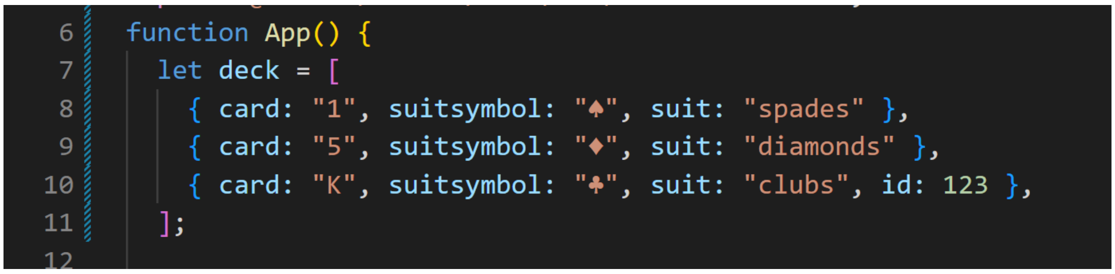
Blackjack tech test by Ekaterina Mulyukova

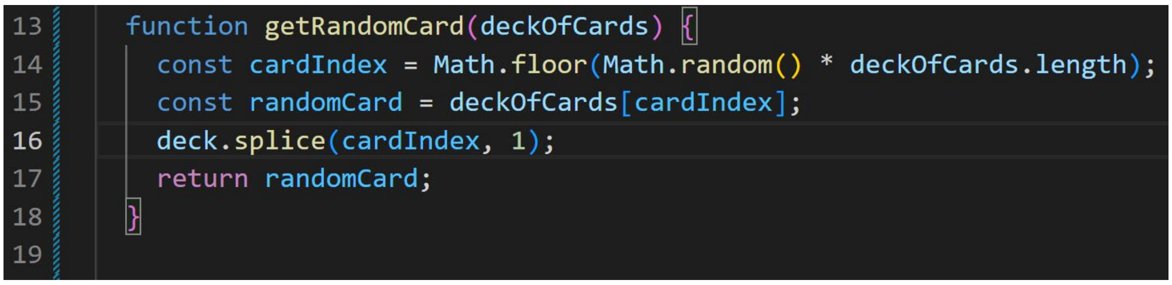
I've opted to code blackjack in JavaScript using React. To start with, I've created the key element of the game, the deck of cards. Below is the code for my trial deck containing only 3 cards. After doing more tests with this minimal deck, I'll code a full 52-card one.



```
function App() {
  let deck = [
    { card: "1", suitsymbol: "♠", suit: "spades" },
    { card: "5", suitsymbol: "♦", suit: "diamonds" },
    { card: "K", suitsymbol: "♣", suit: "clubs", id: 123 },
  ];

```

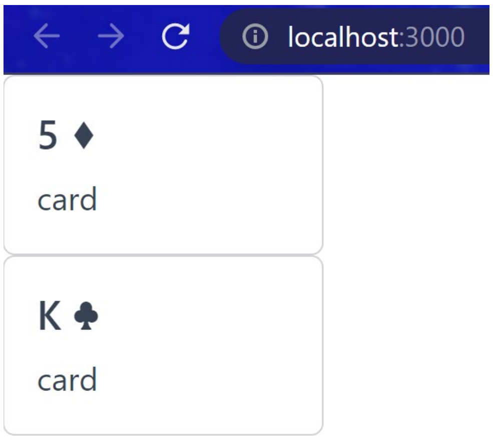
Next thing would be to let a user draw their initial two cards. I wrote a function which picks a random card from my deck and then removes that card from it, so the same card couldn't be drawn again in the current game.



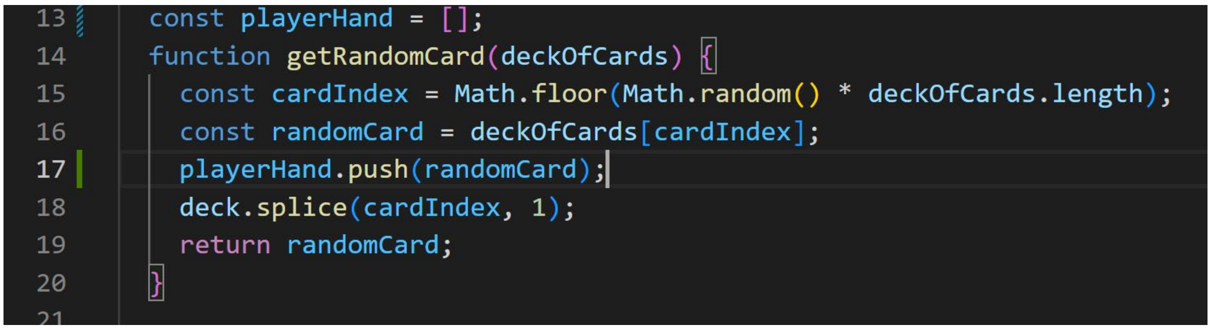
```
function getRandomCard(deckOfCards) {
  const cardIndex = Math.floor(Math.random() * deckOfCards.length);
  const randomCard = deckOfCards[cardIndex];
  deck.splice(cardIndex, 1);
  return randomCard;
}

```

I'm assigning the result of this function to a new variable called firstCard and then doing the same for secondHand. Now I need to add a visual representation for these two initial cards. To speed up the process, I'm using a Card component from CoreUI component library for React to be the body of my 'cards' and displaying the values of firstCard and secondCards on them. That's how the app looks at the moment:



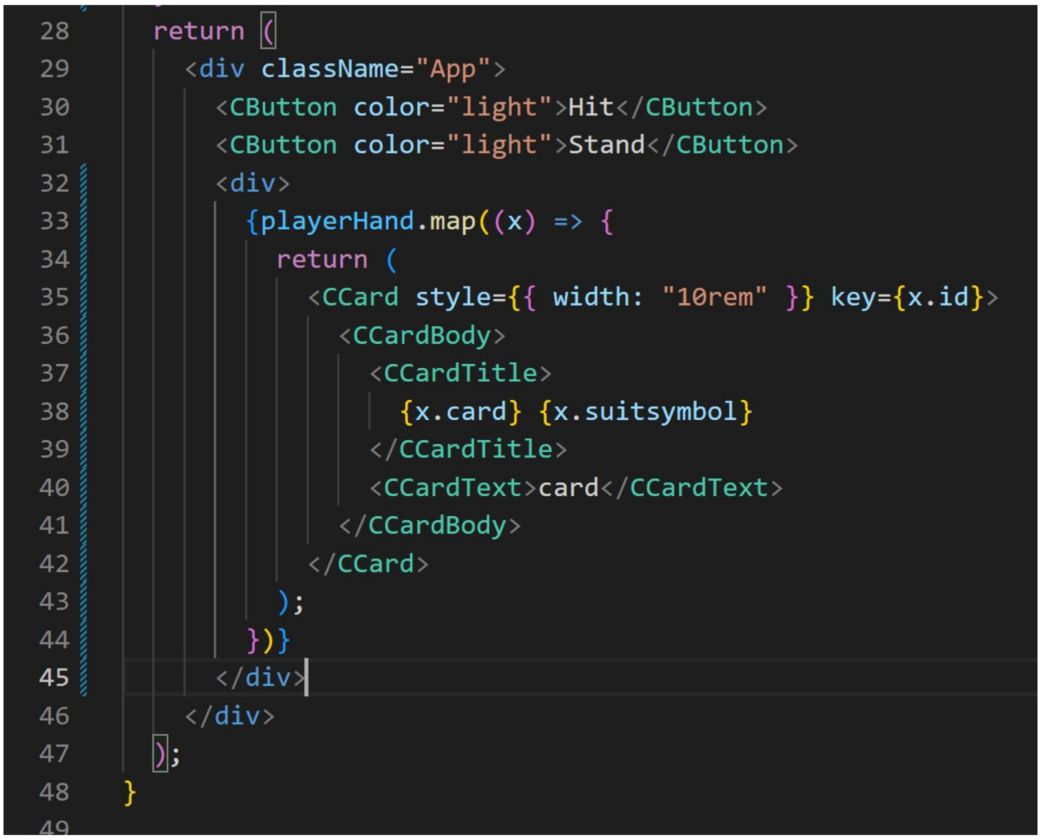
Now you can't do anything in the game, so my next step would be to add some buttons! I've added 'Hit' and 'Stand' buttons and started writing a function to be resolved on click of 'Hit' button. It made me think, how would I code my app to create and display a newly drawn card? And then how would I calculate the total score of my separate variables of cards? It would make sense to create a new variable that would keep my cards in one place, a hand! I've created a new array called playerHand added a line of code to my first two initial cards to be added to the 'hand'.



```
const playerHand = [];
function getRandomCard(deckOfCards) {
  const cardIndex = Math.floor(Math.random() * deckOfCards.length);
  const randomCard = deckOfCards[cardIndex];
  playerHand.push(randomCard);
  deck.splice(cardIndex, 1);
  return randomCard;
}
```

Replaced variables firstCard and SecondCard with 0 and 1 indexes of the new playerHand array.

Changed the code so it checks and maps the playerHand to display the cards currently drawn.

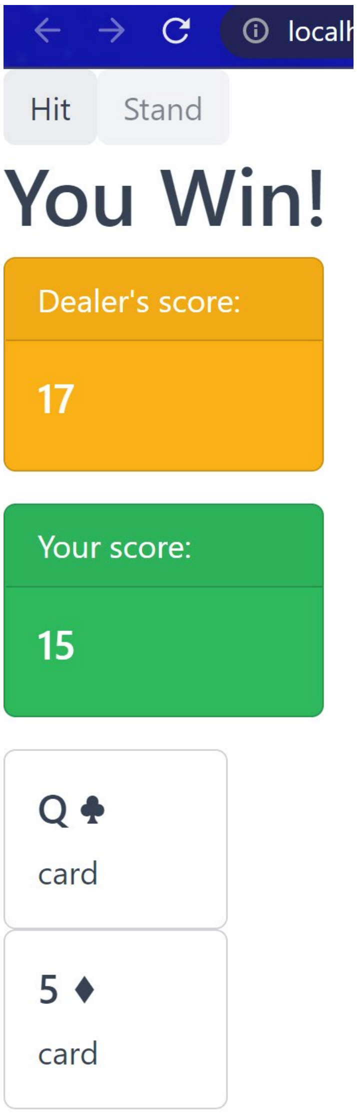


```
  return (
    <div className="App">
      <CButton color="light">Hit</CButton>
      <CButton color="light">Stand</CButton>
      <div>
        {playerHand.map((x) => {
          return (
            <CCard style={{ width: "10rem" }} key={x.id}>
              <CCardBody>
                <CCardTitle>
                  {x.card} {x.suitsymbol}
                </CCardTitle>
                <CCardText>card</CCardText>
              </CCardBody>
            </CCard>
          );
        })}
      </div>
    </div>
  );
}
```

Added functionality to Hit button, however nothing happens on the page, the state is not being refreshed. That's when React's useState comes useful, I've adapted my code using it and now a newly drawn card is displayed immediately.

Now, before pressing 'Stand' button, it would be good to know the current score of your hand. I've added a card displaying the score that changes colour depending on your hand's score (under or over 21). Disabled the option to 'Hit' if your hand isn't valid. Later on it will be useful when the player have a choice of setting Ace to either 1 or 11.

Stand button evaluates the score, triggers the display of a game outcome depending on a player's score, disables self. At the moment a player wins just by scoring under 21 points which isn't in real blackjack's spirit, so I'm introducing a 'dealer' score to compete with. Before making the dealer's behaviour a bit more realistic, I'm hardcoding their score to always be 17. At the moment, a player wins even if the dealer's score is higher, so I'll need to change that.



You Win!

Dealer's score:

17

Your score:

15

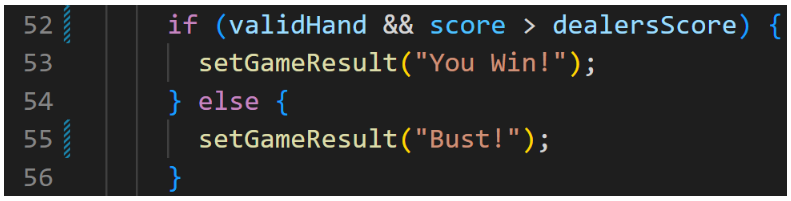
Q ♣

card

5 ♦

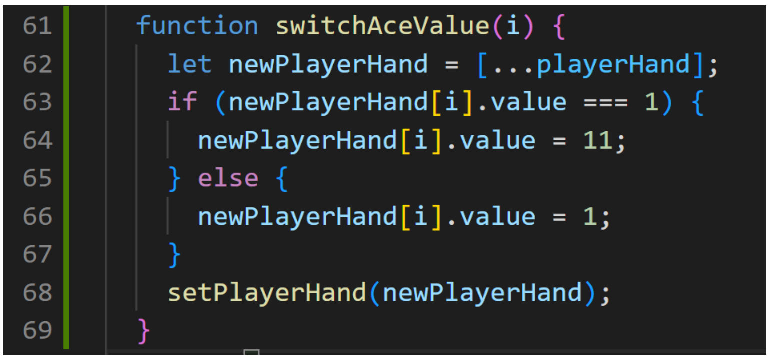
card

Added one more condition to set the game result to win:



```
    if (validHand && score > dealersScore) {
      setGameResult("You Win!");
    } else {
      setGameResult("Bust!");
    }
```

The game works only on condition that there are no aces in deck! I need to adapt my code to be able to deal with their conditional value (either 1 or 11). To keep it more in the spirit of real-life blackjack, I've opted to give a player a choice for their ace's value instead of setting it automatically for them. After adding some special behaviour to aces, I've also added extra buttons for the player to click to switch the value. To start with, I’m setting the default value to 1. Added a button only for Ace cards that performs this function on click:



```
function switchAceValue(i) {
  let newPlayerHand = [...playerHand];
  if (newPlayerHand[i].value === 1) {
    newPlayerHand[i].value = 11;
  } else {
    newPlayerHand[i].value = 1;
  }
  setPlayerHand(newPlayerHand);
}
```

The function switches the Ace values from 1 to 11 and vice versa. With the use of useState, the score automatically updates as well.

Added ‘Play Again’ button that resets the game.

Updated the layout of the game’s components with the grid.

Updating the deck to contain 52 cards, moved it to a separate file, dealer always draws two same cards from it to have the score of 17.

Made three illustration for the dealer cat, now he displays different emotion depending on the game result.