PROJECT PRESENTATION

R-TYPE - MULTIPLAYER SHOOT'EM UP

LANGUAGE: C++
COMPILATION: CMAKE

The goal of the R-Type project is to recreate a **multiplayer real-time space shooter,** inspired by the classic arcade game R-Type, using C++, SFML, and a custom client-server network architecture by doing a **modular game engine**. This project combines game development, real-time communication, and modular system design — with a strong focus on **object-oriented programming**, efficient rendering, and low-latency networking.

The game is split into two main parts:

- **The Server**
  - Manages all game logic, world state, enemy behavior, player actions, collisions, and synchronization.
- **The Client**
  - Renders the game using **SFML**, receives updates from the server, and sends player input in real time.

Players can connect to a game lobby, select their spaceship, and join a shared game session to fight against waves of enemies cooperatively.

- Client/Server Model
  - **TCP/UDP sockets** for real-time communication.
  - Packet serialization and deserialization.
  - Asynchronous event handling (epoll/select or custom loop).
  - **Entity Component System (ECS)** for flexible game logic.
  - **SFML** or custom graphical abstraction for rendering.

- Gameplay Features
  - Gameplay can be configured by **JSON** file.
  - Player spaceship movement & shooting.
  - Collision detection & health system.
  - Power-ups and score tracking.
  - Win/Lose conditions, respawn, etc.

- Interface
  - Main menu + Lobby system.
  - HUD: lives, score, level info.

- Multiplayer Logic
  - The server simulates the full game world.
  - The client is mostly a visual renderer + input sender.
  - Synchronization is done by sending delta packets or state snapshots at fixed intervals.

Why I like this project ?

I like this project because it felt like the culmination of my years at Epitech. Even though the final result wasn't exactly what I had envisioned, we successfully implemented all the core features, including a custom ECS architecture, real-time multiplayer with client-server synchronization, and efficient packet handling. It was a deeply technical challenge that pushed me to apply everything I had learned about networking, low-level programming, and software architecture.

Click here to see the project repository