

PROYECTO FINAL - RIVAL FRONTIERS



**INTEGRANTES:**

- *Diego Godoy Torres*
- *Bruno Zamora Flores*
- *Alessio Tardillo Sablich*
- *José Carlos Carita Condorhuaman*

**Curso: Programación II**

**Profesor: Rubén Rivas Medina**

**Fecha: 4 de diciembre de 2025**

## **Índice**

- Arquitectura general del sistema
- Sistema de eventos
- Balance del juego
- Casos de prueba

## **Arquitectura general del sistema:**

El sistema se estructura en cuatro módulos principales:

### **1. *game\_engine***

Actúa como el núcleo del proyecto. Sus funciones principales son coordinar el turno actual, administrar los recursos del jugador como la comida, el metal y la energía, evaluar las acciones permitidas y ejecutar la simulación de los guardianes. En resumen, este módulo se asegura de que cada fase se ejecute de manera correcta.

### **2. *map***

Representa el tablero bidimensional donde se desarrollan las interacciones. Cada celda guarda la información sobre unidades, edificios, terrenos y propiedades. El mapa expone operaciones de consulta y modificación segura mediante el comando `map.at(x, y)`, encapsulando la lógica de límites y evitando errores de acceso.

### **3. *controller***

Implementa la interfaz que conecta al jugador con la lógica del sistema. Este módulo recibe los comandos ingresados en consola y los traduce a acciones concretas dentro del motor. Además, supervisa errores de entrada, turnos y reglas de validez antes de llamar a `game_engine`.

### **4. *ux***

Este módulo gestiona la visualización en consola. Incluye herramientas para imprimir el mapa, mensajes, menús y resúmenes del estado del juego. No implementa lógica de reglas, sino únicamente la presentación.

## **Sistema de eventos:**

El sistema implementado en el proyecto final divide el turno en dos fases: fase del jugador y fase del mundo.

1. Fase del jugador

En cada turno, el jugador recibe 2 puntos de acción (PA). Con ellos puede realizar las acciones como mover unidades, construir edificios (granja, torre, cuartel), activar habilidades, capturar enclaves neutrales y atacar guardianes.

El módulo *game\_engine* valida cada acción verificando costos, condiciones de movimiento, propiedad del terreno y disponibilidad de recursos.

## 2. Fase del mundo

Durante esta fase el sistema ejecuta acciones automáticas, como por ejemplo, los movimientos de guardianes en dirección al jugador, la defensa de las estructuras, la aparición de nuevas unidades hostiles en ciertos turnos y las reacciones ante zonas capturadas.

A la par, el módulo *ux* imprime en consola los sucesos relevantes, como las construcciones exitosas, unidades derrotadas, cambios de control y acciones del sistema.

### **Balance del juego:**

El balance se centra en controlar el ritmo de expansión del jugador y limitar sus decisiones mediante recursos y eventos adversos.

#### 1. Economía

Recurso	Uso	Generación
Comida	Mantener unidades	Granjas J1Gr
Metal	Construir estructuras, reclutar	Cuarteles capturados
Energía	Habilidades especiales	Torres J1To

El *game\_engine* actualiza la producción por turno según el tipo de edificio, siguiendo la tabla del proyecto (J1Gr = +2 comida, J1Cu = +1 metal, J1To = +1 energía).

#### 2. Costos de acciones

Las acciones consumen PA y en algunos casos recursos. Los costos se basan de la siguiente tabla:

Acción	Comida	Metal	Energía
Mover unidad	0	0	0
Reclutar J1S/J1A/J1C	1	1	0
Construir J1Gr	0	1	1
Construir J1To	0	2	1
Activar J1I / desplegar ingeniero	1	0	1
Habilidad especial	0	0	1

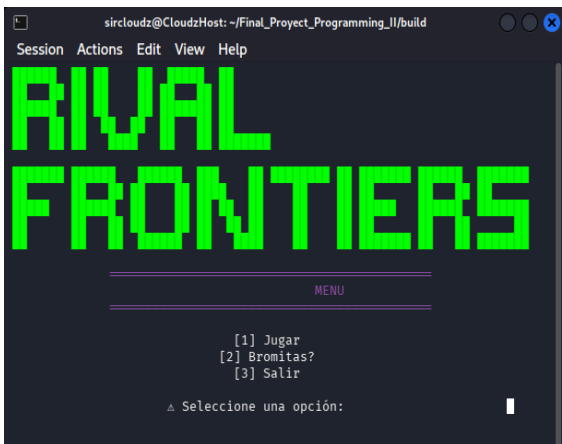
Esto garantiza que el jugador no pueda expandirse sin control y que la estrategia de administración de recursos sea central para avanzar en el juego.

3. Dificultad y presión del sistema

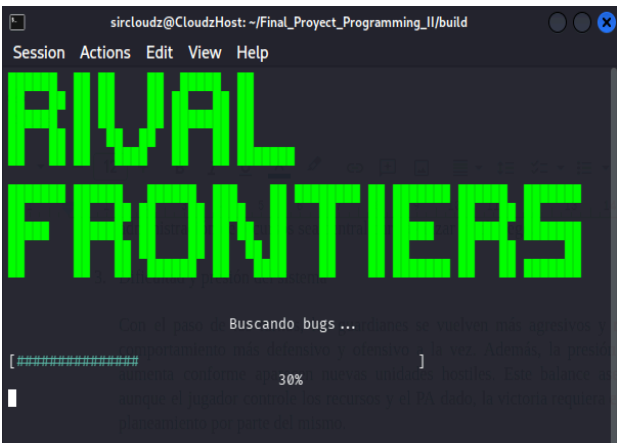
Con el paso de los turnos, los guardianes se vuelven más agresivos y replican un comportamiento más defensivo y ofensivo a la vez, aumentando la presión territorial conforme aparecen nuevas unidades hostiles. Este balance asegura que, aunque el jugador controle los recursos y el PA dado, la victoria requiera estrategia y planeamiento por parte del mismo.

Casos de prueba:

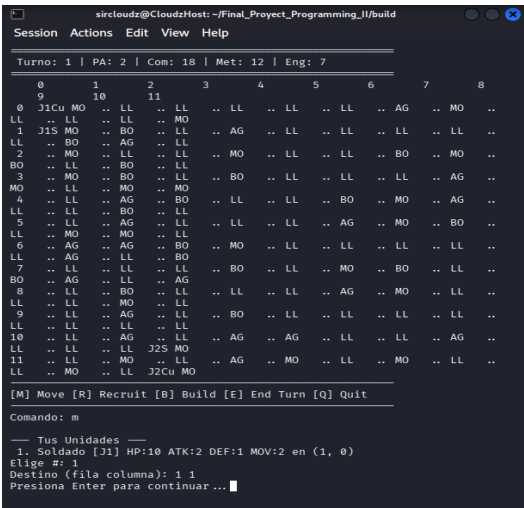
El menú principal del juego:



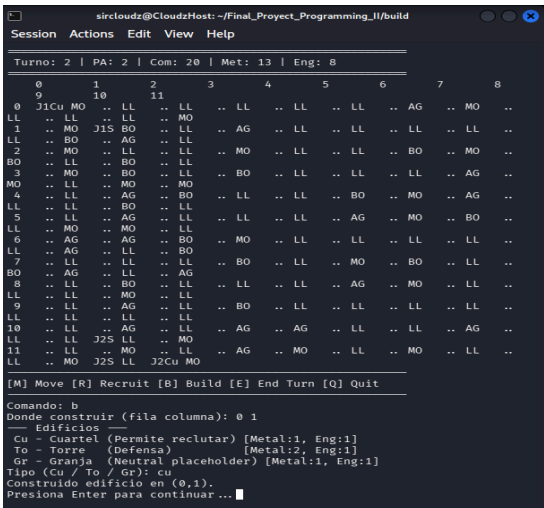
Pantalla de carga del juego:



Ejecución del juego:



Ejecución del juego:



Ejecución de salida:

```
sircloudz@CloudzHost: ~/Final_Proyect_Programming_II/build
Session Actions Edit View Help

RIVAL
FRONTIERS

Cerrando procesos ocultos ... (no preguntes cuales)

[ ##### ]
25%
```

## Diagrama de clases (Adicional):

