Rust Multiplayer Poker Project 3 - GUI Client + Server

Thasanka Kandage, An Huynh, Francis Garcia

Overview

This project implements a multiplayer poker game system written in Rust. It consists of two main components:

- **Server Crate:**
  Handles game logic, player management, lobby coordination, and communication over TCP. It is a headless application that manages backend logic and state.

- **Player Crate (GUI Client):**
  Built using egui, this client allows players to register, log in, select a poker variant, join lobbies, ready up, and play a game. All player interaction occurs through the GUI.

Features

- **GUI-Based Gameplay:**
  Players interact via a graphical interface.

- **Realtime Multiplayer Lobbies:**
  Manage up to 6 players per lobby with a server-controlled game start over TCP.

- **Server-Driven Game Start:**
  The game begins when the required number of players join and ready up.

- **Custom Game Logic:**
  Incorporates a custom deck, shuffling mechanism, and hand evaluation logic.

- **MongoDB Integration:**
  Stores player stats and game history, making it possible for any user to view statistics.

- **Multiple Poker Variants Supported:**

    - Five Card Draw

- Seven Card Stud
- Texas Hold'em

Requirements

Before running the server and client, ensure you have:

- **Rust & Cargo:**
  Install from rustup.rs.
- **MongoDB:**
  Install from MongoDB Community Download.
- **Dependencies:**
  Refer to the dependency list provided below.

Setup Instructions

1. **Install Dependencies:**
   From both the server and client directories, install the required dependencies.
2. **Start MongoDB:**
   Ensure MongoDB is running. You can start it using:
   'mongod' in your terminal.

Running the Project

1. **Start the Server:**
   Navigate to the server crate directory and execute:
   'cargo run' in your terminal.

   The server will:

   - Listen on 0.0.0.0:8080
   - Handle incoming client connections
   - Manage game logic and state

After executing the command, it will ask you to enter the amount of users you want to create a lobby with (default is 2), enter the amount of players through the

command line. Afterwards, select which poker game variant you want to play using keys 1, 2, or 3, each depicting their own poker variant.

2. **Start the GUI Client:**
   In the client crate directory, execute:
   'cargo run' in your terminal.

   The GUI client provides:

   - Registration and login interfaces
   - Game type selection (among the three variants)
   - Lobby joining and "Ready" functionality
   - Game play management once all players are ready

Architecture

Server Crate

- Implements a TCP server using tokio.
- Tracks players and lobbies in shared memory using Arc<Mutex<HashMap<...>>>.
- Uses MongoDB for:
  - Player authentication and statistics
  - Lobby management
  - Saving completed game history
- Sends JSON-formatted updates to clients over persistent TCP connections.

Client Crate (GUI)

- Developed with egui and eframe.
- Provides a user interface for login/registration, game selection, lobby status, and game view.
- Maintains a TCP connection to the server.

- Background threads handle:
    - Lobby polling
    - Receiving real-time server broadcasts (e.g., a "game_start" signal)
- Automatically updates the UI based on server messages.

Game Flow

1. **Server Setup:**
   The server is started via the CLI where the number of players and game variant are specified.

2. **Client Connection:**
   The player launches the GUI client, connects to the server, and either registers or logs in.

3. **Lobby Interaction:**
   Players join a shared waiting room (lobby) that can accommodate up to 6 players. There is also an option to view any user's stats via a search bar.

4. **Game Start:**
   When the required number of players is reached, the server begins the game. The server deals the cards and notifies all connected clients via the TCP connection.

5. **Gameplay:**
   The game features turn-based gameplay and betting mechanics (check, bet/call, raise, fold). The server manages full turn-based gameplay and betting rounds.

Poker Variants

Five Card Draw

- Each player receives 5 cards.
- A custom ranking system is used for hand evaluation.
- The winner is determined and recorded in MongoDB.

- Game state resets after completion.

Seven Card Stud

- Each player is dealt 7 cards (a mix of face-down and face-up).
- Multiple betting rounds occur as cards are dealt.
- Players form the best 5-card hand from their 7 cards.
- The winner is determined at showdown and recorded in MongoDB.
- Game state resets after completion.

Texas Hold'em

- Each player receives 2 private (hole) cards.
- Five community cards are dealt in rounds (flop, turn, and river).
- The game begins with two forced bets:
    - **Small Blind:** A smaller bet to build the pot.
    - **Big Blind:** A larger bet that further contributes to the pot.
- Players form the best 5-card hand using any combination of hole and community cards.
- Multiple betting rounds occur throughout the game.
- The winner is determined at showdown and recorded in MongoDB.
- Game state resets after completion.

Betting Mechanics

- **Check:**
  When no bet has been made, a player may choose to check (i.e., not wager additional chips) while remaining in the hand.

- **Bet/Call:**
  If a bet is placed, a player may call by matching the current bet to continue in the round.

- **Raise:**
  A player can increase the current bet, forcing others to either match the new bet or fold.

- **Fold:**
  A player may opt to fold, thereby withdrawing from the current hand and forfeiting any chips already bet.

References/Dependencies

Crate	Version	Description
futures-util	0.3.31	Utilities for working with asynchronous futures
itertools	0.14.0	Extra iterator adaptors, functions, and macros.
mongodb	3.2.3	MongoDB driver for Rust with asynchronous support.
once_cell	1.17	Single assignment cells and lazy static values.
rand	0.8	Random number generation library.

serde_json	1.0.140	Serialization and deserialization for JSON using Serde.
tokio	1.44.1	Asynchronous runtime for Rust with full features enabled.
eframe	0.27	Framework for building native GUI apps using egui.