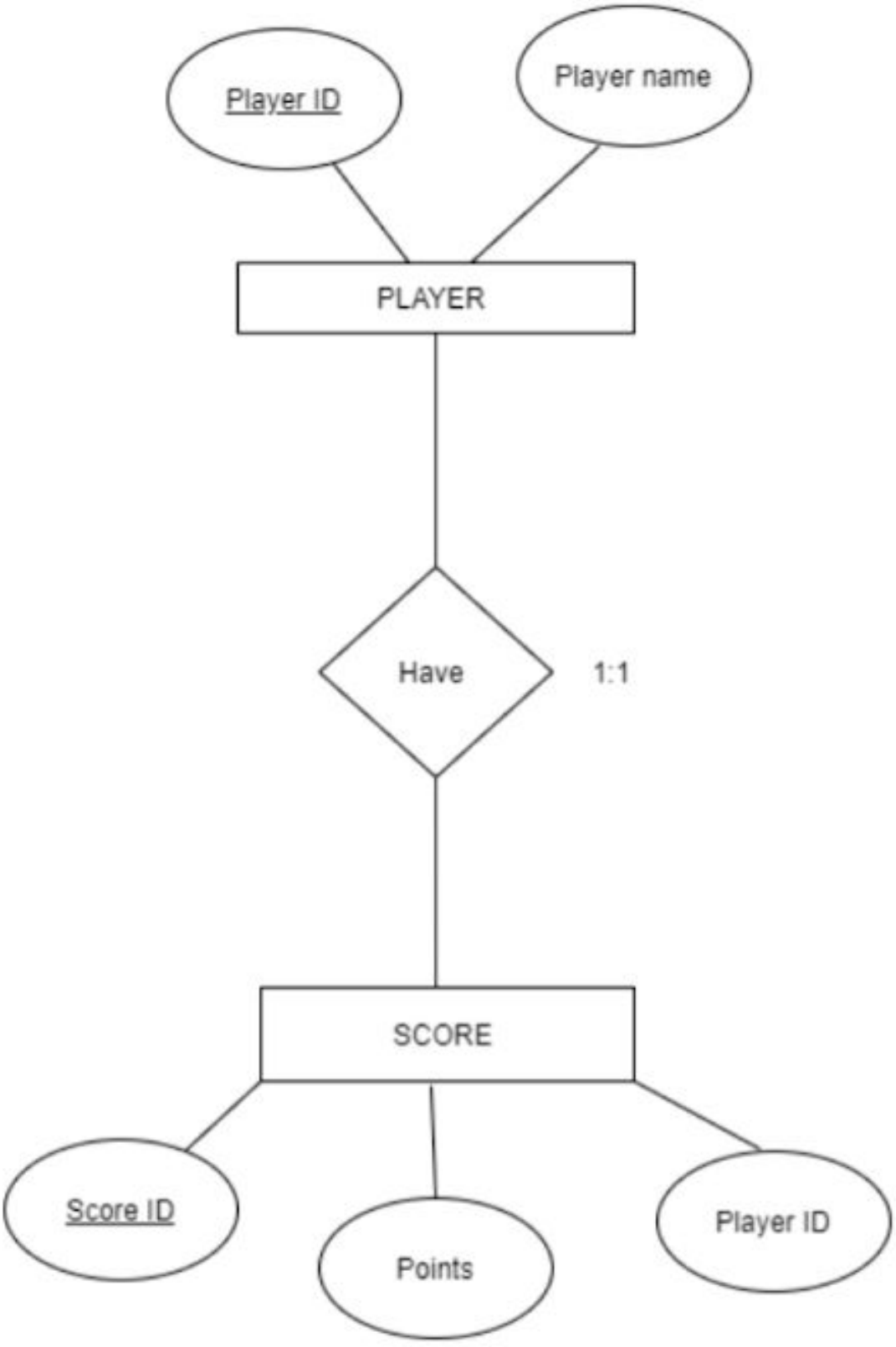


Diagrama relacional normalizado:

Esquema utilizado para la realización de la base de datos:

Conceptos técnicos

Implementación de la interfaz gráfica:

La interfaz gráfica del programa consiste en una ventana global (form), que contiene 3 botones y esos botones se encargan de cargar distintos user control, cada user control tiene su propio código para realizar la acción que se le solicita.

Los user controls utilizados en el programa son:

- ➔ Play.cs
- ➔ StartMenu.cs
- ➔ Top.cs
- ➔ User.cs

Manejo de clases

- ➔ GameData.cs
- ➔ Brick.cs
- ➔ DataBaseConnection.cs
- ➔ PlayerCRUD.cs
- ➔ Score.cs
- ➔ ScoreCRUD.cs
- ➔ Program.cs

Plataforma base

Sistema operativo	Multiplataforma
Tecnologías	JetBrains Rider 2019
Lenguaje	C#
Gestor BD	PostgreSQL

Nomenclaturas:

Para los elementos del entorno gráfico se implementó la siguiente normativa de abreviatura:

<Abreviatura del tipo>_descripción<id correlativo>

Abreviaturas:

Label	lbl
TableLayout	tbly
TextBox	txt
PictureBox	ptb
Button	btn

Eventos y Excepciones

Eventos:

- ❏ GameMouseMove: Se encarga de los movimientos de la tabla.
- ❏ timer1.Tick: Se encarga de ejecutar los métodos que permiten los movimientos de la bola.
- ❏ Play_Click: al ejecutarse en encarga de iniciar el juego.
- ❏ onClickBackTop: Cambia del user control de top 10 al user control del menú principal.
- ❏ onClickBack: Se encarga de cambiar del user control de user al user control del menú principal.
- ❏ onClickAdd: Se encarga de mostrar el user control en el que se desarrolla el juego. onClickPlay: Se encarga de cambiar del user control del menú principal al user control que se encarga de registrar o ingresar con un nickname ya registrado al juego.
- ❏ onClickExit: Se encarga de finalizar el juego.
- ❏ onClickTop: Se encarga de cambiar entre user controls, del menú inicial al top 10.

Excepciones:

Las excepciones usadas en el programa están declaradas como Exception, que es una excepción general para captar diferentes tipos de errores.

Y un par de excepciones personalizadas como su nombre lo indica:

- ❏ InvalidLength.cs
- ❏ InvalidTag.cs
- ❏ NullNickName.cs