




# Manual Técnico

Arkanoid



Benjamín Eduardo Carpio Quintanilla	00205619
Carlos Alfredo Vásquez Carrillo	00062619
Eduardo Stanley Domínguez Payés	00136319
William Josué Pineda Martínez	00225919

## Contenido

Aspectos generales .....	2
Objetivo del documento .....	2
Descripción General .....	2
Software utilizado .....	2
UML .....	2
UML Diagrama de clases .....	2
Diagrama Entidad Relación Extendido .....	4
Diagrama Relacional.....	4
Diagrama de casos de uso.....	5
Conceptos técnicos y distintos tipos de error .....	6
Implementación de interfaz gráfica .....	6
Manejo de clases en modelo.....	6
Plataforma base .....	6
Nomenclaturas.....	7
Eventos y Excepciones. ....	7
Eventos .....	7
Excepciones .....	7

---

## Aspectos generales

---

### Objetivo del documento

Orientar y explicar el diseño del software creado, explicando las herramientas utilizadas y a utilizar para futuras versiones dentro de la materia Programación Orientada a Objetos

### Descripción General

Para la creación del software se hizo uso del modelo – vista–controlador, sus siglas MVC. El programa presenta su principal funcionalidad en el manejo de una plataforma de juego que según el usuario juega así se guarda la puntuación.

### Software utilizado.

Para la creación del programa se utilizó JetBrains Rider 2019.3.4, en conjunto con PostgreSQL 12 para la creación de la base de datos, también se utilizó Trello para la buena distribución de tiempo.

---

## UML

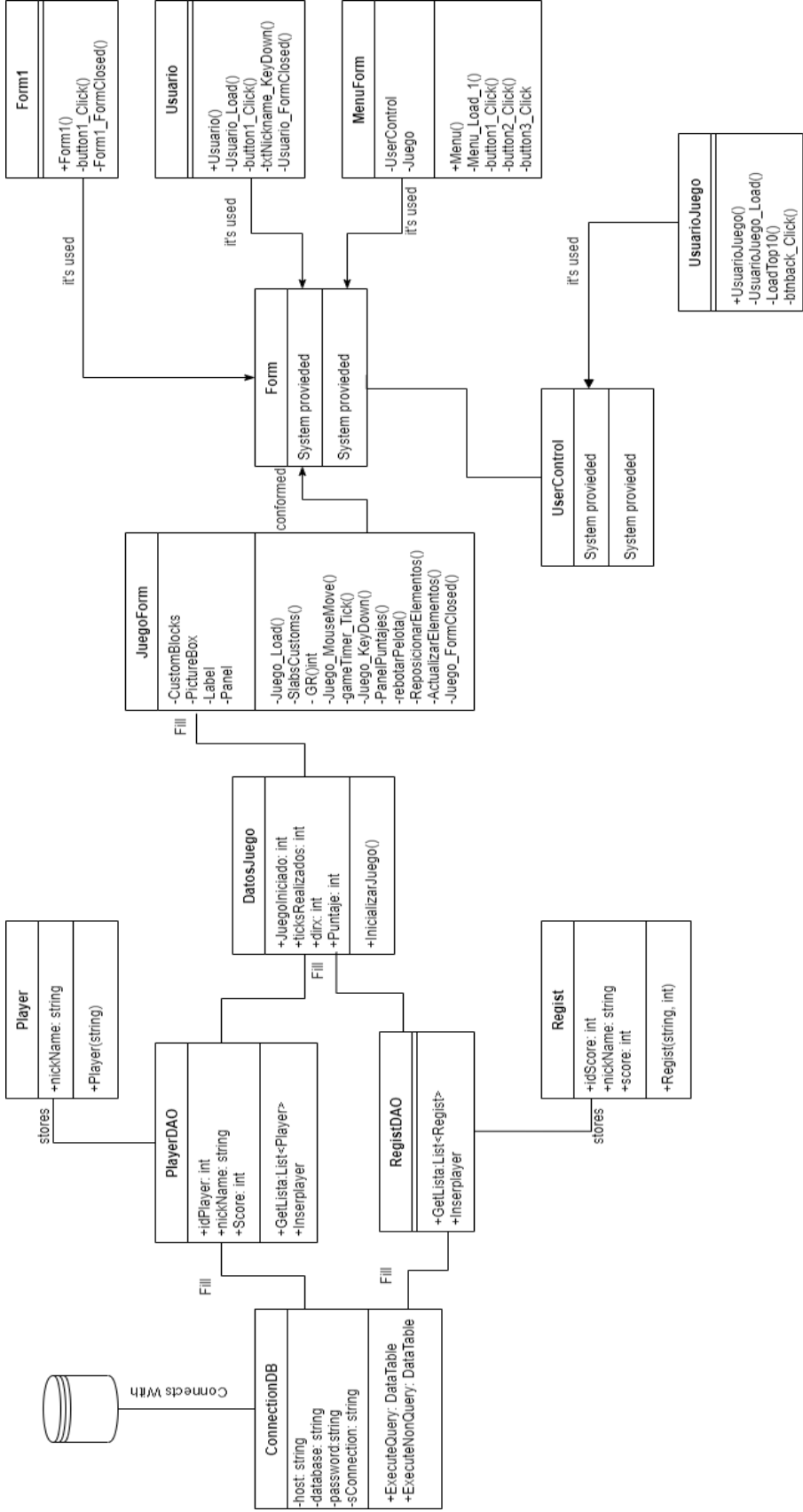
---

El diseño arquitectónico del código está basado en el diagrama de clases.

### UML Diagrama de clases

El siguiente diagrama por presentar consta de la descripción de forma resumida de las clases y componentes que se utilizaron para el buen funcionamiento del código y del juego en si mismo. Puntos por tomar en consideración el diagrama esta planteado de la forma que se cree correctamente tomado de punto de partida la base de dato en la cual se almacena la información que el usuario del juego ingrese. Luego se añaden las clases correspondientes con las que se ara el registro (top de mejores 10 puntuaciones). Seguidamente se muestran los Form y UserControl que se utilizan para la interfaz del proyecto dando la bienvenida al juego, luego se solicita el nombre de usuario que desee (nickname), y se muestra el menú donde se presentan tres opciones de “Jugar” “Ver Top” y “salir”.

Nota: se anexará la imagen del diagrama en la carpeta para una mejor visualización.

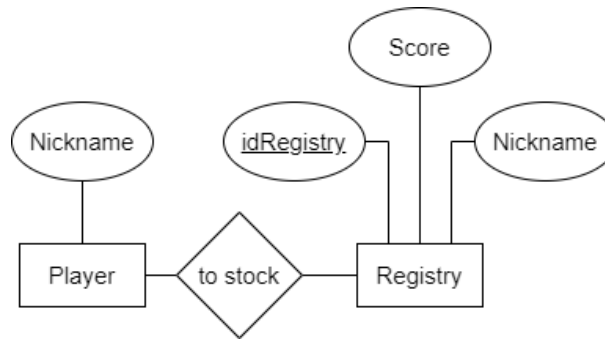


---

### Diagrama Entidad Relación Extendido

---

El diagrama que se utilizó para la creación de la base de datos es el siguiente:



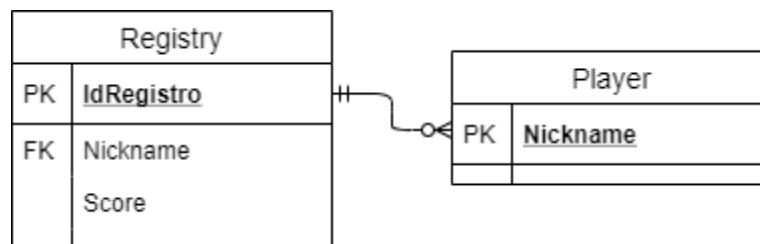
Se plantea de dicha forma debido a que la información que se debe de almacenar del usuario es poca únicamente basta con el nombre que el usuario elija (Nickname) que se almacenara en la entidad Player y por el otro extremo se necesita de la entidad Registry debido a que se almacenara la información de los puntos que el usuario ha realizado.

---

### Diagrama Relacional

---

El diagrama que se utilizó para la creación de la base de datos es el siguiente:

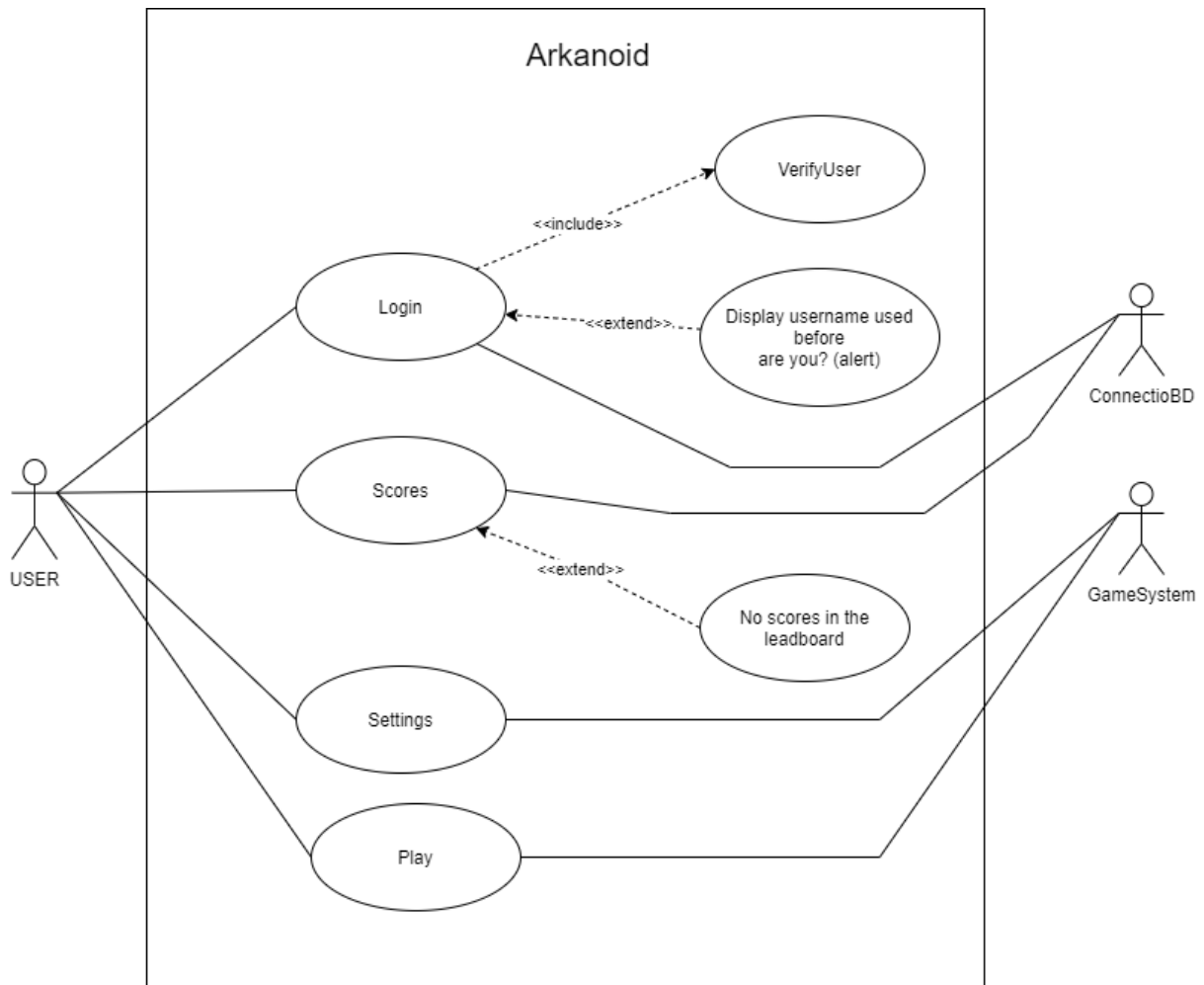


El diagrama únicamente consta de dos tablas debido a que lo que se busca que guarde el usuario (Nickname) y la puntuación que este genera al jugar (score). El grupo de programadores decidió almacenar esta información debido a que en el menú se tiene un top de los mejores jugadores por lo que se necesita de los campos ya antes mencionados.

---

### Diagrama de casos de uso.

---



En este diagrama de caso de uso se describen las acciones de un sistema desde el punto de vista del usuario. Es una herramienta de suma importancia dado que es una técnica de aciertos y errores para obtener los requerimientos del juego, justamente desde el punto de vista del usuario.

En el diagrama esta claramente las funciones que desarrollara estando dentro del código del proyecto, es decir, lo que el Nickname podrá realizar y lo que no. En el diagrama se ve también en el uso de otros actores como ConnectioBD con este se pretende que los programadores almacenen la información que se genera con la participación de usuario, y otro actor con el nombre de GameSystem que se pretende que los programadores tengan la oportunidad de ver el sistema de juego o modificarlo a conveniencia.

---

## *Conceptos técnicos y distintos tipos de error*

---

### Implementación de interfaz gráfica

La interfaz gráfica del programa consiste en una combinación de ventanas global o formulario, compuesta de distintos botones y un panel principal llamado Form1 en el que se le da la bienvenida al usuario, que se encarga de cargar distintos Form's del programa según se avance con en el código, cada control de usuario representa una opción propia del código que debe ser implementada, las formas y controles de usuario que están presentes en el programa son:

- 1) Form1Form.cs
- 2) UsuarioForm.cs
- 3) MenuForm.cs
- 4) JuegoForm.cs
- 5) UsuarioJuegoUserControl.cs

### Manejo de clases en modelo

Para manejar la parte Fundamental del programa, se tienen las siguientes clases creadas.

- 1) ConnectionDB.cs
- 2) PlayerDAO.cs
- 3) RegistDAO.cs
- 4) CustomBlocks.cs
- 5) DatosJuego.cs
- 6) Player.cs
- 7) Program.cs
- 8) Regist.cs

### Plataforma base

Tecnología	JetBrains Rider 2019.3.4
Lenguaje	C#
Gestor de Base de Datos	PostgreSQL

---

## Nomenclaturas

---

Para los elementos del entorno gráfico se implementa la siguiente normativa de nombramiento:

<b>label</b>	label
<b>button</b>	button
<b>PictureBox</b>	Player
<b>Timer</b>	gameTimer
<b>TextBox</b>	TxtNickname

---

## Eventos y Excepciones.

---

### Eventos

Los eventos que se encuentran en el Código generado por los Programadores del Grupo Ella no tema son los siguientes:

1. Cuando se cierra cada parte del programa que hace que el Código se cierre bien es decir sin generar problemas que pueden causar inconformidad al usuario.
2. Enter luego de colocar el Nickname del usuario, en lugar de dar clic en botón de continuar.

### Excepciones

Las siguientes excepciones constan de un constructor que recibe un string con el mensaje de error y se presentan a continuación:

#### EmptyNicknameException

Excepción encargada de verificar que el nombre del usuario (nickname) este escrito de lo contrario lanzara un mensaje predeterminado.

#### NicknameExceedMaxLength

Excepción encargada de verificar que no se exceda de los caracteres permitidos en el nickname.

#### NoMoreLivesException

Esta excepción se genera cuando tiene cero vidas en el juego por lo que se detiene el timer y termina el juego.



### NoBoundException

Excepción encargada de generar un mensaje predeterminado cuando la pelota rebota abajo y activa que se reposicione todo y quite una vida de la tres con las que se cuentan.

### WrongStartKey

Excepción encargada de verificar que si se presiona otra tecla que no sea space a la hora de iniciar el juego se mandara el mensaje predeterminado en el código.