

Manual técnico

Arkanoid

Realizado por:

- **Rodrigo Ernesto Mejia Galea** **00037119**
- **Melissa Yaneeth Guardado Espinoza** **00033719**
- **Gabriela Beatriz Solorzano Nuila** **00185119**
- **Carlos Enrique Guzman Espinoza** **00031119**



CONTENIDO

- Manual técnico Arkanoid
- Aspectos generales
 - Objetivo del Documento
 - Descripción General
 - Software utilizado
- Modelos utilizados
 - UML Diagrama casos de uso
 - UML Diagrama de clases
 - Diagrama relacional normalizado de base de datos utilizada.....
- Conceptos técnicos.....
 - Manejo del funcionamiento del Juego en “Game”
 - Manejo del Funcionamiento del registro e Inicio de Sesión de los usuarios del juego en “DataRegister”
 - Manejo del funcionamiento de Top 10 de los mejores puntajes en “TOP 10”
 - Manejo de Clases en “Controlador”
 - Plataforma Base.....
- Nomenclaturas
 - Abreviaturas
- Eventos y Excepciones
 - Eventos
 - Excepciones

ASPECTOS GENERALES

Objetivo del documento

Detallar el diseño y funcionamiento del software desarrollado, describiendo las herramientas utilizadas, facilitando el escalamiento del mismo.

Descripción general

Partiendo del Modelo-Vista-Controlador, cuyas siglas son MVC, se desarrolló un programa que pretende emular el juego “Arkanoid”, permitiendo a sus usuarios experimentar el juego clásico, con un toque futurista.

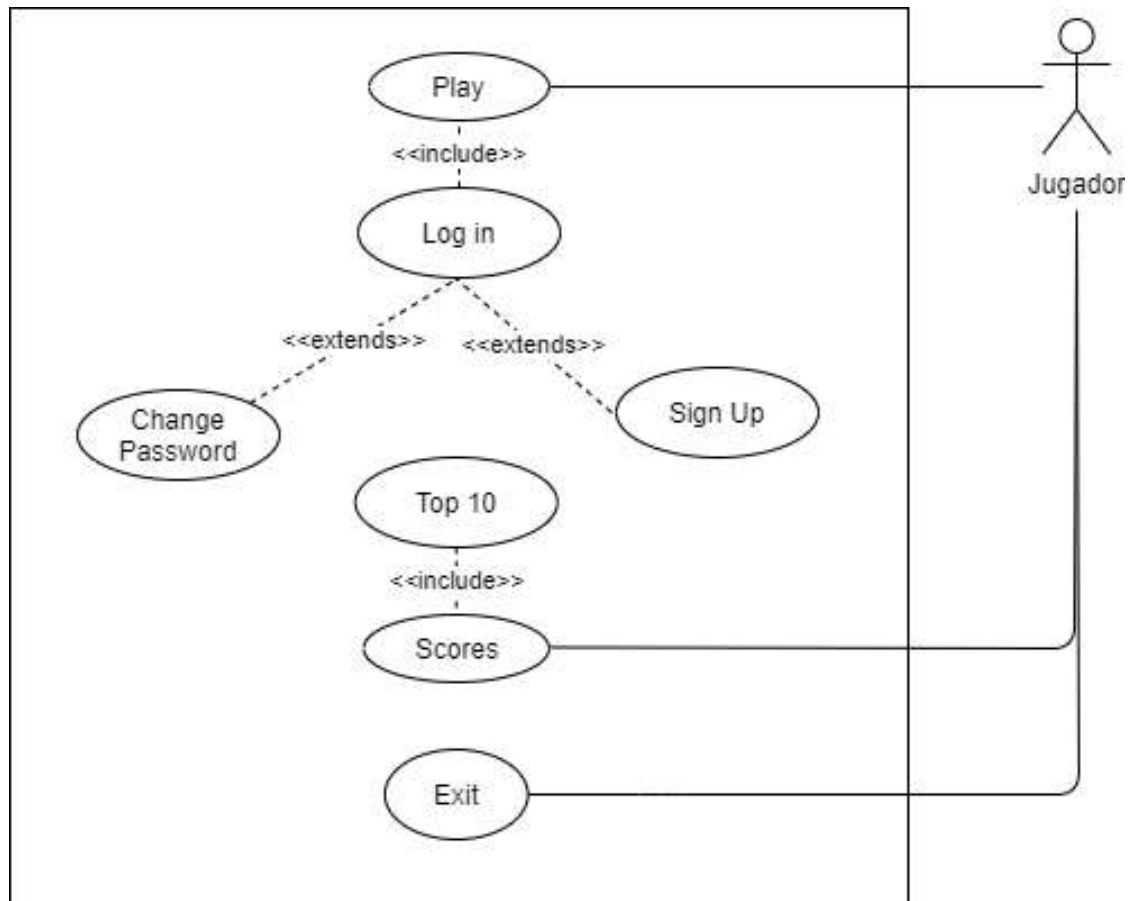
Software utilizado

A lo largo del desarrollo del programa, se utilizó tanto el IDE JetBrains Rider para programar el software, cómo PostgreSQL 12 y pgAdmin 4 para manejar la base de datos. Complementando dichas herramientas, también se utilizó Npgsql para realizar la conexión a la base de datos.

MODELOS UTILIZADOS

UML Diagrama de casos de uso

El siguiente diagrama sirve para especificar la comunicación y el comportamiento del juego mediante su interacción con los usuarios o jugadores. En pocas palabras, un diagrama que muestra la relación entre los jugadores y los casos de uso del juego Arkanoid.

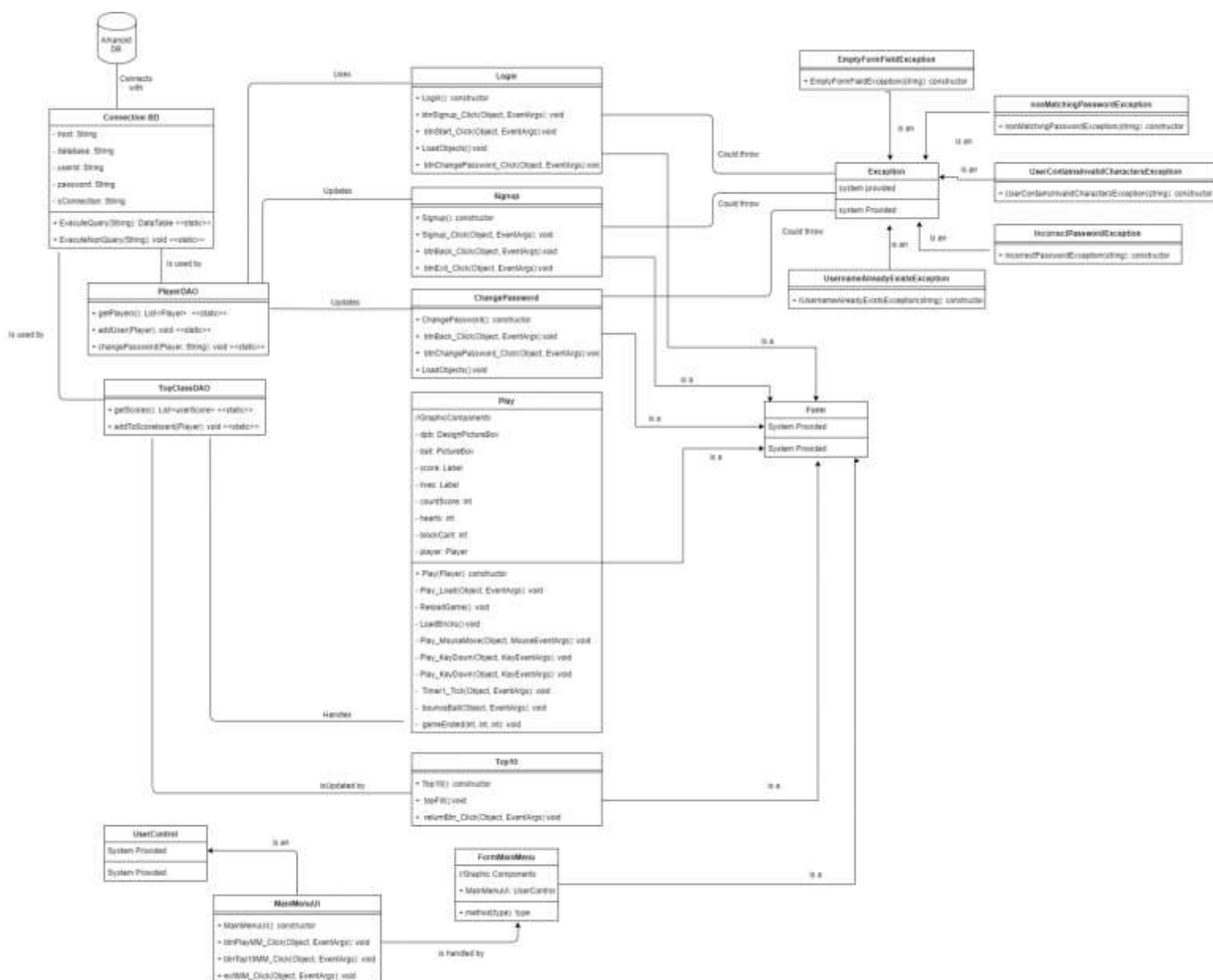



UML Diagrama de clases

El diseño arquitectónico del proyecto está basado en el siguiente diagrama UML de clases.

Se adjunta un hipervínculo para una vista más detallada:

<https://drive.google.com/file/d/1nXzJeTJkTdRnN3qHAsoJZzUW-ogf3o2/view?usp=sharing>






Explicando el diagrama, la ejecución del programa se lleva a cabo por medio de Forms, las cuales funcionan como ventanas en las que el usuario interactúa con el programa. El juego inicia con el menú principal, llamado FormMainMenu.


La ventana FormMainMenu consta de un UserControl llamado MainMenuUI. A partir de esta ventana se hace la navegación a las demás.

La ventana LogIn permite al jugador ingresar las credenciales de su usuario, conectándose a la base de datos por medio de PlayerDAO, siendo esta clase compuesta solamente de métodos estáticos que permiten leer y modificar los datos de la tabla “Players” de la base de datos.

La ventana SignUp permite al jugador crear un usuario, empleando PlayerDAO para ingresar los datos de dicho usuario a la base de datos.

La ventana ChangePassword es utilizada para modificar la contraseña de cualquier jugador existente, empleando PlayerDAO para ejecutar dichas modificaciones en la base de datos.





Las ventanas mencionadas anteriormente, exceptuando el menú principal, son sujetas a devolver excepciones, por lo tanto, se han creado 5 excepciones personalizadas, las cuáles son hijas de Exception, una clase propia de c#. Dichas excepciones buscan eliminar errores al momento de crear, modificar o utilizar los datos del jugador.

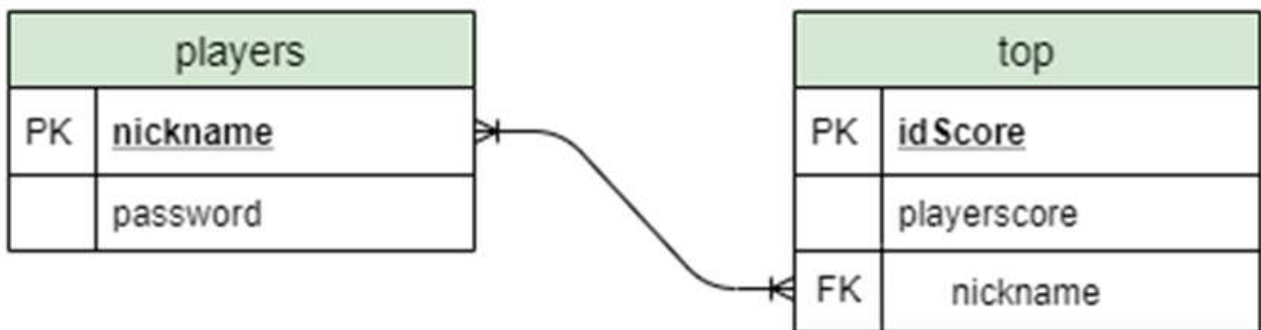
La ventana Play muestra todos los componentes gráficos del juego. Al terminar cada partida, Play envía la información del jugador, y su puntaje, a la clase TopClassDAO, la cuál almacena dicho puntaje en la base de datos.

En la ventana Top10 se muestran los diez mayores puntajes almacenados en la base de datos. Se accede a dicha información mediante TopClassDAO.



Diagrama relacional normalizado de base de datos utilizada

El siguiente esquema permitiría relacionar a los usuarios creados con su respectivo “nickname” y “contraseña” a la tabla “top”, donde se obtendría el “score” correspondiente del jugador que se encuentra en el Top 10.



CONCEPTOS TÉCNICOS

Manejo del funcionamiento del juego en "Game"

Para que la parte principal del proyecto, o sea el juego en si, pueda funcionar, se tienen las siguientes clases e interfaces graficas:

1. Play.cs
2. DesignPictureBox.cs
3. DataGame.cs

Manejo del funcionamiento del registro e Inicio de Sesión de los usuarios del juego en "DataRegister"

Para que cada uno de los usuarios puedan iniciar sesión o registrarse y así poder jugar, se tienen las siguientes clases e interfaces:

1. MainmenuUI.cs
2. Form1.cs
3. Signup.cs
4. Login.cs
5. ChangePassword.cs

Manejo del funcionamiento de Top 10 de los mejores puntajes en "TOP 10"

Para que los usuarios registrados puedan ver el ranking de los mejores 10 puntajes del juego, se tienen las siguientes clases e interfaces:

1. Top10.cs
2. TopClass.cs
3. TopClassDAO.cs

Manejo de clases en "Controlador"

Para que pueda existir una conexión con la Base de Datos y poder enlazar el código con ella, se tienen las siguientes clases:

1. Connection BD.cs
2. Player.cs
3. PlayerDAO.cs

Plataforma Base

Sistema Operativo	Multiplataforma
Tecnologías	Rider 2020.1.3
Lenguaje	C#
Gestor de DB	PostgreSQL

NOMENCLATURAS

Abreviaturas

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

<Abreviatura de tipo>_descripción_<id correlativo>

Las abreviaturas son las siguientes:

<i>Label</i>	lbl
<i>Button</i>	bttm
<i>ComboBox</i>	cmb
<i>PictureBox</i>	pic
<i>TextBox</i>	txt
<i>TableLayoutPanel</i>	tlp
<i>GroupBox</i>	grp
<i>DataGridView</i>	dgv
<i>DateTimePicker</i>	dtp

EVENTOS Y EXCEPCIONES

Eventos

A la hora de desarrollar *Arkanoid*, se implementaron distintos eventos, objetos que envían un mensaje indicando la ocurrencia de una acción gracias a la interacción con el usuario.

- **PlayEventArgs:**

Registra un clic y que permite el acceso a `Login.cs` para que los usuarios con cuenta puedan acceder con sus datos al juego.

- **StartEventArgs:**

Permite al usuario acceder al juego luego de insertar sus credenciales.

- **SignupEventArgs:**

Permite a los usuarios que aún no se encuentran registrados acceder a `Signup.cs` para registrarse en la BD.

- **ChangePasswordEventArgs:**

Permite al usuario acceder a `ChangePassword.cs` si es que este desea cambiar su contraseña actual.

- **ScoresEventArgs:**

Permite al usuario acceder a ver los scores del top 10.

- **ExitEventArgs:**

Utilizado en caso de que el usuario desee abandonar el juego.

- **BackEventArgs:**

Registra un clic y permite al usuario retroceder a la pestaña anterior.

- **MouseMoveEventArgs:**

Registra el Movimiento de la plataforma del juego mediante el mouse.

Excepciones

Bajo la necesidad de reportar errores que ocurran en tiempo de ejecución, se implementaron las siguientes excepciones, que gracias a sus nombres se puede determinar fácilmente su función.

- EmptyFormFieldException.cs
- IncorrectPasswordException.cs
- NonMatchingPasswordValidationException.cs
- UserNameAlreadyExistsException.cs
- UsernameContainsInvalidCharactersException.cs