INSTITUTO TECNOLÓGICO AUTÓNOMO DE MÉXICO

Proyecto alpha

¡Pégale a Riquelme!



Integrantes

Leopoldo Morante Castillo – 200821

Eduardo Velázquez Barragán – 201432

# Tabla de contenidos

[**Tabla de contenidos 2**](#_xh2a8tld05m3)

[**Definición del proyecto 3**](#_igrd4eeivdql)

[**Definición del experimento 4**](#_g6fijbonirad)

[**Proceso de Estrés 4**](#_zh5v1mu96goq)

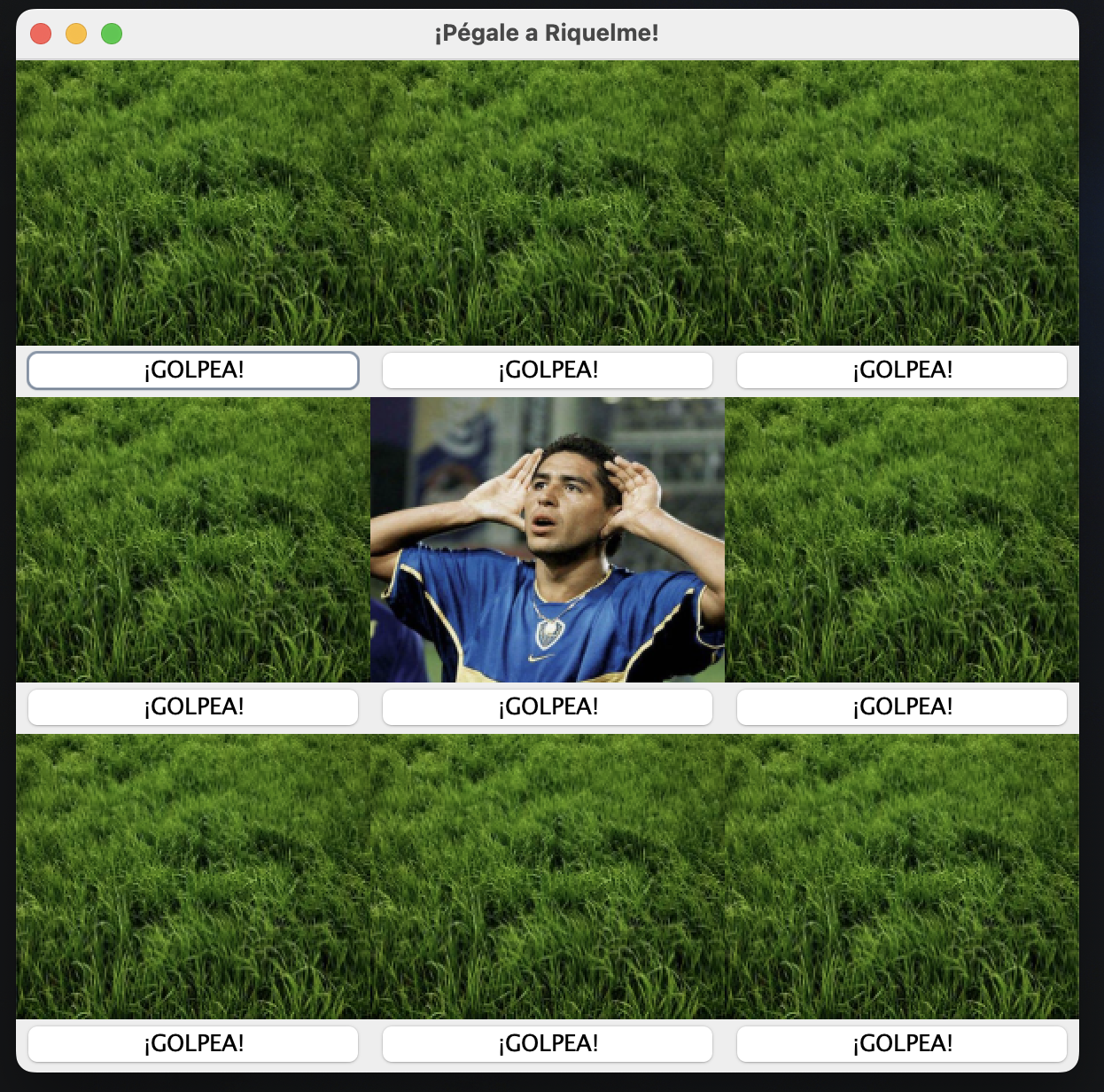
# 

# Definición del proyecto

El proyecto alpha consiste en la implementación de una versión multijugador del clásico juego Whac-A-Mole. Se tiene un servidor al cual varios jugadores pueden conectarse y desde el cual reciben información sobre dónde aparece el siguiente topo al que deben golpear. La comunicación entre clientes y servidor se realiza vía TCP sockets y vía tópicos de ActiveMQ.

El juego cuenta con:

* Capacidad de alojar partidas multijugador, dónde el primer jugador en golpear al topo enviado por el servidor obtiene un punto y aquel que alcance una cantidad de puntos objetivo recibe un mensaje de felicitación.
* Una interfaz gráfica a través de la cual el jugador puede golpear a los topos que aparecen en pantalla.
* Capacidad para soportar el ingreso y salida de usuarios durante la partida sin afectar el juego de los usuarios que permanecen en ella.



Interfaz gráfica del juego

durante una partida

# Definición del experimento

Con el objetivo de evaluar el desempeño de nuestro juego, se realizaron versiones ligeramente diferentes tanto del cliente como del servidor para ser sometidos a un proceso de estrés.

Se establecieron las siguientes condiciones para el proceso de estrés:

* Se llevarían a cabo pruebas con diferentes cantidades de usuarios que intentan acceder y jugar al juego simultáneamente. Estas cantidades serían 50, 100,150,200,300,400 y 500 usuarios.
* Para cada una de las pruebas, se realizarían 5 rondas de juego, cada una con un puntaje objetivo de 5.
* Los usuarios del juego, con el fin de llevar al límite la capacidad de procesamiento del servidor, “golpearían” al topo tan pronto como la posición de este fuera revelada por el servidor.

Las pruebas de estrés se realizaron en una única computadora que fungió como cliente y servidor.

Las computadora utilizada cuenta con las siguientes especificaciones:

* Modelo: MacBook Pro
* Chip: Apple M2 Pro
  + CPU de 8 núcleos con 4 núcleos de rendimiento y 4 de eficiencia
  + GPU de 10 núcleos
  + Neural Engine de 16 núcleos
  + 100 GB/s de ancho de banda de memoria
* Memory: 16 GB
* Sistema operativo: Sequoia 15.0

Utilizando la aplicación *Speedtest Internet & WiFi Test* se recolectó información sobre la conexión a internet:

* Bajada: 104 Mbps
* Subida: 11.4 Mbps
* Pérdida de paquetes: 0.00%
* Latencia de pings
  + Antes de la prueba: 21 ms
  + Durante el proceso de descarga: 113 ms
  + Durante el proceso de subida: 27 ms

# Proceso de Estrés

1. **¿Qué modificaciones se hicieron al código original?**

La primera modificación que se hizo fue que se eliminó por completo la interfaz gráfica en el lado del jugador. Este responde al servidor con un hit en cuanto recibe una notificación del servidor de un monstruo mediante el tópico.

La segunda modificación que se realizó es que el servidor ahora contesta a los hits con un true, una vez que procesa el hit. Esto se hace para que del lado del cliente se pueda medir la latencia de respuesta.

La tercera modificación que se realizó fue que el jugador ahora no juega de forma indefinida, sino que se le da un número fijo de rondas las cuales jugará y luego dejará de recibir mensajes.   
Finalmente, la última modificación que se realizó en el cliente es que este leyó y guardó los tiempos de respuesta del login y de los hits para poder posteriormente calcular estadísticas.

1. **Servidor de disparo del proceso de estrés (StressLauncher)**

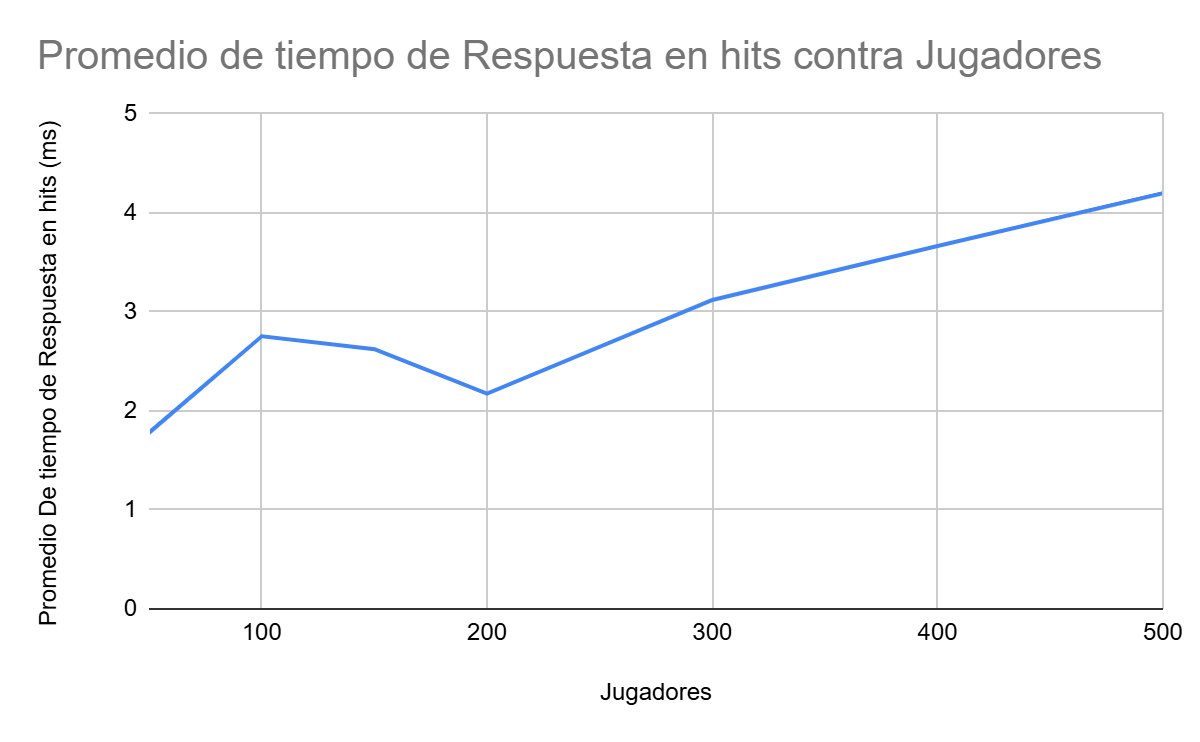
Está clase es la encargada de llevar a cabo el proceso de estrés. Lo que hace es que primero inicia el servidor, luego lanza a varios clientes a consumir el servidor, cada uno como un hilo, luego espera a que finalicen los hilos y calcula las estadísticas. La clase recibe los siguientes parámetros: número de jugadores, número de golpes para ganar en una ronda, número de rondas, el tiempo de envío entre topos y la dirección del documento donde se quiere que se impriman las estadísticas.

1. **Ejecución de las pruebas**

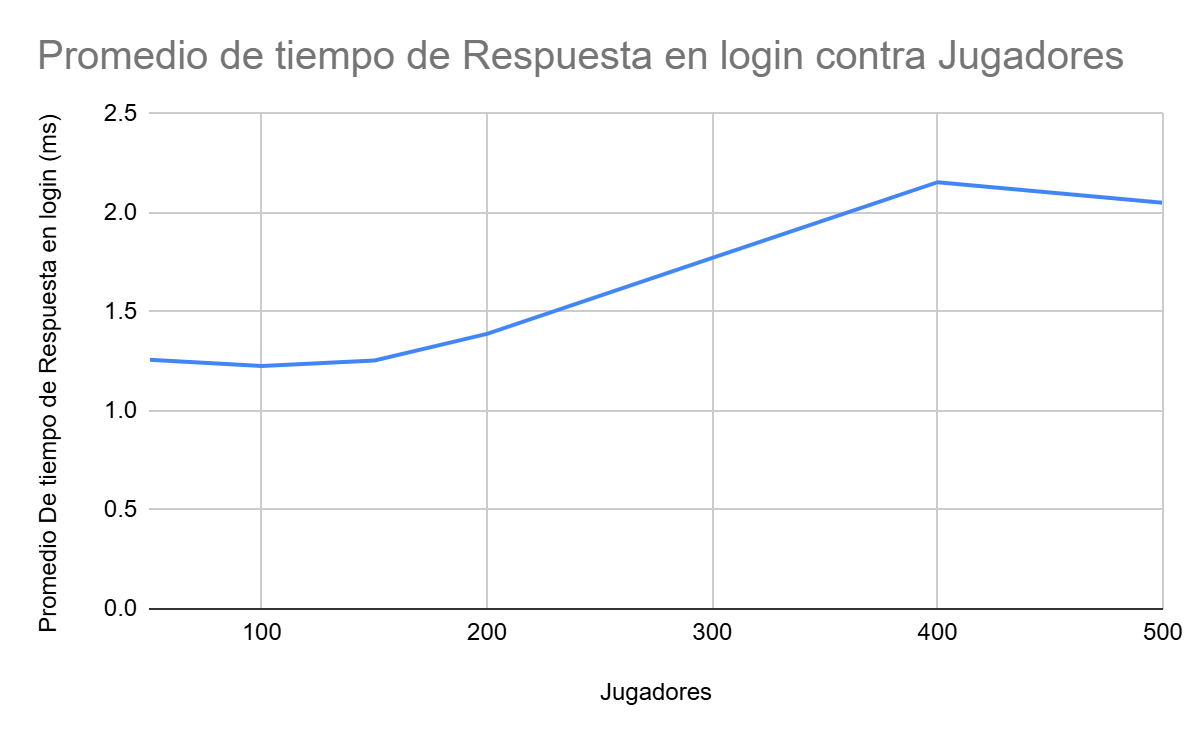
Se usó la clase de StressLauncher para estresar al servidor y obtener las estadísticas. Se corrió para 50,100,200,200,400 y 500 jugadores y para cada uno de estos valores se ejecutó 10 veces para obtener datos más confiables. Cada una de las ejecuciones se hizo con la siguiente configuración además del número de jugadores: 5 golpes para ganar, 5 rondas y 10 milisegundos entre los envíos del servidor.

1. **Resultados**

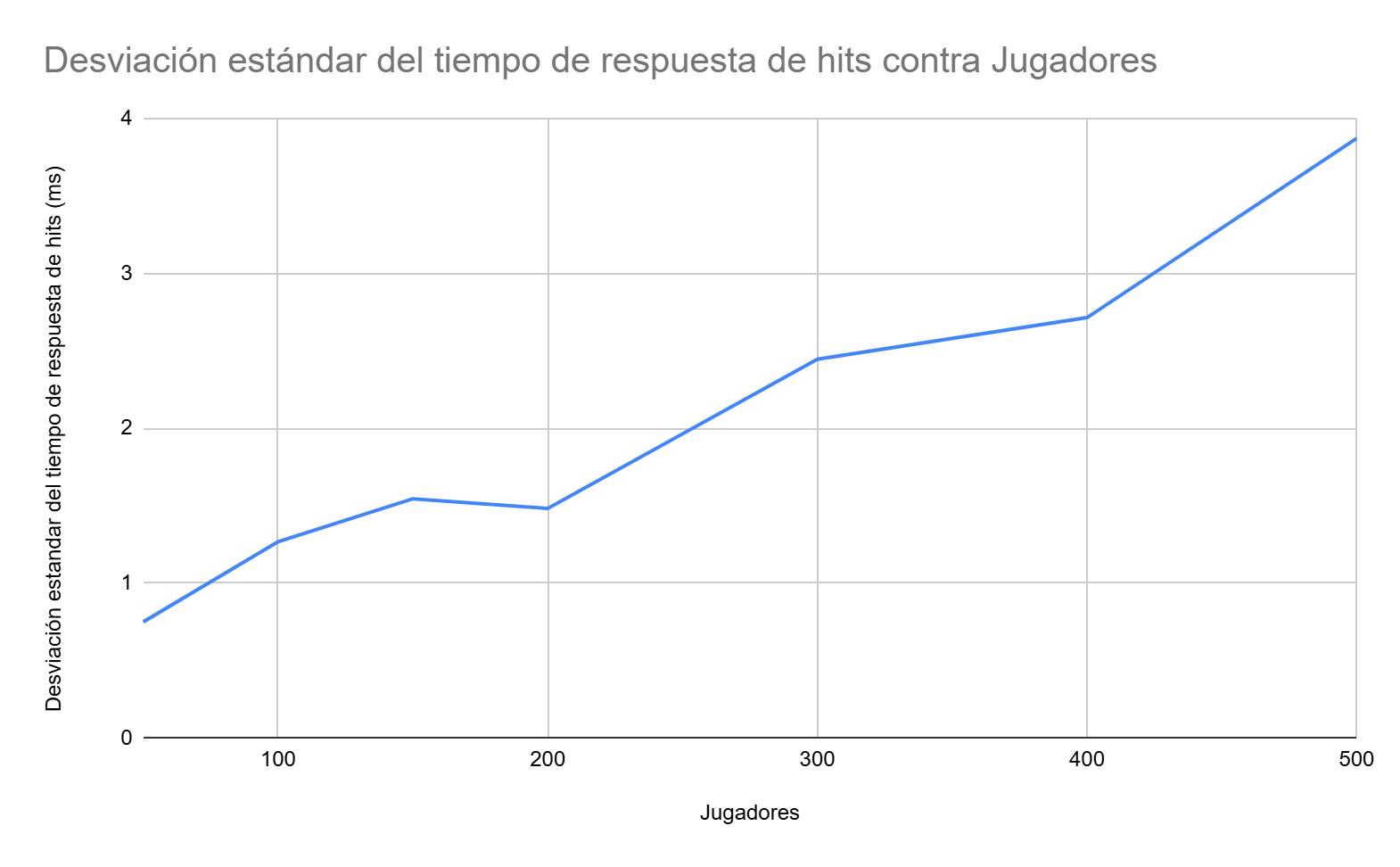
Gráfica 1



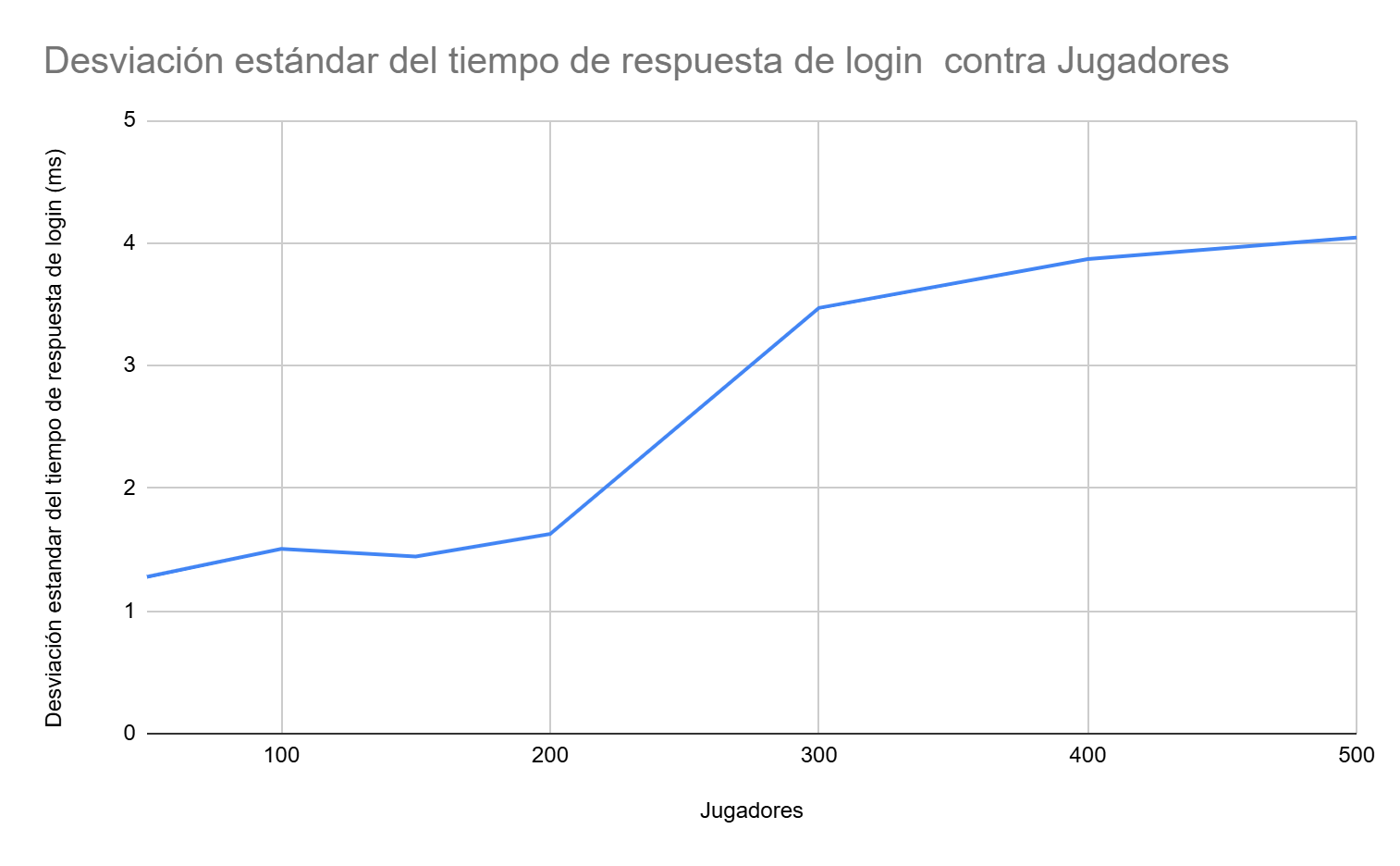
Gráfica 2



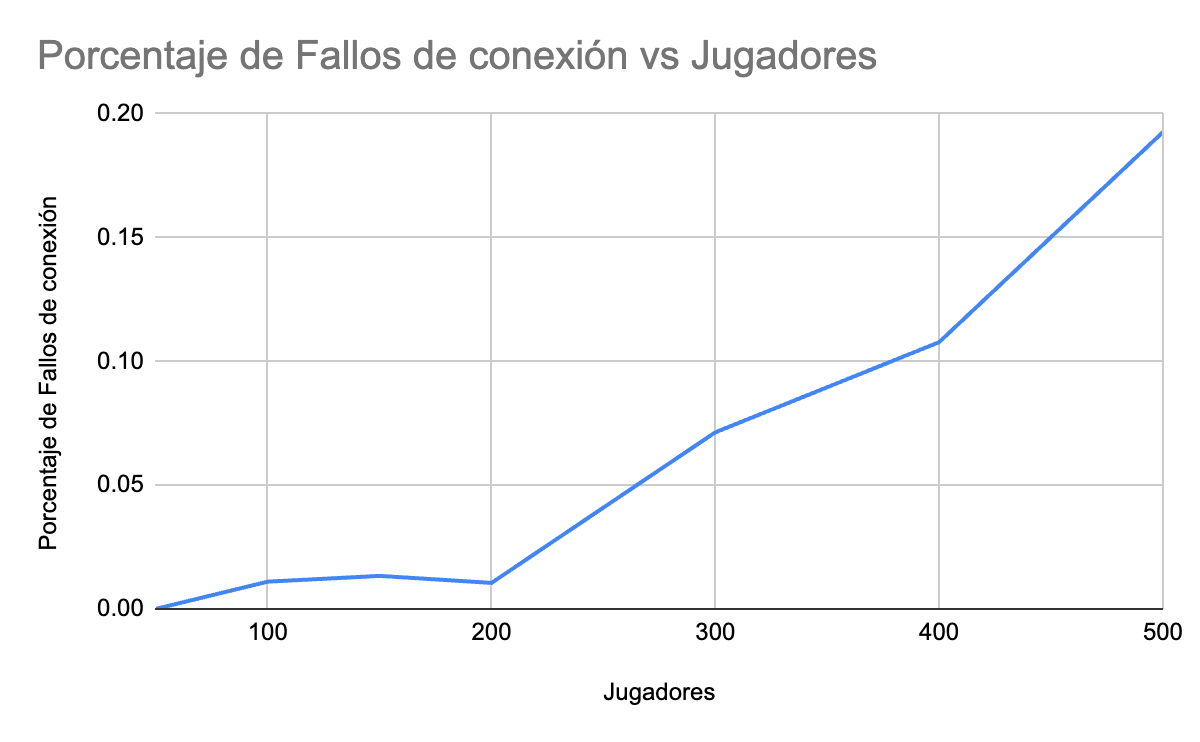
Gráfica 3

****

Gráfica 4

****

Gráfica 5

****

1. **Análisis de resultados**

La Gráfica 1, que muestra el promedio del tiempo de respuesta en hits contra el número de jugadores, evidencia que el tiempo de respuesta se mantiene bajo hasta cierto umbral de concurrencia, pero incrementa significativamente a partir de los 200 jugadores. Esto sugiere que el servidor enfrenta dificultades al procesar múltiples golpes simultáneamente sin degradar el rendimiento.

En la Gráfica 2, donde se analiza el promedio del tiempo de respuesta en el login frente a la cantidad de jugadores, se observa un comportamiento más estable. A diferencia del tiempo de respuesta de los golpes, la autenticación inicial no parece verse tan afectada por la carga del sistema, lo que indica que el proceso de login no representa un cuello de botella crítico.

La Gráfica 3, que representa la desviación estándar del tiempo de respuesta en los golpes, muestra que la variabilidad en los tiempos de respuesta aumenta conforme crece el número de jugadores. Esto sugiere que el servidor no siempre responde con la misma rapidez, posiblemente debido a la acumulación de solicitudes en la cola de procesamiento.

En la Gráfica 4, que mide la desviación estándar del tiempo de respuesta en el login, muestra una variabilidad similar a la del tiempo de respuesta de los golpes. Esto muestra que, independientemente del proceso medido, la variabilidad en tiempos de respuesta aumenta conforme lo hace la cantidad de jugadores.

Finalmente, la Gráfica 5, que muestra el porcentaje de conexiones rechazadas en función del número de jugadores, destaca que el servidor comienza a rechazar conexiones cuando la carga supera los 200 jugadores. Este fenómeno indica que, a partir de cierto punto, la infraestructura no puede manejar más solicitudes sin afectar la estabilidad del sistema.

En conclusión, el análisis de las gráficas sugiere que el servidor tiene una capacidad límite clara para procesar golpes sin degradación severa en el rendimiento, la cual se encuentra en 150 jugadores concurrentes. Hasta este punto, el tiempo de respuesta en los golpes se mantiene relativamente bajo y estable. Sin embargo, a partir de los 200 jugadores, el tiempo de respuesta en los golpes aumenta considerablemente, acompañado de una mayor variabilidad en los tiempos de respuesta, como se muestra en la Gráfica 3. Además, la Gráfica 5 indica que el servidor comienza a rechazar conexiones cuando el número de jugadores supera los 200, lo que evidencia que la infraestructura no puede manejar más solicitudes sin afectar la estabilidad del sistema.

Mientras que el proceso de login crece de manera estable conforme aumenta la cantidad de jugadores, la degradación en el procesamiento de golpes sugiere la necesidad de optimizaciones, como la implementación de un mecanismo de distribución de carga o mejoras en la gestión de las conexiones TCP abiertas, para permitir la escalabilidad del sistema más allá del umbral identificado.