



*Provincia de Tierra del Fuego,
Antártida e Islas del Atlántico Sur
República Argentina*
Ministerio de Educación, Cultura,
Ciencia y Tecnología

*** CASINO ROYAL ***

TP Curso Full Stack

Descripción breve

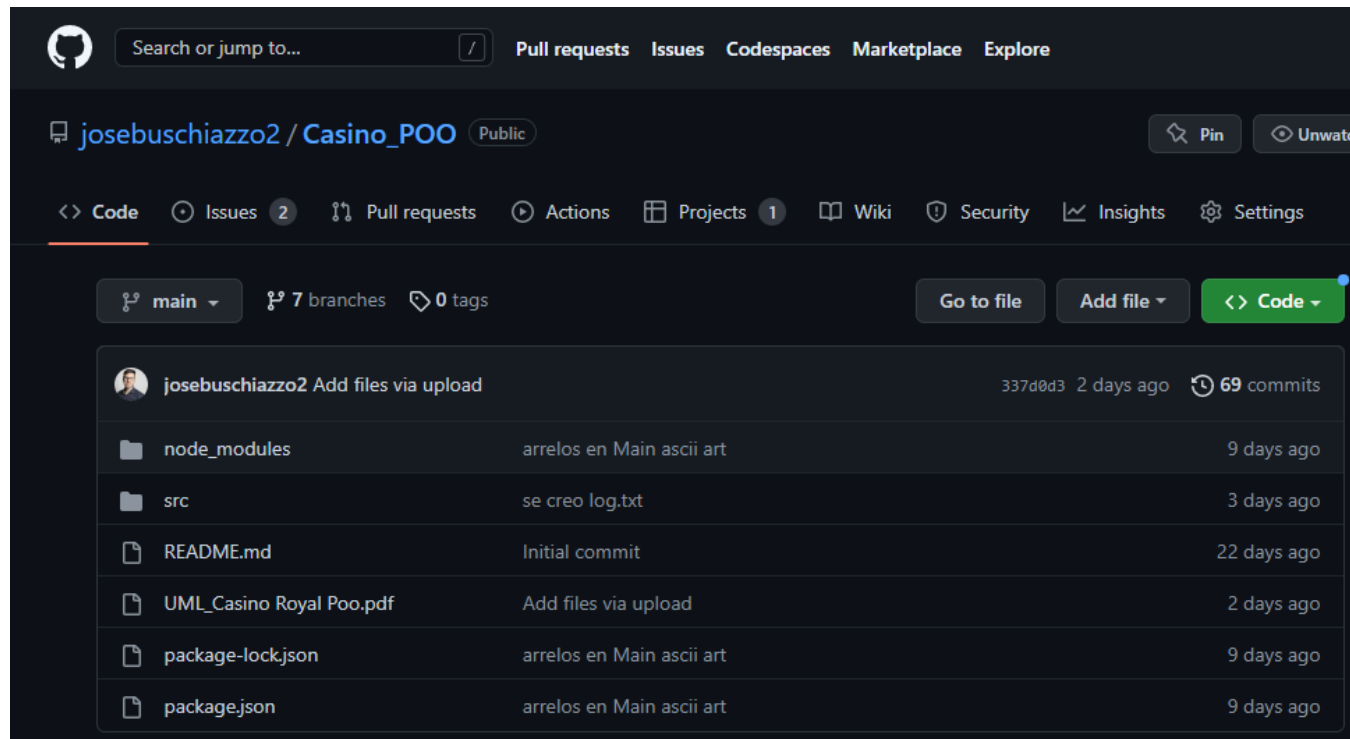
Se realizó un juego de casino, plasmando el marco teórico y práctico de la enseñanza brindada a lo largo del curso de Full Stack

Darío Lopez – Fernando Diaz - José Buschiazzo

https://github.com/josebuschiazzo2/Casino_POO

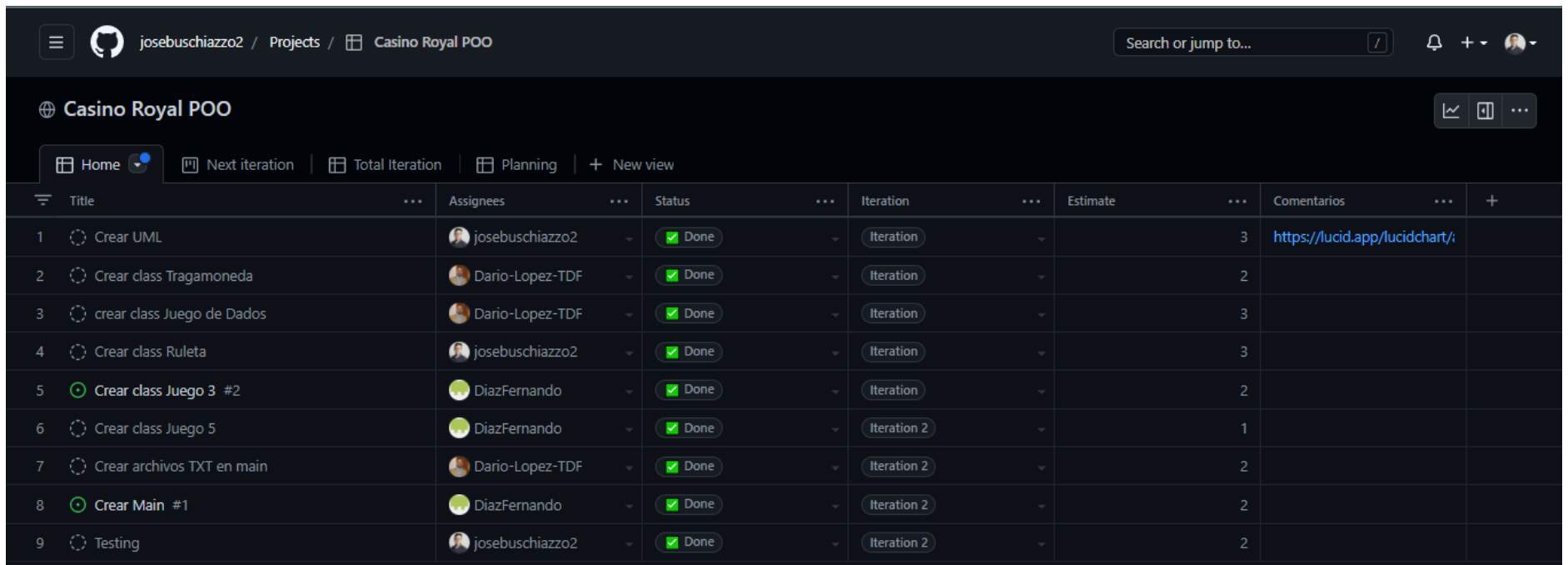
Desarrollo

En primera instancia, se creó el repositorio para trabajar en equipo, realizando todas las contribuciones en GitHub y creando el proyecto desde el nivel inicial. Utilizamos diferentes formas de trabajar sobre este mismo repositorio, realizando los Pulls y Pusheando las veces que fuese necesario por terminal y de la aplicación Visual Studio Code.



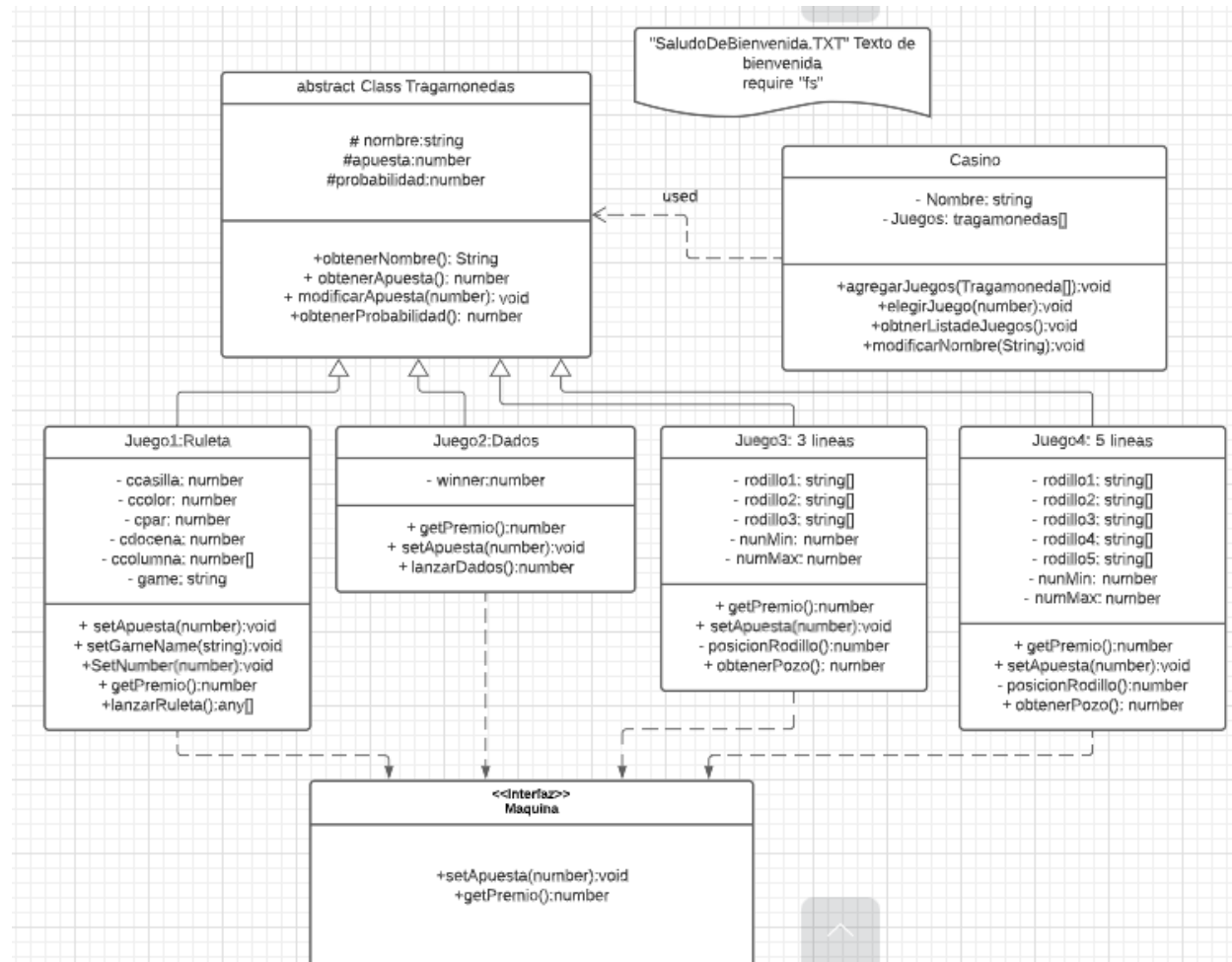
En todo momento buscamos seguir una estructura y planificación de segmentación de actividad, pero compartiendo el conocimiento de cada uno de los integrantes. Para esta segmentación de actividades, se optó por desarrollar el Planning en la misma página del repositorio. Se readejó los ítems del Projects con los ítems que se considero necesarios para su correcta ejecución y consensuado las asignaciones de actividades y tiempo con el team.

En el transcurso de las actividades se fueron realizando las modificaciones permitientes a las interacciones y planning, pero se buscó respetar los tiempos pactados y realizar las reuniones de sincronización al inicio de la jornada, simulando en algún aspecto, a una reunión de la metodología de SCRUM.



<div> <div> <div></div> <div>josebuschiazzo2 / Projects / Casino Royal POO</div> </div> <div> <div>Search or jump to...</div> <div></div> <div></div> <div></div> </div> </div>											
<div> <div>Casino Royal POO</div> <div> <div></div> <div></div> <div></div> </div> </div>											
<div> <div>Home</div> <div>Next iteration</div> <div>Total Iteration</div> <div>Planning</div> <div>+ New view</div> </div>											
Title	Assignees	Status	Iteration	Estimate	Comentarios						
1 Crear UML	josebuschiazzo2	Done	Iteration	3	https://lucid.app/lucidchart/						
2 Crear class Tragamonedas	Dario-Lopez-TDF	Done	Iteration	2							
3 crear class Juego de Dados	Dario-Lopez-TDF	Done	Iteration	3							
4 Crear class Ruleta	josebuschiazzo2	Done	Iteration	3							
5 Crear class Juego 3 #2	DiazFernando	Done	Iteration	2							
6 Crear class Juego 5	DiazFernando	Done	Iteration 2	1							
7 Crear archivos TXT en main	Dario-Lopez-TDF	Done	Iteration 2	2							
8 Crear Main #1	DiazFernando	Done	Iteration 2	2							
9 Testing	josebuschiazzo2	Done	Iteration 2	2							

La primera actividad fue la realización de la UML, donde se utilizó las herramientas de [Lucid.app](https://lucid.app). Consta de una Clase Padre, un abstracta, interfaz y las clases con herencia de los 4 juegos propuesto.



El trabajo practico desarrollado, consiste en un programa que emula un casino Online, donde recibe los distintas apuestas, números, etc.; de cuatro juegos de tragamonedas (dados, ruleta, 3 en línea y 5 en línea).

Las librerías utilizadas

ReadlineSync: es una librería que se utiliza para que el usuario pueda ingresar datos y valores por teclado. En este trabajo practico, esta librería se utilizó para ingresar los datos como Nombre, Juego, apuesta, números, preguntas cerradas como Yes o No.

FS: es una librería que se utiliza para poder leer y escribir valores desde un archivo txt y poderlo utilizar en nuestro programa. La principal finalidad fue de tomar las reglas del juego precargado en archivos txt y la utilización de un registro Log, donde se lleva el historial de las jugadas.

Ascii-art: es una librería de medio artístico que utiliza recursos computarizados fundamentados en los caracteres de impresión del Código. Se utilizó la presentación inicial de "Casino Royal" y los modificar el color del texto

Figlet: Esta librería cuenta con 2 métodos para mostrar un banner en pantalla, la primera forma es asíncrona, pasándole un callback como segundo parámetro, en nuestro caso no necesitamos asincronía para mostrar "Casino Royal".

Descripción del proyecto

El programa comienza mostrando el Banner inicial y entra en un bucle del menú principal, preguntándole al usuario a que juego desea jugar. Donde tiene las 4 opciones (array de juegos): Dados, Ruleta, 3 en línea y 5 en línea.

```
*** CASINO ROYAL ***
```

```
Bienvenido A Casino Royal, "disfrute de nuestros juegos"
```

```
**advertencia**
```

```
"el juego de azar en exceso es nocivo para la salud"
```

```
Por favor selecciones uno de los juegos de la lista
```

```
[1] juego de dados
```

```
[2] Ruleta
```

```
[3] Tres en linea
```

```
[4] Cinco en linea
```

```
[0] CANCEL
```

```
a continuacion coloque el numero del juego deseado [1...4 / 0]: 
```

1. Presionando el número 1 ira al juego Dados, donde dará las reglas iniciales precargadas en un TXT y entrará al bucle del juego dados. En dicho bucle se comienza consultando el nombre del jugar, registrándolo en el archivo "Log.txt" y se procederá a consultar el monto de la apuesta, también guardado en el archivo "Log.txt". En dicho juego mostrara el resultado de la suma de los dos dados y el resultado del valor ganado o si perdió la apuesta, brindado la posibilidad de lanzar los dados y realizar diferentes apuestas según el usuario desee, caso contrario saldrá al menú principal
2. Presionando el número 2 ira al juego Ruleta, donde dará las reglas iniciales precargadas en un TXT y entrará al bucle del juego ruleta. En dicho bucle se comienza consultando el nombre del jugar, registrándolo en el archivo "Log.txt" y se procederá a consultar el tipo de juego (6 tipos: Color, ParImpar, Columna, Docena, Cuadro o Pleno), luego el monto de la apuesta, donde también guardado en el archivo "Log.txt". En dicho juego mostrara el resultado de la ruleta, mostrando el resultado del número que salió, color, si es par o impar, docena, columna y los 4 cuadrantes que toca y, a continuación, mostrara el resultado del valor ganado o si perdió la apuesta, brindado la posibilidad de lanzar nuevamente la ruleta y realizar las 6 diferentes tipos apuestas y montos según el usuario desee, caso contrario saldrá al menú principal.
3. Presionando el número 3 ira al juego de 3 en línea, donde dará las reglas iniciales precargadas en un TXT y entrará al bucle del juego de 3 en línea. En dicho bucle se comienza consultando el nombre del jugar, registrándolo en el archivo "Log.txt" y se procederá a consultar el monto de la apuesta, también guardado en el archivo "Log.txt". En dicho juego mostrara el resultado de los tres rodillos y mostrara el monto del valor ganado o si perdió la apuesta, como así también el pozo acumulado, brindado la posibilidad de lanzar nuevamente los rodillos y realizar diferentes apuestas según el usuario desee, caso contrario saldrá al menú principal.
4. Presionando el número 4 ira al juego de 5 en línea, donde dará las reglas iniciales precargadas en un TXT y entrará al bucle del juego de 5 en línea. En dicho bucle se comienza consultando el nombre del jugar,

registrándolo en el archivo "Log.txt" y se procederá a consultar el monto de la apuesta, también guardado en el archivo "Log.txt". En dicho juego mostrara el resultado de los cinco rodillos y mostrara el monto del valor ganado o si perdió la apuesta, como así también el pozo acumulado, brindando la posibilidad de lanzar nuevamente los rodillos y realizar diferentes apuestas según el usuario desee, caso contrario saldrá al menú principal.

Funciones utilizadas

Funciones que utilizamos la clase Casino:

- *agregarJuegos(Tragamonedas[]):void* : La finalidad es poder agregar distintos tipos de juegos
- *elegirJuego(number):void*: Brinda la posibilidad de elegir el juego
- *obtenerListadoJuegos():void*: Obtiene el listado de los juegos cargados
- *modificarNombre(String):void*: Puede modificar los nombres de los juegos cargados

Funciones que utilizamos como interfaz:

- *getPremio():number* : La utilizamos para consultar el premio ganado en cada jugada
- *setApuesta(number):void* : Ingresa el monto de la apuesta de cada jugada

Funciones de la clase Tragamonedas:

- *obtenerNombre(): String* : Obtener el nombre de cada juego
- *obtenerApuesta(): number* : Devuelve la apuesta inicial.
- *modificarApuesta(number)*: Brinda la posibilidad de modificar la apuesta

Para finalizar, cada clase de cada juego tiene la heredera de la interfaz y las particulares como por ejemplo: para lanzar los dados, la ruleta o los rodillos.

Agradecimientos y Conclusión

Fue muy gratificante trabajar en equipo y poder aplicar lo aprendido en las clases, nos hizo asentar los conocimientos adquiridos y poder entender la dinámica del trabajo virtual como programador.

Queremos agradecer a los profesores y tutores por el soporte brindado en cada clase y evacuar todas nuestras dudas. Sentimos en todo momento el acompañamiento de cada uno de ellos.