TEC | Tecnológico de Costa Rica

IC 2101 Programación Orientada a Objetos

Proyecto Bingo

Integrantes

Oscar Alberto Roni Ordoñez

Jefferson Pozo Vallejos

Jozafath Josué Fernández Pérez

Docente:

Luis Pablo Soto Chaves

Grupo 60

Semestre II 2023

2023

Tabla de contenido

Introducción

El proyecto "Gestor de Bingos" surge en respuesta a la necesidad de adaptar un juego clásico como el bingo, donde la interacción presencial se ve limitada. Tradicionalmente, el bingo ha sido un juego de azar muy popular que involucra la extracción de bolas numeradas y la coincidencia de números en cartones de juego. A medida que el mundo evoluciona hacia una mayor virtualidad y conectividad, es fundamental encontrar soluciones para permitir que las personas continúen disfrutando de este juego de manera remota.

El juego de bingo, con sus reglas y dinámicas específicas, plantea una serie de desafíos que requieren una solución de software efectiva y eficiente. El objetivo de este proyecto es diseñar, implementar y documentar una solución de software que permita la gestión completa de un juego de bingo en un entorno virtual.

Metodología de trabajo

Este informe presenta una metodología de trabajo para el desarrollo del proyecto "Gestor de Bingos". El objetivo de esta metodología es abordar el proyecto de manera eficiente, garantizando la recopilación de información necesaria y la entrega de un producto completo y funcional. Esta metodología combina enfoques de investigación y desarrollo de software ágil.

Herramientas utilizadas:

Para llevar a cabo este proyecto, se utilizarán diversas herramientas que permitirán a nuestro equipo trabajar de manera colaborativa y efectiva. Estas herramientas incluyen:

- **Entorno de Desarrollo Integrado (IDE)**: Se utilizará el entorno de Visual Studio Code, incorporando con una extensión de Java para escribir y compilar el código de la aplicación.
- **Herramientas de Control de Versiones:** El uso de la herramienta GitHub (Repositorio) para el sistema de control de versiones para el seguimiento de los cambios en el código y la colaboración entre miembros del equipo.
- **Pizarra virtual:** Utilizaremos Excalidraw (Pizarra) para el planeamiento de ideas y abstracción del proyecto, de cómo implementarlo y que deberíamos de tomar en cuenta.
- **Herramientas de Comunicación**: mensajería instantánea (como Microsoft Teams o WhatsApp) y videoconferencias serán esenciales para la comunicación regular entre los miembros del equipo.

El desarrollo del proyecto se basará en una metodología ágil, que permite una entrega continua de funcionalidades y ajustes a medida que avanza el proyecto. El proceso se dividió en los siguientes ciclos:

Planificación y Diseño Inicial: En esta fase, se definirán los requisitos del proyecto, se realizará un análisis de casos de uso y se diseñará la arquitectura de la aplicación. También se elaborarán los primeros borradores de la interfaz de usuario.

Desarrollo Iterativo: Trabajaremos en incrementos de funcionalidades específicas, siguiendo un enfoque de programación orientada a objetos. Estos incrementos se someterán a pruebas exhaustivas antes de avanzar.

Pruebas y Retroalimentación: Cada incremento se someterá a pruebas para identificar errores y problemas. Los miembros del equipo proporcionarán retroalimentación, lo que permitirá realizar ajustes y mejoras.

Integración y Despliegue: A medida que se completen incrementos, se integrarán en la aplicación principal y se realizará un despliegue continuo en un entorno de desarrollo o pruebas.

Documentación y Manuales de Usuario: Se elaborará la documentación técnica y los manuales de usuario de forma paralela al desarrollo para garantizar una documentación completa y actualizada.

Revisión y Entrega Continua: Se programarán revisiones regulares entre el grupo para obtener retroalimentación. Los incrementos se subirán continuamente a nuestro repositorio de GitHub.

Cronograma de trabajo

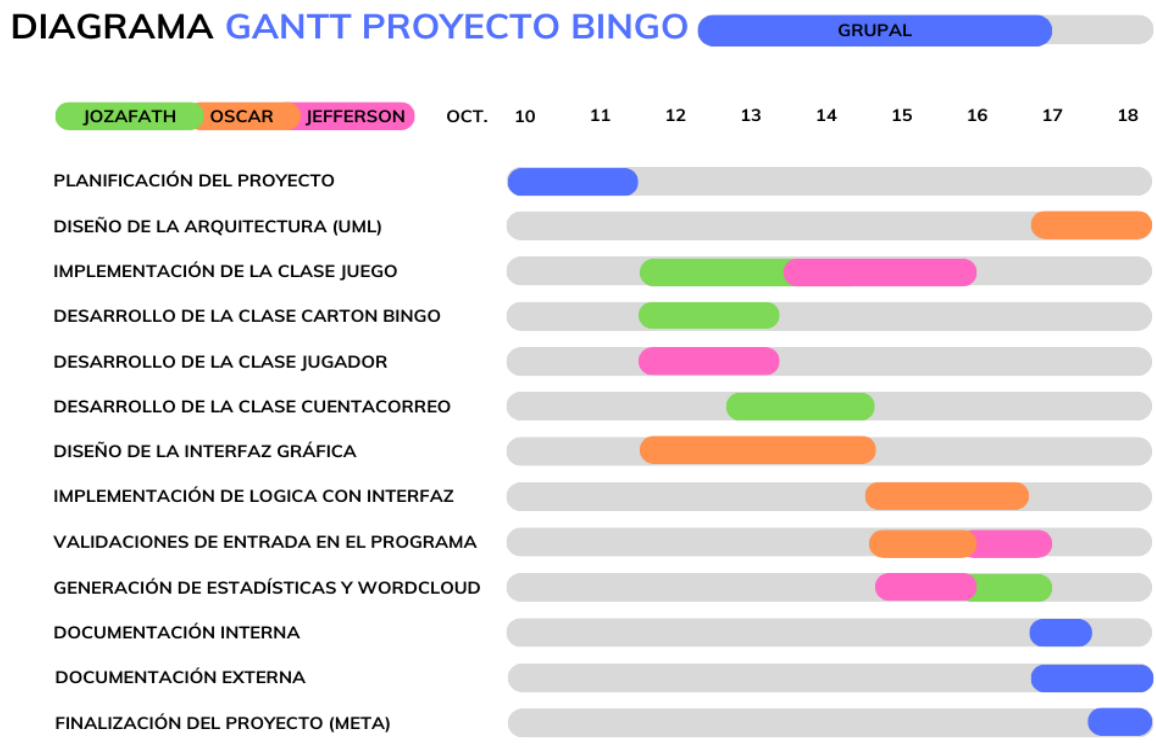


Diagrama de clases de bajo nivel

Diagrama de Clase de Bingo.pdf

Diagrama de paquetes

Diagrama de Paquetes.pdf

Análisis de Resultados

En el desarrollo de este proyecto, se han llevado a cabo diversas tareas que se planificaron inicialmente. A continuación, se presenta un análisis detallado del estado de finalización de cada tarea:

Tarea 1. Diseño de la Arquitectura (UML)

- Porcentaje de realización: 100%
- Justificación: Se elaboró el diseño de la arquitectura utilizando UML, detallando la estructura del sistema.

Tarea 2. Implementación de la Clase Juego

- Porcentaje de realización: 100%
- Justificación: La clase Juego se implementó según los requisitos y se probó con éxito.

Tarea 3. Desarrollo de la Clase CartonBingo

- Porcentaje de realización: 100%
- Justificación: La Clase CartonBingo se desarrolló y probó satisfactoriamente.

Tarea 4. Desarrollo de la Clase Jugador

- Porcentaje de realización: 100%
- Justificación: La clase Jugador se desarrolló completamente y se integró en el sistema.

Tarea 5. Desarrollo de la Clase Cuentacorreo

- Porcentaje de realización: 100%
- Justificación: La clase Cuentacorreo se desarrolló y se incorporó al sistema sin problemas.

Tarea 6. Diseño de la Interfaz Gráfica

- Porcentaje de realización: 100%

- Justificación: Se diseñó la interfaz gráfica de acuerdo con los requisitos y se implementó con éxito.

Tarea 7. Implementación de Lógica con Interfaz
- Porcentaje de realización: 100%
- Justificación: La lógica del juego se integró con la interfaz gráfica y se probó con éxito.

Tarea 8. Validaciones de Entrada en el Programa
- Porcentaje de realización: 100%
- Justificación: Se implementaron validaciones para garantizar la integridad de los datos de entrada.

Tarea 9. Generación de Estadísticas y WordCloud
- Porcentaje de realización: 100%
- Justificación: La generación de estadísticas y WordCloud se completó con éxito como se planeó.

El proyecto ha concluido satisfactoriamente, y todas las tareas se han realizado al 100% de acuerdo con lo planificado. Se ha logrado implementar el juego de bingo con todas las funcionalidades requeridas y documentación adecuada.

JavaDoc

JavaDoc\index.html

Aspectos relevantes y lecciones aprendidas

Jozafath Pérez Fernández:

1. **Uso de la biblioteca Kumo:** Aprendí a utilizar la biblioteca Kumo para generar nubes de palabras a partir de los comentarios de correo electrónico. Esta herramienta fue esencial para procesar datos de texto y visualizar patrones de palabras clave en los comentarios.

2. **Integración de servicios de correo electrónico:** Descubrí cómo conectar nuestra aplicación a un servidor de correo electrónico (Gmail) para descargar los comentarios de los usuarios. Esto fue crucial para recopilar datos de entrada del mundo real.

3. **Implementación de persistencia de datos:** Aprendí que el diseño de la base de datos influye en la eficiencia y en la capacidad de recuperar información de manera efectiva, me gusto trabajar con el archivo de xml de partidas y jugadores, ya que es el primer proyecto donde se implementa una memoria de los datos y no solo temporal.

4. **Desarrollo de API de terceros:** Descubrí cómo integrar APIs externas en nuestro software, como la biblioteca Kumo, para ampliar las capacidades de la aplicación.

5. **Herencia de paneles:** Aprendí a utilizar la herencia para crear paneles personalizados que compartieran características comunes, como disposición y comportamiento.

Oscar Roni Ordoñez

1. **Control de excepciones:** Para realizar ciertas validaciones fue necesario investigar y utilizar el sistema de excepciones de java.
2. **Necesidad de ejecución en paralelo:** Note como la interfaz se detenía en ocasiones en lo que esperaba que algunas funciones de la lógica se terminarán de ejecutar. Aunque en este caso nos beneficia que fuera así, sigue siendo una experiencia que va a ser valiosa en un futuro.
3. **jerarquía de clases en objetos de interfaz:** Aprendi a como se puede utilizar este método de herencia para crear objetos de la interfaz personalizados, de forma que se ahorra mucho trabajo.
4. **Clase mediadora interfaz-lógica:** En mi intento por unir la interfaz con la lógica me topé con varios problemas al relacionar las clases. Esto pudo haber sido arreglado con una clase planteada desde el principio para servir como intermediario entre esta relación.
5. **VS Code para Java:** La elección de nuestro entorno de trabajo nos mostró la flexibilidad de VS Code como editor de código, sin embargo, también mostró sus limitantes a la hora de realizar tareas muy específicas de Java.

Jefferson Pozo Vallejos:

1. **Manejo de librerías de terceros:** Implementa funcionalidades aprovechando el código ya realizado por otras personas, esto me permitió obtener soluciones a problemas de una manera más rápida y eficaz.
2. **Generación de documento XML:** Utilice la herramienta JDOM2 para la creación de documentos XML de una manera sencilla y eficaz, lo cual lo mantengo presente para futuros proyectos.
3. **Ventaja de un IDE:** Conocí las bondades que puede ofrecer un IDE a comparación de un editor de texto como VS Code, durante el desarrollo cosas pequeñas y simples se podían complicar en especial si no se tiene mucha experiencia en el lenguaje así como en manejo de dependencias.
4. **Generación de gráficos:** Aprendí cómo crear gráficos de una manera sencilla utilizando la librería jfree.
5. **Manejo de interfaces gráficas:** Obtuve nociones de la creación e integración de interfaces en java, con el uso de la librería swing.

Bitácora o diario de trabajo

Jozafath Pérez Fernández : https://blogdejozabingo.blogspot.com/
Oscar Roni Ordoñez : https://bingooscar.blogspot.com
Jefferson Pozo Vallejos : https://jeff-pozo.blogspot.com

Bibliografía

JFreeChart. (s. f.). https://www.jfree.org/jfreechart/

Kennycason. (s. f.). *GitHub - kennycason/kumo: Kumo - Java Word Cloud*. GitHub. https://github.com/kennycason/kumo

JavaMail API - Quick Guide. (s. f.). https://www.tutorialspoint.com/javamail_api/javamail_api_quick_guide.htm

Java (s. f.).*Swing Documentation.* https://docs.oracle.com/javase%2F7%2Fdocs%2Fapi%2F%2F/javax/swing/package-summary.html

Geeks for Geeks. (marzo de 2023) *Introducción a Java Swing.* *https://www.geeksforgeeks.org/introduction-to-java-swing/*

Manual de usuario

Esta sección tiene como objetivo proporcionar una guía detallada y precisa sobre cómo ejecutar y utilizar el programa de manera efectiva. A lo largo de esta sección, se incluyen instrucciones paso a paso, capturas de pantalla y comentarios para mostrar las partes y funcionalidades del programa desarrollado.

Requerimientos del programa

Para asegurar el funcionamiento adecuado del programa, es necesario contar con **Java** en su **versión 20.0.2** instalado en su equipo. Además, se requiere **Visual Studio Code** junto con la extensión **Pack for Java** para poder compilar el programa. Todas las dependencias esenciales se encuentran en las carpeta lib del proyecto, por lo que se recomienda no realizar modificaciones en la estructura de dichos documentos. El programa se puede descargar del siguiente enlace. https://github.com/JozafathPerez/Proyecto_01_JAVA

Ejecución del programa

Para la ejecución del programa se deben de seguir los siguientes pasos:

1. Abrir el proyecto en Visual Studio Code, asegurándose de tener la configuración adecuada para Java y abrir específicamente la carpeta **Proyecto_01_JAVA-main**, en ella se deben de encontrar los archivos del programa y una carpeta .**vscode**, la cual contiene la configuración necesaria para la gestión de dependencias.
2. Buscar el archivo “App.java” en la estructura del proyecto.
3. Dar clic derecho en el archivo para abrir el menú contextual
4. En el menú contextual se debe buscar y seleccionar la opción “Run Java” o una opción similar, la cuál permitirá la ejecución del programa.

Instrucciones de juego

Antes de poder iniciar una partida es necesario realizar pasos previos que se detallan a continuación.

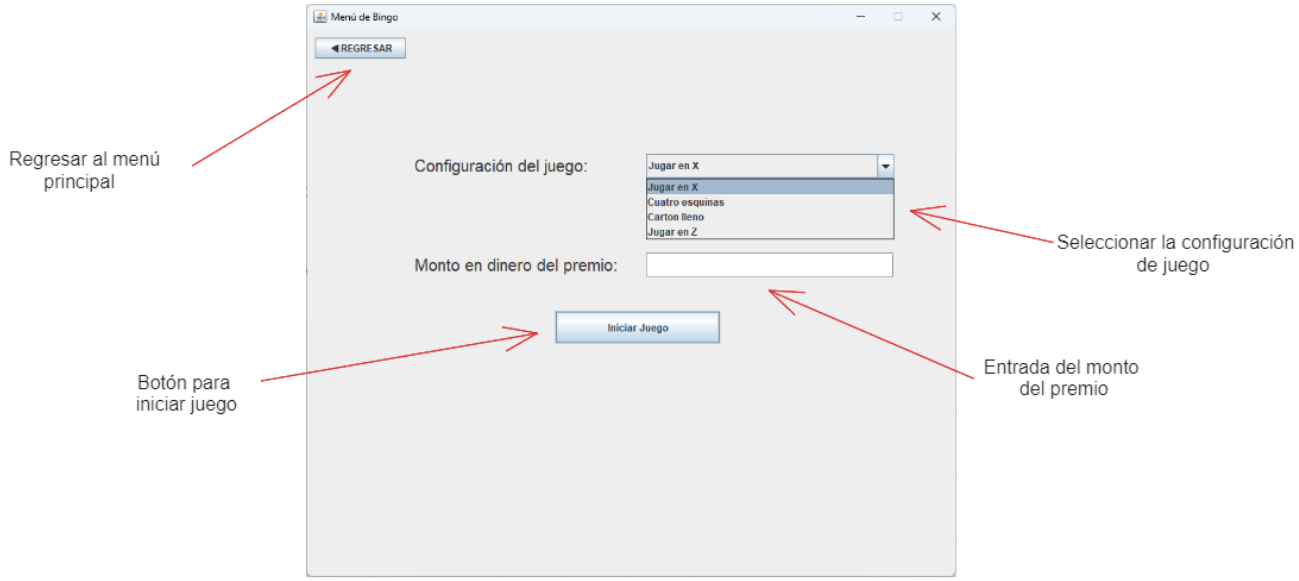
1. Crear los cartones de juego: si no existen cartones creados no se podrá iniciar una partida.
2. Añadir jugadores: Si no existen jugadores registrados es necesario registrar nuevos, una vez agregado el jugador no es necesario volver a realizarlo, ya que se guarda en los archivos internos del programa.
3. Enviar cartones: Para poder enviar un cartón es necesario tener cartones creados, además de jugadores registrados, los cartones se asignan únicamente a un jugador y realiza de manera aleatoria, es importante tener en cuenta que cada vez que se generen cartones nuevos, los cartones enviados dejan de estar en juego, por lo que es necesario volver a enviar los cartones al jugador, el programa no podrá iniciar si no cuenta con al menos un jugador con cartones asignados.
4. Iniciar partida: Una vez de haber cumplido los pasos anteriores, es posible iniciar una partida, al momento de hacerlo es necesario indicar un monto de premio y configurar la modalidad del juego, una vez realizado esto, la partida iniciara y terminara hasta que encuentre un cartón ganador, los cartones ganadores se mostraran en pantalla y el jugador favorecido será notificado mediante correo.

GUI del programa

Una vez ejecutado el programa se desplegará la ventana del menú principal, el cuál contiene 6 botones para acceder a las funciones del programa, las cuales se explican a continuación.

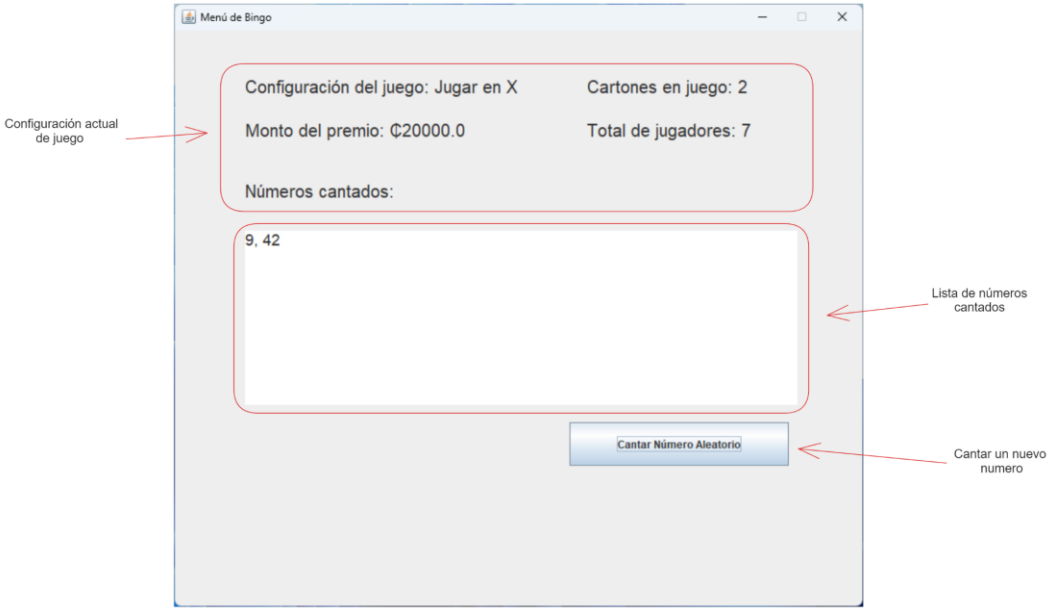
A continuación se detalla las funciones que despliega cada botón:

Iniciar Juego
Esta sección permite crear una nueva partida, en la cuál se deben elegir el monto del premio que estará en juego, cabe aclarar que si no existen cartones creados o no se han asignado cartones a jugadores la partida no podrá iniciar.



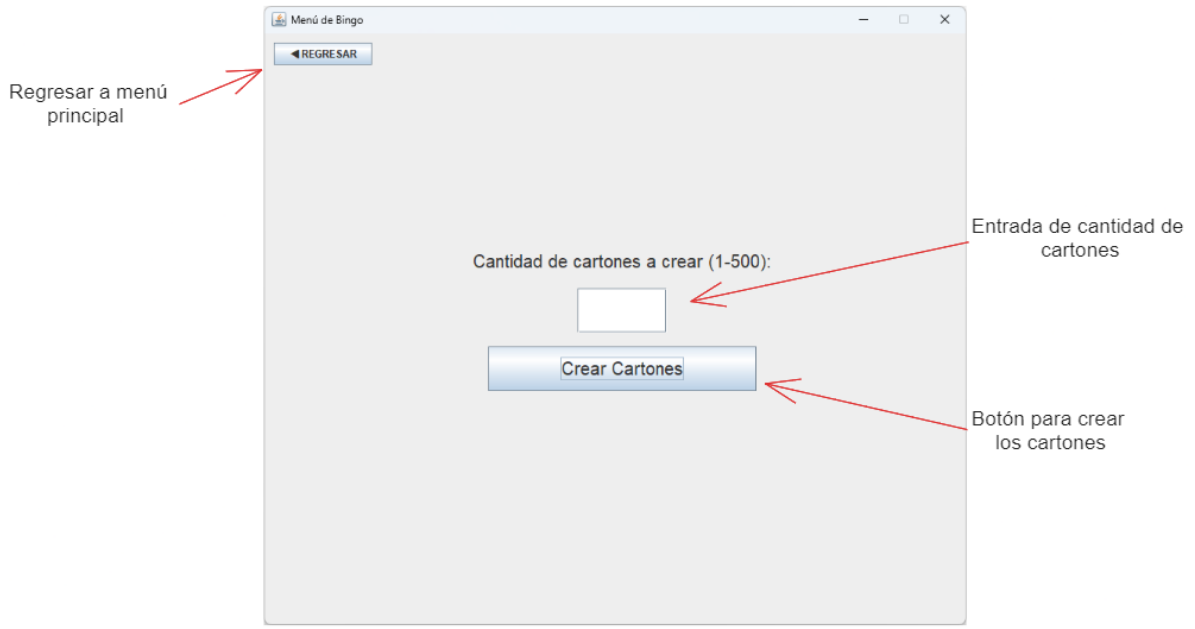
Una vez iniciado el juego, se mostrará el siguiente panel, que se mantendrá hasta encontrar a un ganador, en esta instancia la unica accion que se puede realizar es cantar los números

cada vez que se cante un número el sistema revisará si existe un cartón ganador, en caso de encontrar se informará el cartón favorecido, se enviará un correo a los favorecidos y por último retorna al menú principal.



Crear Cartones

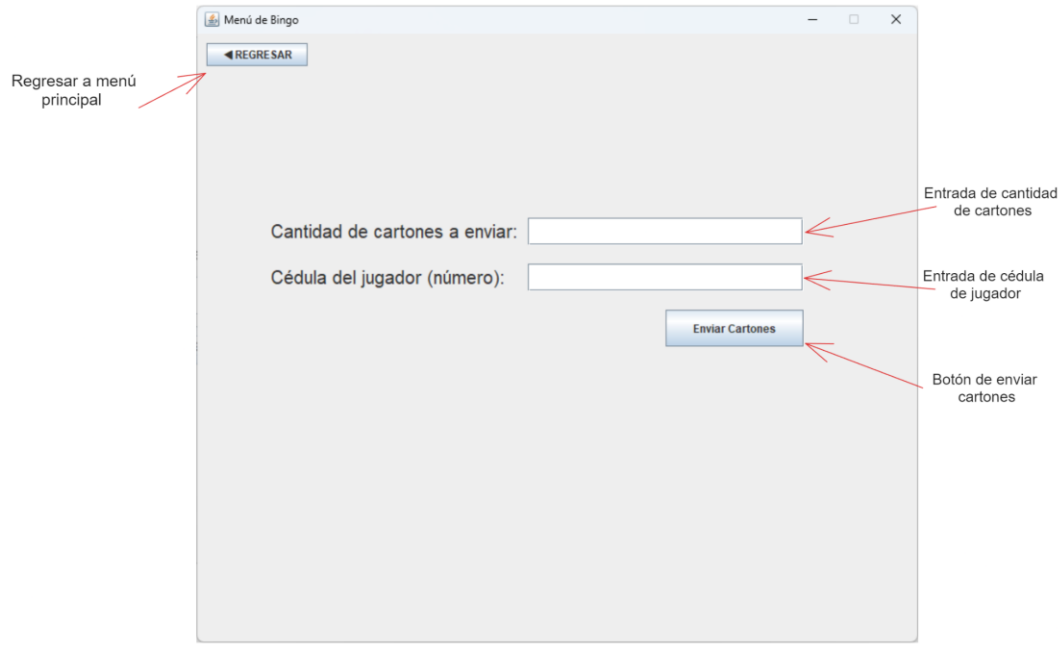
Permite crear cartones nuevos, se deberá ingresar la cantidad de cartones que debe ser mayor a 1 y menor a 500, cada vez que se creen cartones nuevos se eliminarán todos los anteriormente creados, por lo que también será necesario volver a enviar cartones a los jugadores.



Enviar Cartones

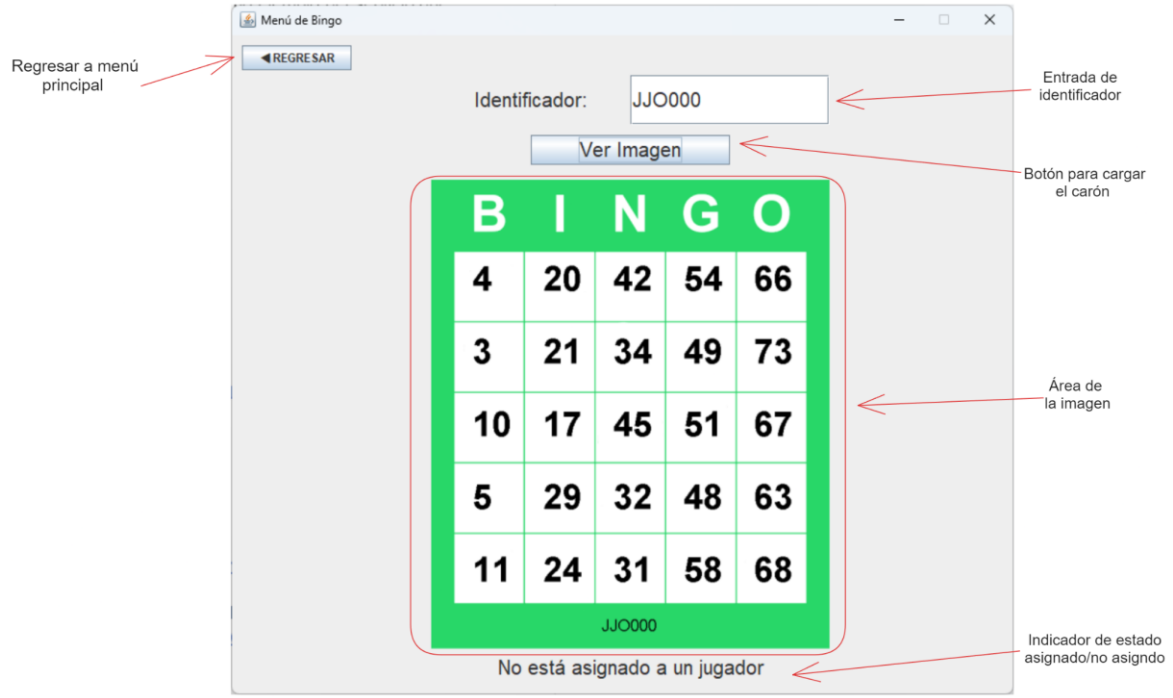
Permite enviar cartones a jugadores mediante correo electrónico, cada vez que se envía un cartón se le asigna dicho cartón al jugador, por lo que se disminuye la cantidad de cartones

disponibles para enviar. Es importante tomar en cuenta que el jugador debe estar previamente registrado, en caso de ingresar una cédula de jugador inexistente no se realizará la acción.



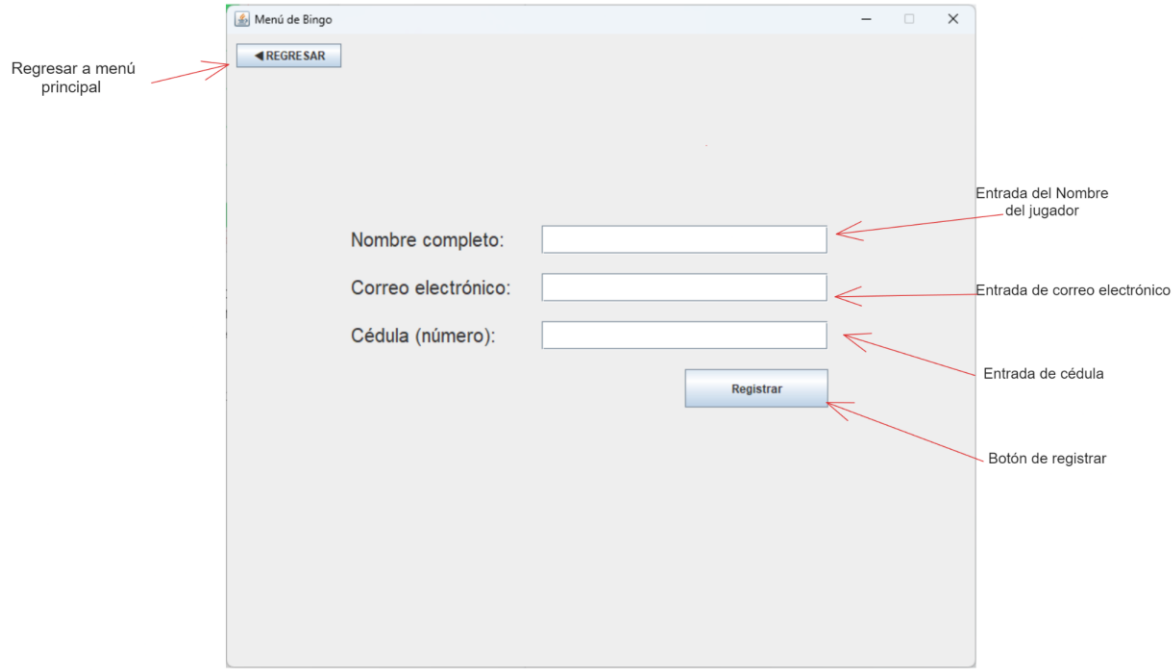
Consultar Cartones

Proporciona un panel para consultar los cartones y mostrarlos en la pantalla, se deberá ingresar el identificador de un cartón existente, además se indicará si el cartón se le fue asignado a un jugador.



Registrar Jugador

Permite el registro de nuevos jugadores, para registrar un nuevo jugador, se debe ingresar el nombre de este, un correo electrónico válido y la identificación esta última debe ser única, es decir que no se puede repetir una identificación o cédula que ya fue ingresada en el sistema.

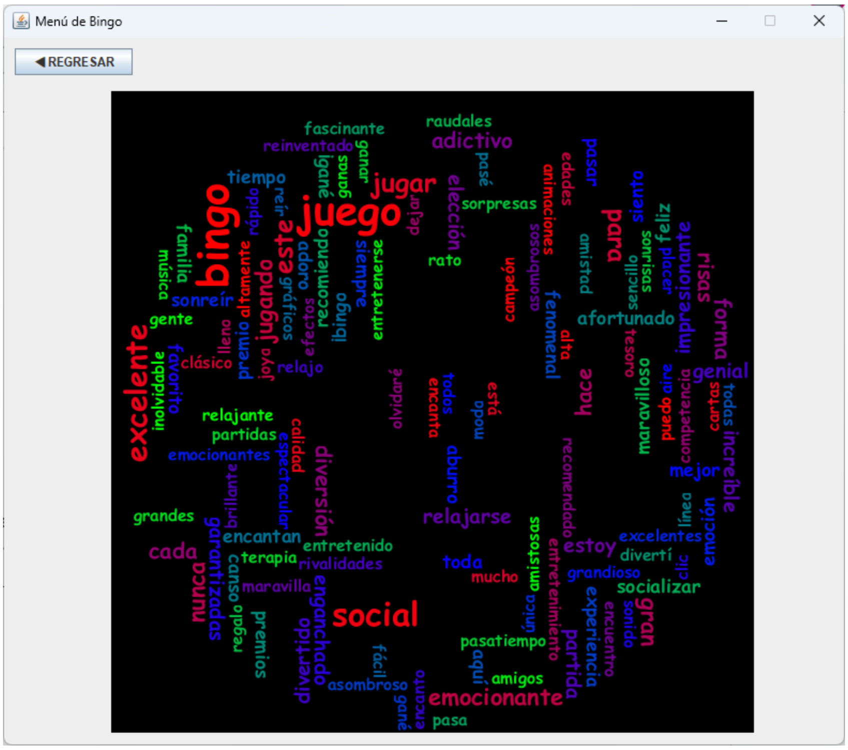


Estadísticas

Genera y muestra estadísticas de los 10 números más cantados y el porcentaje de modalidades de juego utilizadas a lo largo de todas las partidas realizadas hasta el momento.

WordCloud

Crea un WordCloud a partir de todos los mensajes que fueron enviados a la cuenta de correo administrador del sistema.


Menú de Bingo
REGRESAR
juego
bingo
social
excelente
jugar
jugando
este
adictivo
emocionante
relajarse
diversión
para
hace
fascinante
raudales
reinventado
tiempo
sorpresas
rato
afortunado
impresionante
risas
forma
genial
maravilloso
increíble
mejor
estoy
socializar
gran
partida
experiencia
pasatiempo
amigos
asombroso
divertido
enganchado
premios
nunca
cada
grandes
encantan
entretenido
terapia
rivalidades
maravilla
relajante
partidas
emocionantes
inolvidable
favorito
clásico
gente
sonreír
música
familia
relajo
toda
mucho
grandioso
excelentes
diverti
emoción
pasa