**Introduction**:

You are required to implement a Hi-Lo card game. Game Card deck is standard 52 cards <https://en.wikipedia.org/wiki/Standard_52-card_deck>. Each round player must guess if next card drawn from the deck is equal, higher or lower to the base card. Card ranks are from 2 to 10. Jack, Queen, King and Ace have rank 10.

**Gameplay requirements**:

1. When round is started frontend client (React/Angular) sends to server (Spring) StartRoundRequest. Client receives and must display base card from StartRoundRequest.
2. Client has 10 seconds to make an action. During this time player has a choice of three actions - 'higher', 'lower' or 'equal'.
3. When countdown reaches zero backend stops accepting actions and responds „TIME\_OUT“ to the following query
4. Server checks internally player action and compares base card rank to result card rank. - If player action was correct, player gets a correct answer point - If player action was incorrect, player does not get a correct answer point
5. Previous result card became new 'base card' and is again displayed to client as in step nr 1
6. Player has three lives and afterwards the game is over

Steps 1 to 6 can be implemented without database.

1. Make it possible to register player’s name to database and add every finished game with the count of correct answers and time of play to the database associated with the player.
2. Make scoreboard with the possibility to sort by duration or correct answers.
3. Make it possible to click on the name and show all this player’s games.
4. Have sufficient unit test coverage

**General implementation requirements:**

1. Separate client and server applications.
2. Minimal use of external frameworks.
3. Code must compile, be easily readable and follow 'Clean Code' principles
4. Provided project skeleton is only to get started. Existing objects can be enhanced as necessary.
5. Do not over complicate. Simplest solution is most likely the correct one.

**Deliverables:**

1. Zip file with sources

2. Instruction how to run both client and server