Technical Backend Test

Imagine that you are developing a backend for a dice game.

The game consists of betting on whether the next number drawn will be even or odd. If you win, you double your bet; if you lose, you forfeit your bet.

The frontend will have access to three endpoints:

- `Wallet`: Request to retrieve the player's balance
- `Play`: Request with the bet amount and type
- `EndPlay`: Request to credit the winnings and close the play

Simple Workflow Example:

1. The frontend calls the Wallet endpoint with `clientId`, and the backend returns the player's balance.
2. The frontend calls the Play endpoint with `clientId`, `bet amount`, and `type`. The backend randomizes a number, debits the bet amount from the balance, and returns the number and result (win or lose).
3. The frontend calls the EndPlay endpoint with `clientId`, and the backend credits the winnings to the balance.

You can use HTTP requests or WebSockets for this, with a preference for WebSockets.

You may use any programming language, but Go is preferred.

We will not provide an API for requests and responses. You are expected to use your imagination to draw the API you want to add additional protection logic to your code.

Example of Additional Protections:

- A new play should not be allowed without ending the previous one.
- Bets should not exceed the available balance.

The project should be deliverable for local testing, along with a Postman collection to test your application.