Distributed Systems

AUCTION

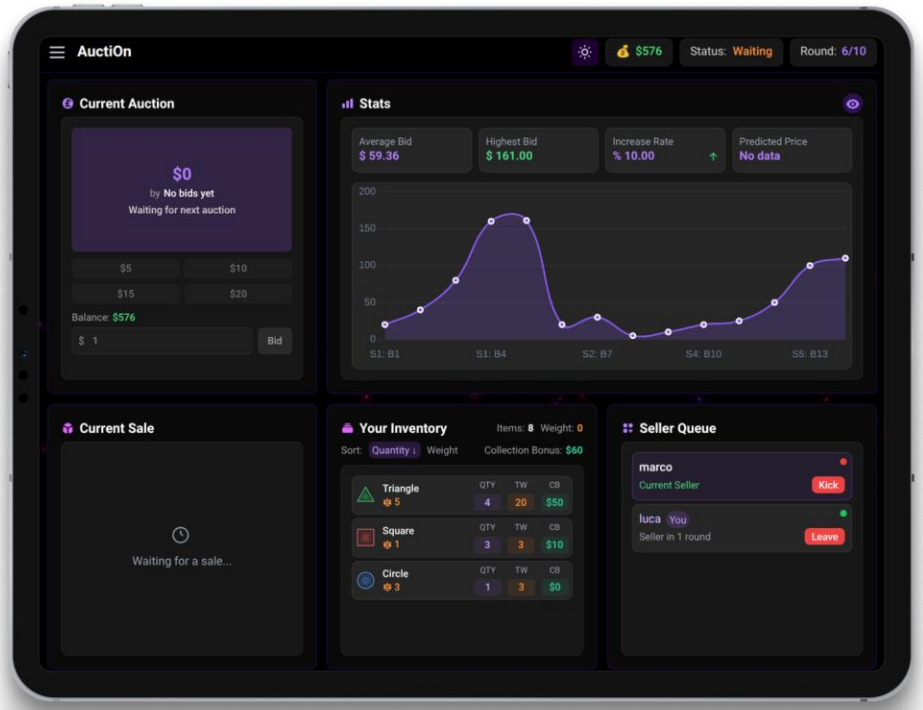
GOAL

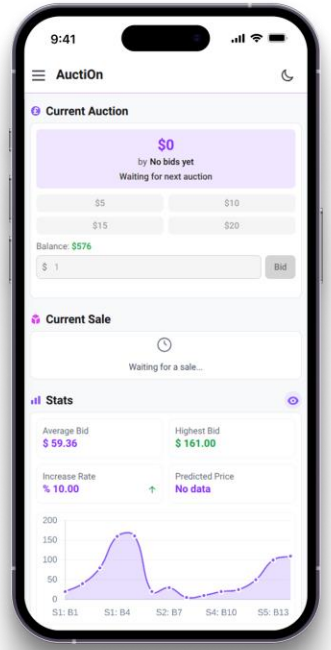
AuctiOn is a real-time, multiplayer online platform designed for strategic auction-based gameplay.

Each player begins with an initial allocation of virtual items and currency. The game proceeds through timed **rounds** where one designated player acts as the **seller**, offering items for auction. Concurrently, other players submit **bids** within the specified time limit. The highest bidder acquires the auctioned items, transferring the bid amount to the seller.

The primary goal is to maximize accumulated currency by the game's conclusion.

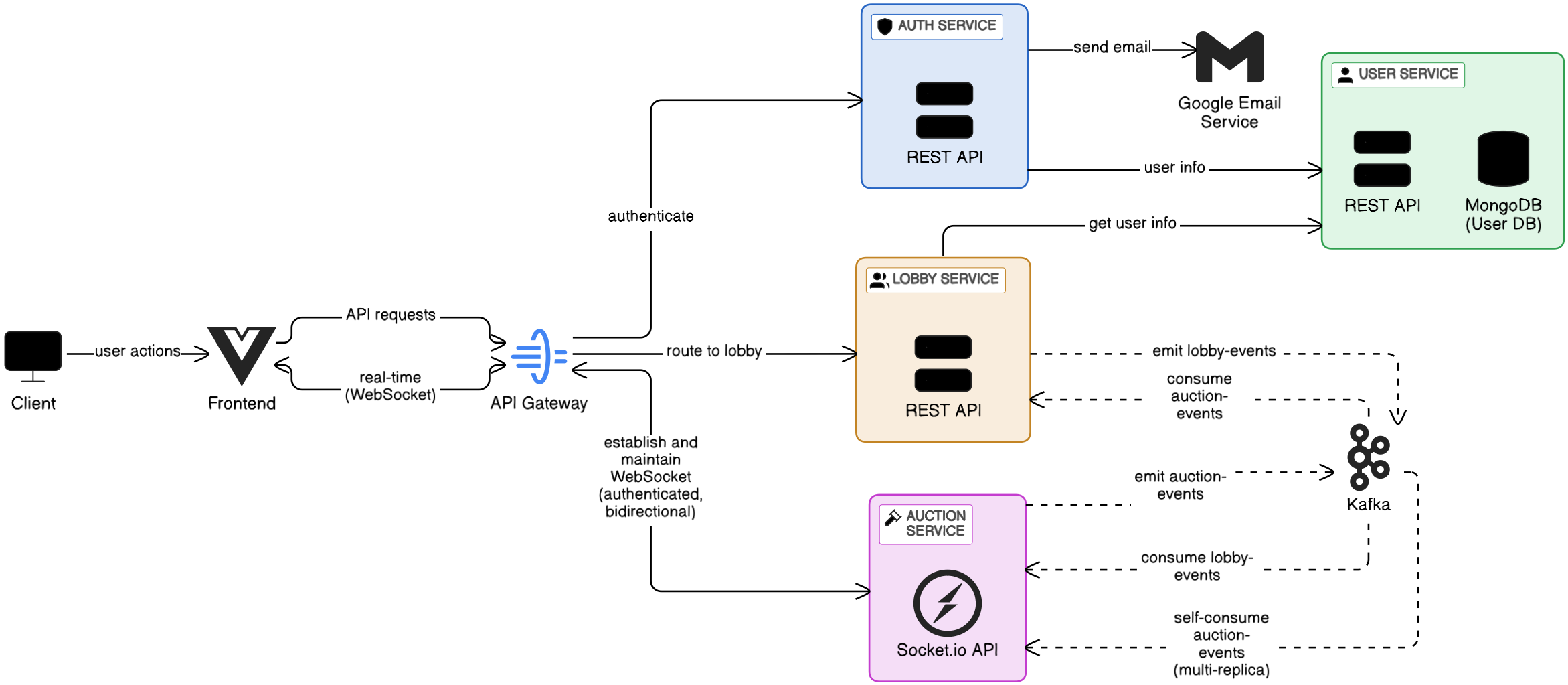
For the complete rules check here





SYSTEM OVERVIEW

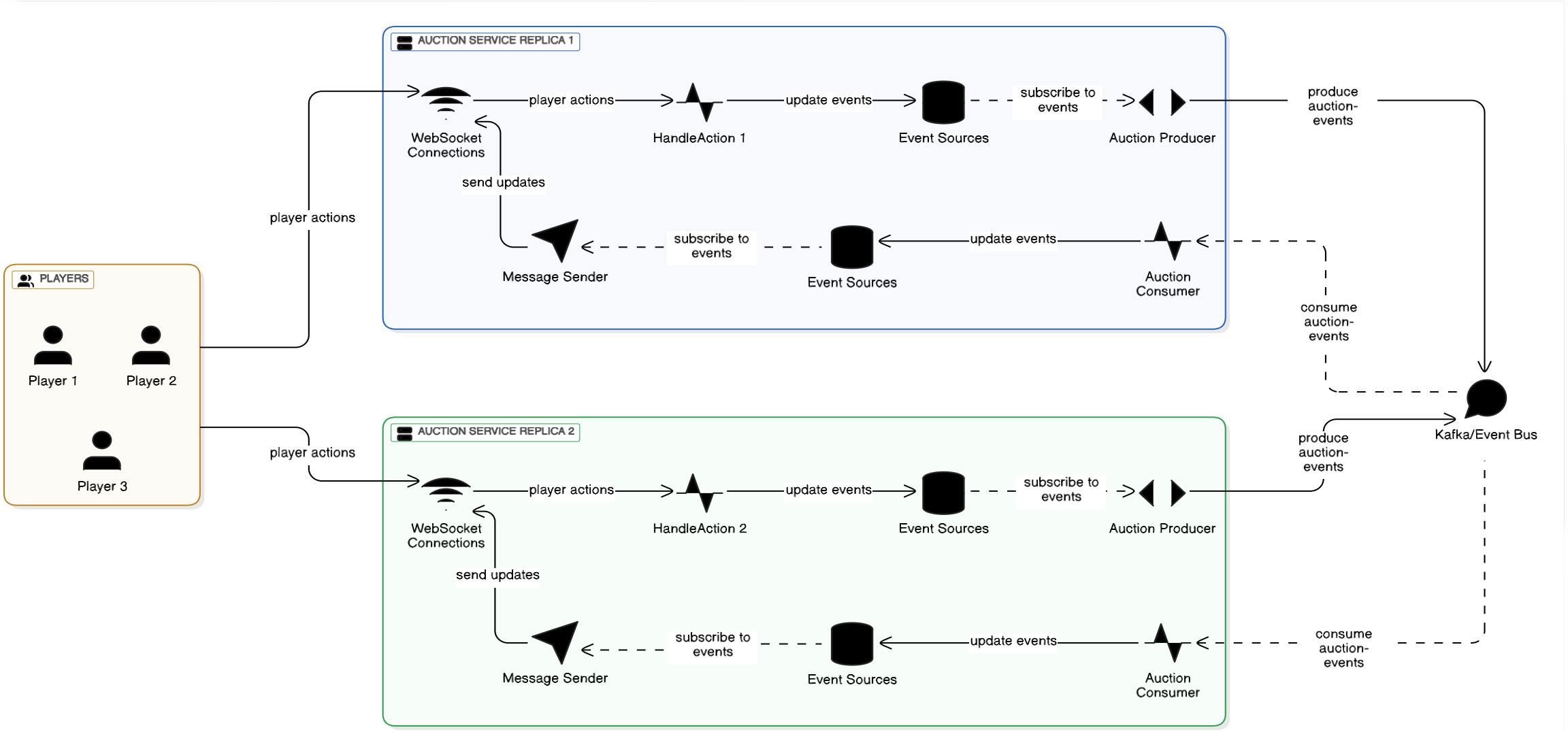
ARCHITECTURE



INTERACTIONS

FROM	TO	TYPE
Auction Service	Lobby Service	Event-driven communication
Auction Service	Frontend	WebSocket
Lobby Service	Auction Service	Event-driven communication
Auth Service	User Service	REST-API calls
Auth Service	Email Service (External)	Mail Client

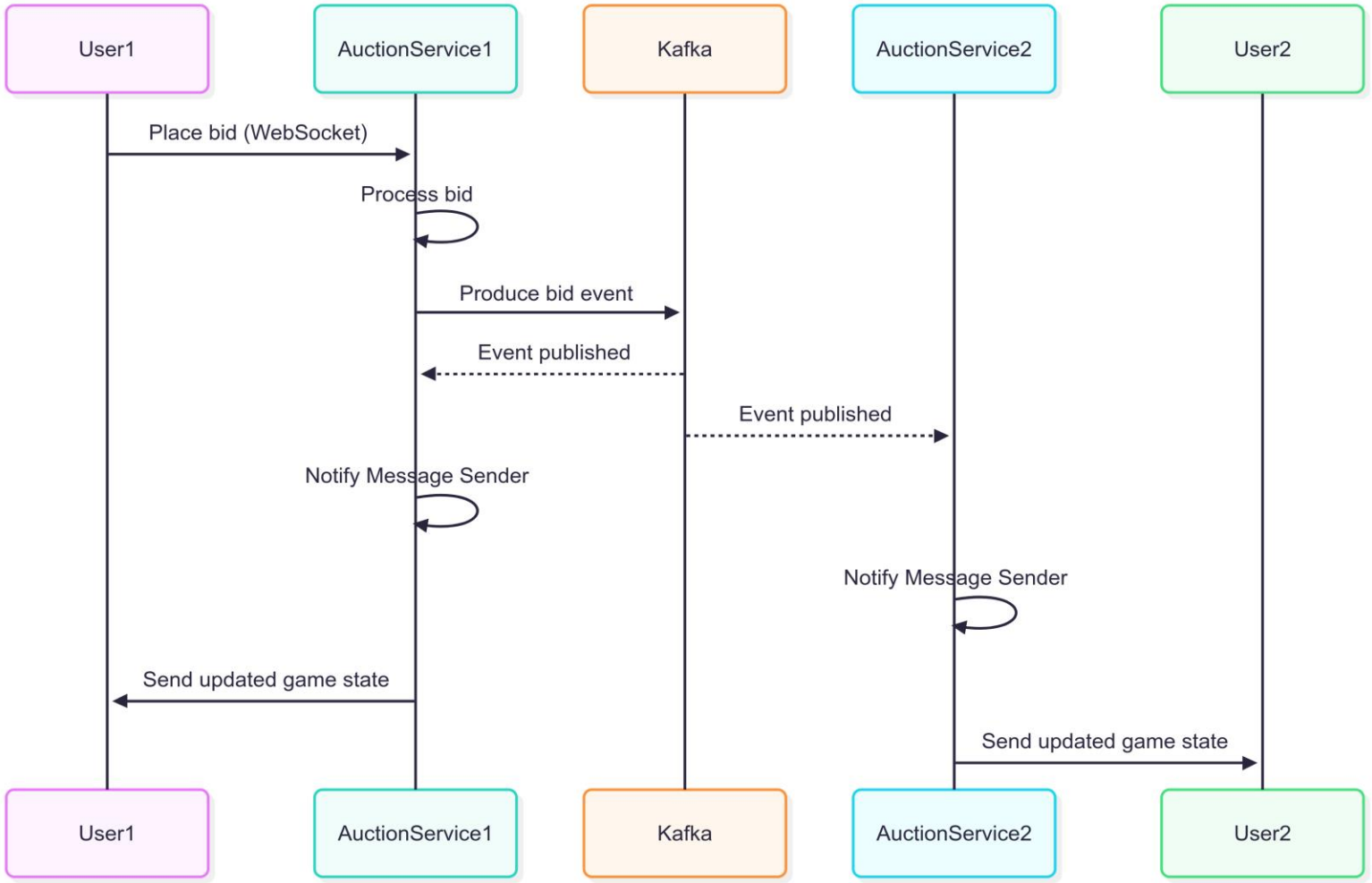
AUCTION SERVICE



AUCTION SERVICE

In this example we can see how different users playing on the same lobby, can be connected to different auction-service **replica**.

More info on the structure of the auction service here



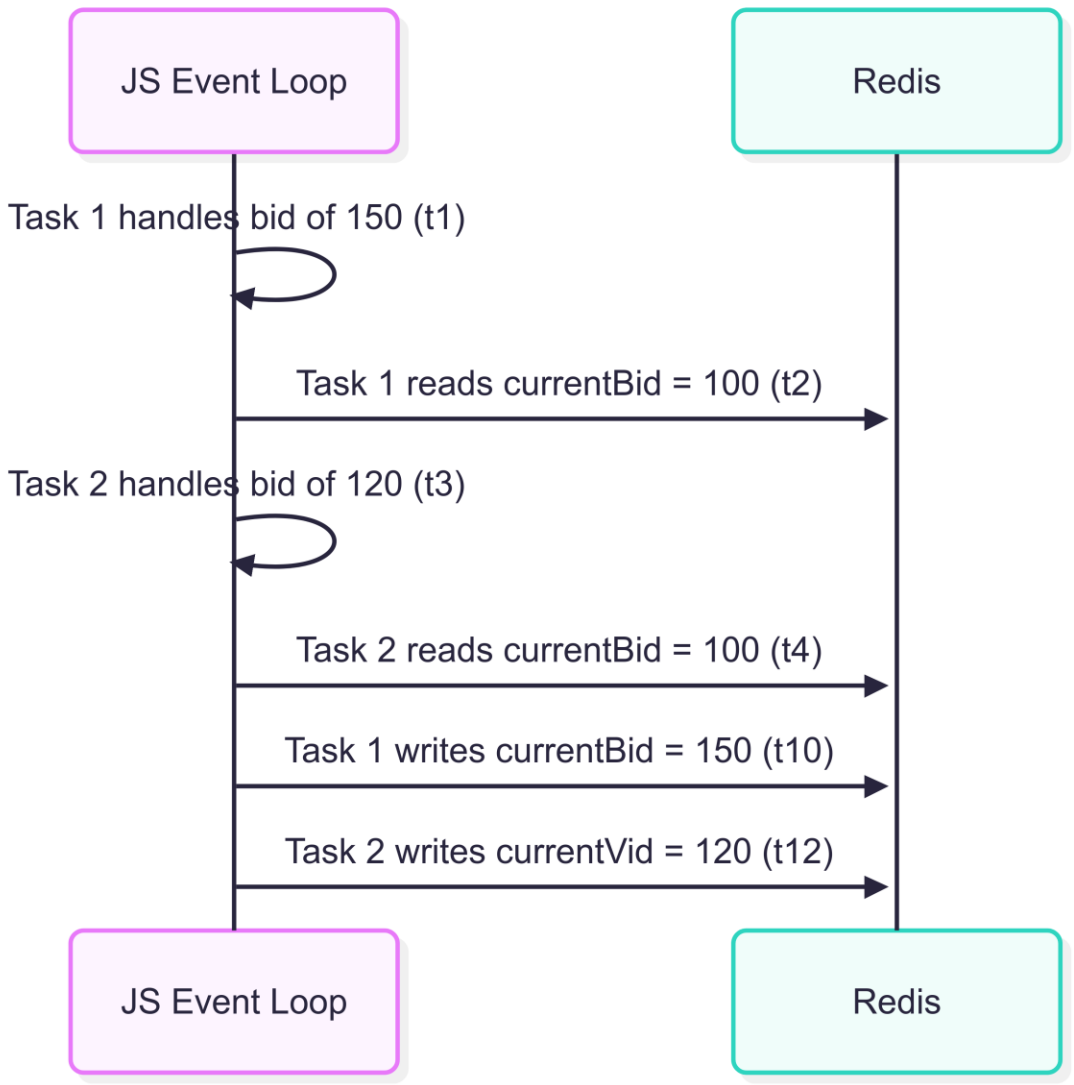
CHALLENGES

HANDLING CONCURRENCY

The auction service faces two primary concurrency challenges:

1. Single-Replica Concurrency: Within a single service instance, asynchronous operations can lead to race conditions when multiple functions attempt to read and modify the same Redis resources concurrently.
2. Multi-Replica Concurrency: When the auction service is horizontally scaled, multiple replicas may simultaneously attempt to modify the same auction state, potentially overwriting each other's changes.

More info here



TIME SYNCHRONIZATION

The auction system relies on accurate timing for critical gameplay features, particularly for auction **countdowns**. However, there's an inherent challenge: client system clocks may **differ** from the server's clock, which could lead to inconsistent experiences across different players if not properly addressed.

In a distributed system, time synchronization is a fundamental challenge due to:

- **Different System Clocks**: Client devices may have system clocks that are ahead or behind the server's clock.
- **Network Latency**: The time taken for messages to travel between client and server introduces delays.
- **Auction Timing Requirements**: For auctions to function fairly,

all clients need to see the same countdown timers.

Check here to see how synchronization was achieved

> The live demo is available here
>
> Marco Frattarola