Start

Number Guessing Game

SOCKET PROGRAMMING & RMI

Socket-Programming enables communication between devices using network sockets.RMI (Remote Method Invocation) is a Java feature to call methods on remote objects like local ones.Java RMI simplifies client-server communication with remote calls to methods like createRoom(), joinRoom(), and submitGuess().

CLOCK SYNCHRONIZATIONO

Logical Clocks (ClockSync class) track event order.Each RMI call increments the clock to maintain consistency.
When a player submits a guess, the server increments the clock before processing.
Ensures all clients see events in the correct order.

MULTIPLE NODES & LOCAL CLOCKS

Each client has its own:
Local guess list (to track new guesses).
Thread for periodic server updates.
The server maintains a global game state for each room.
Why Local Clocks?
Reduces server load by letting clients manage their own state.

DEADLOCK MANAGEMENT

Synchronized methods in GameLogic prevent race conditions.

- Example:

synchronized boolean joinRoom() ensures only one player joins at a time.
Avoiding Deadlocks:

- No nested locks; short critical sections.

MUTUAL EXCLUSION (TOKEN-BASED)

Implementation:

- Implicit token system: The server acts as a central authority.
- Only the server can:Start a game,Validate guesses,Declare a winner.

Why This Approach?

- Ensures only one client can modify game state at a time.

LOAD MANAGEMENT

Implementation:

- Thread-per-client updates (non-blocking):Clients poll the server periodically for updates.
- GameRoom instances isolate room-specific logic.

Scalability:

- New rooms can be created dynamically.

MULTITHREADING

Implementation:
Client-side thread checks for game updates every second.Server-side synchronization ensures thread-safe operations.
Example:AtomicBoolean in the client controls the update thread.
Benefit:Smooth UI experience without freezing.

Deep Salunkhe
Sahil Pokharkar
Omkar Patil
Pranav Redij