

The gaming app: Rock-Scissor-Paper

(Test documentation)

1. Introduce about the app

We have created a gaming app 'Rock Paper Scissor' with two game modes: Human vs Human, Human vs Computer. Player can enter his/her own name and the name will be displayed in the game playing screen.

In mode 'Human vs Human':

Firstly, each player will make a choice of rock/scissor/paper. The result will be displayed that shows that who is a winner or the match is draw. Each player's score will be saved to the history table, which has a marker to show who won and who lost. Game mode as well as selecting player will be locked during the match. After finishing a match, player can play again or end game.

In mode 'Human vs Computer':

Firstly, player will make a choice of rock/scissor/paper. Computer is picked randomly. The result will be displayed that shows that player win or lose or the match is draw. Player's and computer's score will be saved to the history table, which has a marker to show who won and who lost. Game mode as well as selecting player will be locked during the match. After finishing a match, player can play again or end game.

History table to save result of matches in which: Header saves total score of each player. Each row is result of a match Show the latest 10 results.

End game button will reset results and allow user to start a new game.

2. Testing types:

2.1 Unit testing (React testing library): write unit test cases for all components in the app to ensure that all elements and functions are rendered and called as expected. Unit tests help catch bugs early in the development process and ensure that changes to the code do not break existing functionality.

Here is list of test cases and name of each test cases mention it's purpose.

- Header component(**Header.test.js**):

```
PASS src/components/header/_Header-test_/header.test.js
Header component
  ✓ renders game mode selection (87 ms)
  ✓ renders player 1 name input (10 ms)
  ✓ updates game mode when selected (17 ms)
  ✓ updates player 1 name when input changes (10 ms)
  ✓ updates player 2 name when input changes in game mode 1 (19 ms)
  ✓ does not render player 2 name input in game mode 2 (10 ms)
  ✓ renders header component without crashing (5 ms)
  ✓ disables game mode select field during gameplay (8 ms)
  ✓ disables game mode select field after game has ended (9 ms)
  ✓ enables player 2 name input field in game mode 1 (10 ms)
  ✓ calls handleGameModeChange function when game mode is changed (9 ms)
  ✓ calls handlePlayer2NameChange function when player 2 name is changed (10 ms)

Test Suites: 1 passed, 1 total
Tests:       12 passed, 12 total
Snapshots:   0 total
Time:        2.212 s, estimated 3 s
```

- Player selection component (**UserSelectionForm.test.js**):

```
PASS src/components/userSelectionForm/_test_/UserSelectionForm.test.js
userSelectionForm
  ✓ should render playerName (171 ms)
  ✓ text 'Picked random' should be rendered when player is 'computer' (13 ms)
  ✓ should render playing icons when choiceOfPlayer is null (18 ms)
  ✓ should render 'You picked' after player made a choice (9 ms)
  ✓ should call the setChoiceOfPlayer function when player makes a choice (30 ms)
  ✓ Human vs Computer: should call the setUpdatedResult function after player makes a choice (18 ms)
  ✓ humanvs computer: should call the setResults function two times after player make a choice (17 ms)

  ✓ human vs computer: should call the setWinner function after player make a choice (19 ms)
  ✓ Human vs Human: should call the setUpdatedResult function after both players made their choice (17 ms)
  ✓ human vs human: should call the setResults function when player make a choice (16 ms)
  ✓ human vs human: should call the setWinner function after both players made their choices (14 ms)
```

- Selection component (**Selection.test.js**)

```
PASS src/components/selection/_test_/Selection.test.js
Selection
  ✓ renders the score (60 ms)
  ✓ renders the winner after each match finishes (11 ms)
  ✓ calls the setPlaying function when the Play again button is clicked (17 ms)
  ✓ calls the setUpdatedResult function when the Play again button is clicked (7 ms)
  ✓ calls the setGameMode function when the End game button is clicked (8 ms)
  ✓ calls the setPlayer1TotalScore function when the End game button is clicked (7 ms)
  ✓ calls the setPlayer2TotalScore function when the End game button is clicked (8 ms)
  ✓ calls the setUpdatedResult function when the End game button is clicked (7 ms)
  ✓ calls the setPlaying function when the End game button is clicked (9 ms)
  ✓ calls the setResults function when the End game button is clicked (7 ms)
```

- History component(**History.test.js**)

```
PASS src/components/history/__history__/History.test.js
History with number of plays is less than 10
  ✓ should render the correct table headers (102 ms)
  ✓ should render the correct table rows (15 ms)
  ✓ should call setPlayer1TotalScore and setPlayer2TotalScore with the correct values (9 ms)
  ✓ should render the correct number of rows when both players have less than 10 results (46 ms)
History with number of plays is larger than 10
  ✓ should render only the latest 10 results in History table (17 ms)
```

2.2 E2E testing: run test cases according to user's flows. In our app, there are some user flows:


Flow 1: A player chooses to play against another player. After both players enter their name, they can start game. Each player will make one his choice, the result will be calculated and shown on the screen. Total score as well as players' choice are shown in history table.

Flow 2: A player chooses to play against computer. He enter a name. Start gaming by picking a choice, computer will make a choice randomly. The result will be calculated and shown on the screen. Total score as well as player and computer's choice are shown in history table.

Flow 3: Two players can play against after finishing a match. similar to flow 1


Flow 4: A player can play against computer after finishing a match. similar to flow 2

Flow 5 &6 (Human vs Human & Human vs computer): Player can choose to end game and start a new game.

 Specs

✓ 6 ✗ -- ⌂ --

▼ ↺

 **E2ETesting** cyjs 00:04

▼ E2E test

- ✓ Human vs Human: Two players can play against each other and their score as well as choice is updated correctly
- ✓ Human vs Human: Player can play again after finishing a match
- ✓ Human vs Human: Player can choose to end game and start a new game
- ✓ Human vs Computer: A player can play against the computer and the score is updated in the history table
- ✓ Human vs Computer: Player can play again after finishing a match
- ✓ Human vs Computer: Player can choose to end game and start a new game