SIMULADOR BLACKJACK

Laboratorio 3

**Morreale Lucas, Bereilh Maria Paula y Nivio Luca**

**INTRODUCCIÓN**

En este informe, presentaremos una aplicación de simulación de uno de los conocidos juegos de cartas propio de los casinos, llamado BlackJack.

El programa esta diseñado para brindar una experiencia de juego virtual, sin la necesidad de apostar dinero real. Esta simulación nos permite vivir la experiencia del juego, con dinero ficticio otorgado por la aplicación al comienzo del juego.

Los jugadores pueden realizar apuestas y enfrentarse al crupier utilizando este dinero virtual. El jugador deberá registrarse dentro de la aplicación antes de comenzar a jugar y cada partida es guardada y se puede jugar hasta que el jugador ya no tenga mas dinero.

A lo largo de este informe, explicaremos detalladamente el diseño y la implementación del programa. Analizaremos las caracteristicas principales, como llegamos a realizar cada una de las cosas, las problemáticas y las soluciones planteadas a lo largo del proyecto y las decisiones tanto del código como el diseño para ofrecer este programa automatizado de Blackjack. Ademas, presentaremos ejemplos de código relevante y explicaremos los conceptos fundamentales detrás de la creación del programa.

Descripción del sistema:

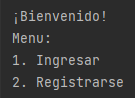
1. Mazo: Se decidió usar una clase genérica de tipo Pila. Tiene funcionalidades de crear, mezclar y sacar carta.
2. Carta: Tiene un rango y un palo que son dos clases de tipo ENUM, a su vez se le asigna un valor numérico a cada objeto de tipo carta.
3. Enums Palo y Rango: Se utilizan para cargar los atributos de la carta.
4. Partida: Es una clase envoltorio del juego. Tiene metodos iniciarPartida para sumar las manos de los jugadores, definir ganador. Tiene un ID, usuario, dealer, mazo, resultado y cantidad apostada.
5. Jugador: Una super clase abstracta que tiene una lista de cartas “mano”, para controlar la mano de los jugadores. Tiene un atributo turno y metodos para recibir y listar cartas.
6. Usuario: Tiene de atributos, nombre de usuario, nombre completo, email, contraseña, saldo y puntaje. Hereda de Jugador
7. Dealer: Hereda de jugador. Es una clase secundaria en el juego y solamente tiene el metodo para listar sus cartas.
8. Clase Genérica Pila: Implementa interfaz I\_Administrar y tiene el comportamiento de una Pila, su atributo es una lista generica, que apila, desapila, tope, pilaVacia, listar y buscar.
9. Clase Genérica Mapa: Implementa interfaz I\_Administrar. Es un mapa generico.
10. Controladora Archivos: Tiene funciones para leer y grabar archivos de usuarios y de partidas.
11. Controladora JSON: Solo lee JSON.
12. Controladora Usuario (ABM (carga controlada)): Cuenta con un mapa de usuarios y tiene todas las funciones para el control y busqueda de los usuarios.
13. Controladora Juego: Utiliza la pila generica para guardar partidas, crea, finaliza, busca y muestra partidas.
14. Interfaz I\_Administrar: La utilizamos para implementar metodos en las clases genericas. Tiene metodos para buscar, listar y ver si la coleccion esta vacia.
15. Menú: Es la clase en la cual se controla el flujo del programa.

**Matriz de soluciones**

|  |  |
| --- | --- |
| PROBLEMA | SOLUCIÓN |
| -En primera instancia, el programa iba a ser un casino virtual, que contenga 2 o 3 juegos, entre ellos blackjack, ruleta y apuestas deportivas. Cuando comenzamos a hacer el código nos dimos cuenta que era muy difícil adaptar todos los juegos con su respectiva lógica. | -Decidimos realizar un programa de simulación de blackjack virtual. |
| -El dinero iba a ser ingresado por el usuario directamente de su tarjeta. | -Decidimos que sea una simulación con dinero ficticio otorgado por la aplicación, ya que es una versión automatizada del juego y no una casa de apuestas. |
| -A lo primero estábamos muy trabados ya que habían muchos temas que no sabíamos como aplicarlos al proyecto que queríamos hacer, y estuvimos por cambiar de idea a algo mas simple como un sistema de compra y venta, o un sistema de una facultad con alumno, profesores, etc. | -Poco a poco se nos fue ocurriendo como incorporarlos y por eso decidimos mantener esta idea. |
| -No sabíamos como incorporar Genericidad en el proyecto. | -Lo solucionamos creando una clase genérica, de una pila genérica, donde se pueden aplicar cartas del mazo y también partidas, que son dos cosas diferentes pero con las mismas funcionalidades. |
| -Otra problemática fue que había que incorporar tres tipos de colecciones y notábamos que este programa no necesitaba tantas. | -Para poder cumplir con lo pedido, fuimos buscando cosas para guardar, realizando un set de partidas, un mapa de usuarios y una lista de partidas. |
| -No sabíamos donde aplicar Herencia, y si crear una clase crupier e instanciarla o que sea la maquina misma. | -La solución fue que usuario y crupier hereden de jugador. |
| -Utilizando asi la herencia perdiamos una interfaz que habíamos realizado. | -Lo solucionamos utilizando una interfaz para las clases genéricas. |

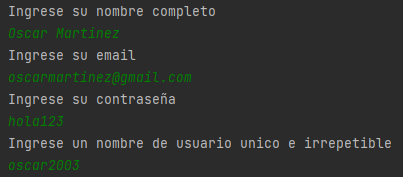
**Manual de usuario**

**1)**



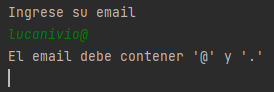
Al ingresar al programa, lo primero que nos muestra es un mensaje de bienvenida, seguido del menú, donde tenemos dos opciones, la primera es ingresar y la segunda registrarse, en caso de ya estar registrados seleccionaremos la opción 1 y de lo contrario la opción 2.

**2)**



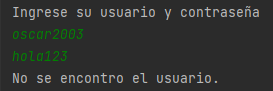
Al seleccionar la opción 2 (registrarse), se le pedirá al usuario que ingrese su nombre completo, email, contraseña y nombre de usuario para luego poder ingresar a la aplicación, sin la necesidad de volver a registrarse.

**3)**

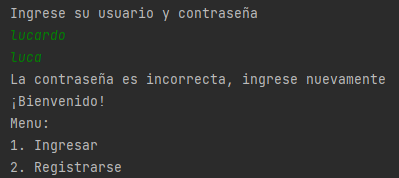


En caso de ingresar mal el email, se le pedirá al usuario volver a ingresarlo.

**4)**

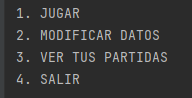


Al seleccionar la opción 1 (ingresar) se le pedirá al usuario ingresar su nombre de usuario y contraseña, para luego poder ya comenzar a jugar, en caso de que el usuario no se encuentre, volverá a enviarnos al menú, para poder registrarnos.



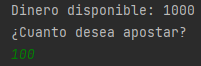
En caso de encontrar el usuario pero poner mal la contraseña, te dirá que la contraseña es incorrecta, y que ingreses nuevamente.

**5)**



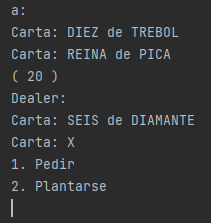
Una vez ya nos registremos o ingresemos a la aplicación, se nos mostraran cuatro opciones, jugar, modificar datos, ver tus partidas y salir.

**6)**



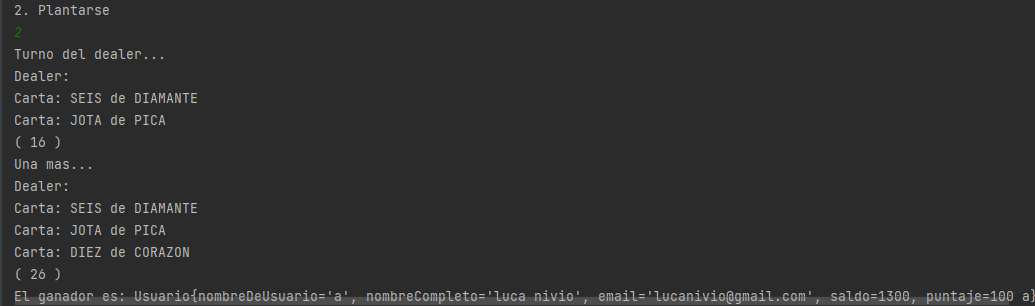
Al seleccionar la opción 1 (jugar), automáticamente se le cargaran $1000 al usuario y se le preguntara cuanto desea apostar en la siguiente partida.

**7)**



Luego de ingresar el dinero que desea apostar, se mostrara en pantalla el nombre del usuario, seguido de las dos cartas que le tocaron. Ademas, se mostrara una de las cartas del dealer y la segunda permanecerá oculta. El usuario deberá decidir seleccionar la opción 1 (pedir) o la opción 2 (plantarse).

**8)**

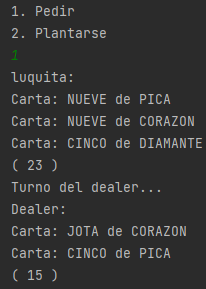


En caso de seleccionar la opción 2 (plantarse), proseguirá el turno del dealer, mostrando las cartas que le van tocando y a lo último se mostrará quien de los dos es el ganador o si hay un empate. Actualizando el saldo y puntaje del usuario, según el resultado de la partida.



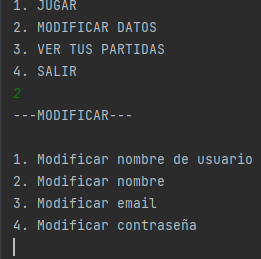
Luego de finalizar la partida se le pregunta al usuario si desea continuar jugando con la misma cuenta, en caso de que si, ingresa ‘s’.

**9)**

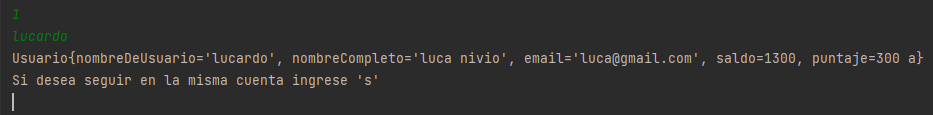


En caso de seleccionar la opción 1 (pedir), proseguirá a agarrar otra carta, en caso de no haberse pasado de 21, podrá volver a elegir entre pedir o plantarse, o como en este caso si se pasa de 21, proseguirá automáticamente al turno del dealer y el usuario se declarara como perdedor, perdiendo asi el dinero apostado.

**10)**

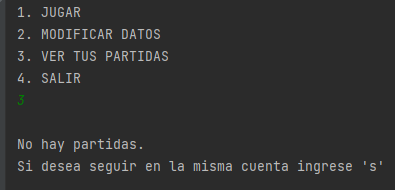


En caso de seleccionar la opción 2 (modificar datos), se abrirá un menú en el cual nos va a preguntar cual dato es el que deseamos modificar, si el nombre de usuario, el nombre, la contraseña o el email. Seleccionando una de estas cuatro opciones podremos cambiar el dato elegido.



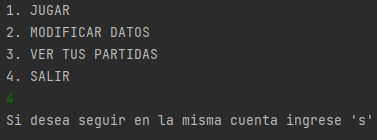
Por ejemplo, en este caso eligiendo la opción uno, cambiamos el nombre de usuario y automáticamente se nos guarda el que elegimos.

**11)**



En el caso de elegir la opción 3, se nos mostrará nuestro historial de partidas jugadas, si no las hay, el mensaje será “No hay partidas.

**12)**



En caso de seleccionar la opción 4, saldremos de la aplicación.

**DIARIO DE TRABAJO**

|  |  |
| --- | --- |
| Fecha | Detalle |
| 29/06/2023 | Primer encuentro grupal para defirnir en que se va a trabajar, cuales serán los pasos generales a seguir y designar las primeras tareas. \* Se definen la temática del trabajo. Se decide diseñar un juego de BlackJack. \* Se decide comenzar a diseñar las clases \* Se analiza la factibilidad del proyecto |
| 01/06/2023 | \* Se investiga información sobre el juego de Blackjack  \* Se decide armar el diagrama de clases para visualizar la funcionalidad del programa y dividir las tareas. |
| 07/06/2023 | Se realiza la división de funciones a codificar y tareas:   1. Mazo 2. Carta 3. Enums Palo y Rango 4. Partida 5. Jugador 6. Usuario 7. Dealer 8. Clase Genérica Pila 9. Clase Genérica Mapa 10. Controladora Archivos 11. Controladora JSON 12. Controladora Usuario (ABM (carga controlada) 13. Controladora Juego 14. Interfaz 15. Menú |
| 15/06/2023 | Se crearon las principales clases del código:   1. Usuario 2. Dealer 3. Mazo 4. Carta 5. Partida 6. Genérica Map 7. Genérica Pila 8. Interfaz 9. Controladora Usuario |
| 17/06/2023 | Se busco una forma de implementar una api y se encontraron dos apis para cargar usuarios. Esta información sirve para probar las funcionalidades del juego. Se continuo desarrollando las clases controladoras y complementarias:   1. Controladora Archivos 2. Controladora JSON 3. Controladora Usuario 4. Controladora Juego 5. Menú |
| 18/06/2023 | Se solucionaron problemas de funcionamiento. Se unifico el código y se siguieron desarrollando funcionalidades. |
| 22/06/2023 | Se probaron las funcionalidades del juego y se hizo toda la documentación necesaria. Se guardo la última versión del programa. |

**Trello:** [https://trello.com/b/7v562oAb/tareas](https://trello.com/b/7v562oAb/tareas" \o "https://trello.com/b/7v562oAb/tareas" \t "https://discord.com/channels/@me/_blank)

**UML:** [https://miro.com/welcomeonboard/MkM2dXBRbjhpZkpSUGJVdGZ3VFRoRzVZRnprdVgyNHhFcnZId2tvczNQMjRlNWdCQ1ZoT3dGRjlzTzloTlk4eHwzNDU4NzY0NTU3NzkxNjkzNjM3fDI=?share\_link\_id=484026693622](https://miro.com/welcomeonboard/MkM2dXBRbjhpZkpSUGJVdGZ3VFRoRzVZRnprdVgyNHhFcnZId2tvczNQMjRlNWdCQ1ZoT3dGRjlzTzloTlk4eHwzNDU4NzY0NTU3NzkxNjkzNjM3fDI=?share_link_id=484026693622" \o "https://miro.com/welcomeonboard/MkM2dXBRbjhpZkpSUGJVdGZ3VFRoRzVZRnprdVgyNHhFcnZId2tvczNQMjRlNWdCQ1ZoT3dGRjlzTzloTlk4eHwzNDU4NzY0NTU3NzkxNjkzNjM3fDI=?share_link_id=484026693622" \t "https://discord.com/channels/@me/_blank)

**Fuentes de información consultadas**

[https://www.edrawsoft.com/es/article/uml-class-diagram-explained.html](https://www.edrawsoft.com/es/article/uml-class-diagram-explained.html" \o "https://www.edrawsoft.com/es/article/uml-class-diagram-explained.html" \t "https://discord.com/channels/@me/_blank)

[https://www.casino.org/es/blackjack/como-jugar/](https://www.casino.org/es/blackjack/como-jugar/" \o "https://www.casino.org/es/blackjack/como-jugar/" \t "https://discord.com/channels/@me/_blank)