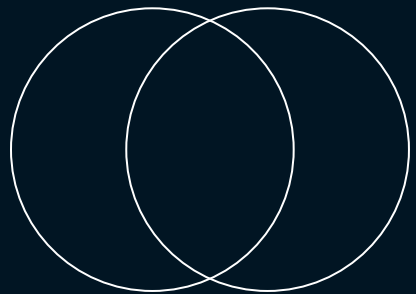


JavaScript-Project

Dobble game

최지윤



Dobble game



인원수 맞춰 카드 분배

플레이어 수에 따라 카드를 나눠주거나, 각자 한 장씩 갖습니다.

시작 카드 놓기

가운데에 카드를 한 장 뒤집어 놓습니다.

공통 그림 찾기

자신의 카드와 가운데 카드를 비교하여 똑같은 그림을 빠르게 찾습니다.

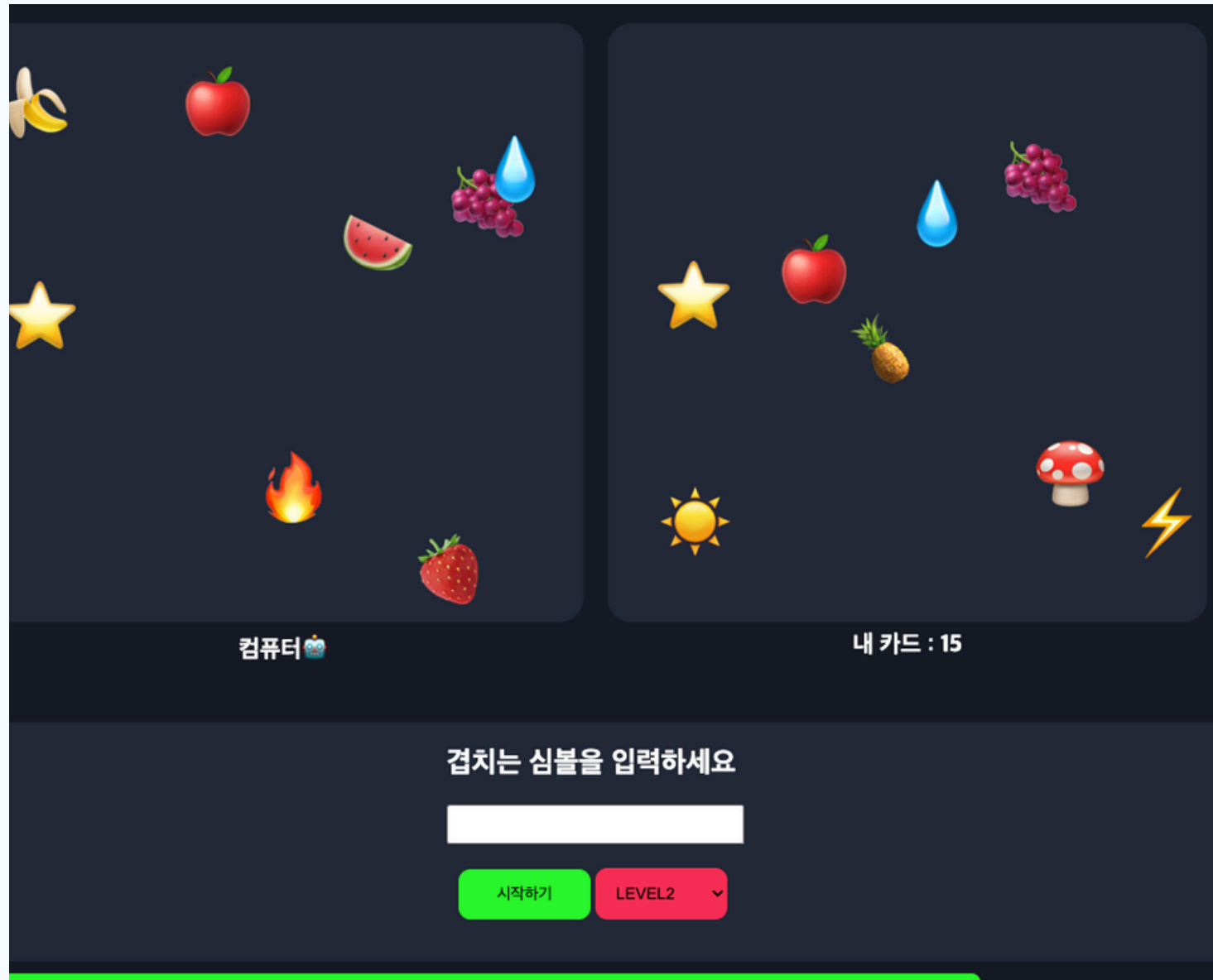
외치며 내려놓기

공통 그림을 찾으면 그림 이름을 외치면서 카드를 가운데에 내려놓습니다

승리

자신의 손에 남은 카드를 가장 먼저 모두 내려놓는 사람이 승리합니다.

Dobble game



카드

두 카드에는 각각 8개의 랜덤 심볼이 있으며, 어떤 두 카드든 최소 1개의 같은 심볼이 존재한다.

공통 그림 찾기

플레이어는 화면에 보이는 두 카드에서 겹치는 심볼의 이름을 입력한다

진행

입력하면 플레이어의 남은 카드 수가 1 감소하고, 오답이면 1 증가한다.









승리

제한 시간 안에 카드 수를 0개로 만들면 승리, 시간이 끝났는데 카드가 남아 있으면 실패한다.








난이도 설정

난이도에 따라 시작 카드 수가 달라진다.

실행화면



컴퓨터 🖥️



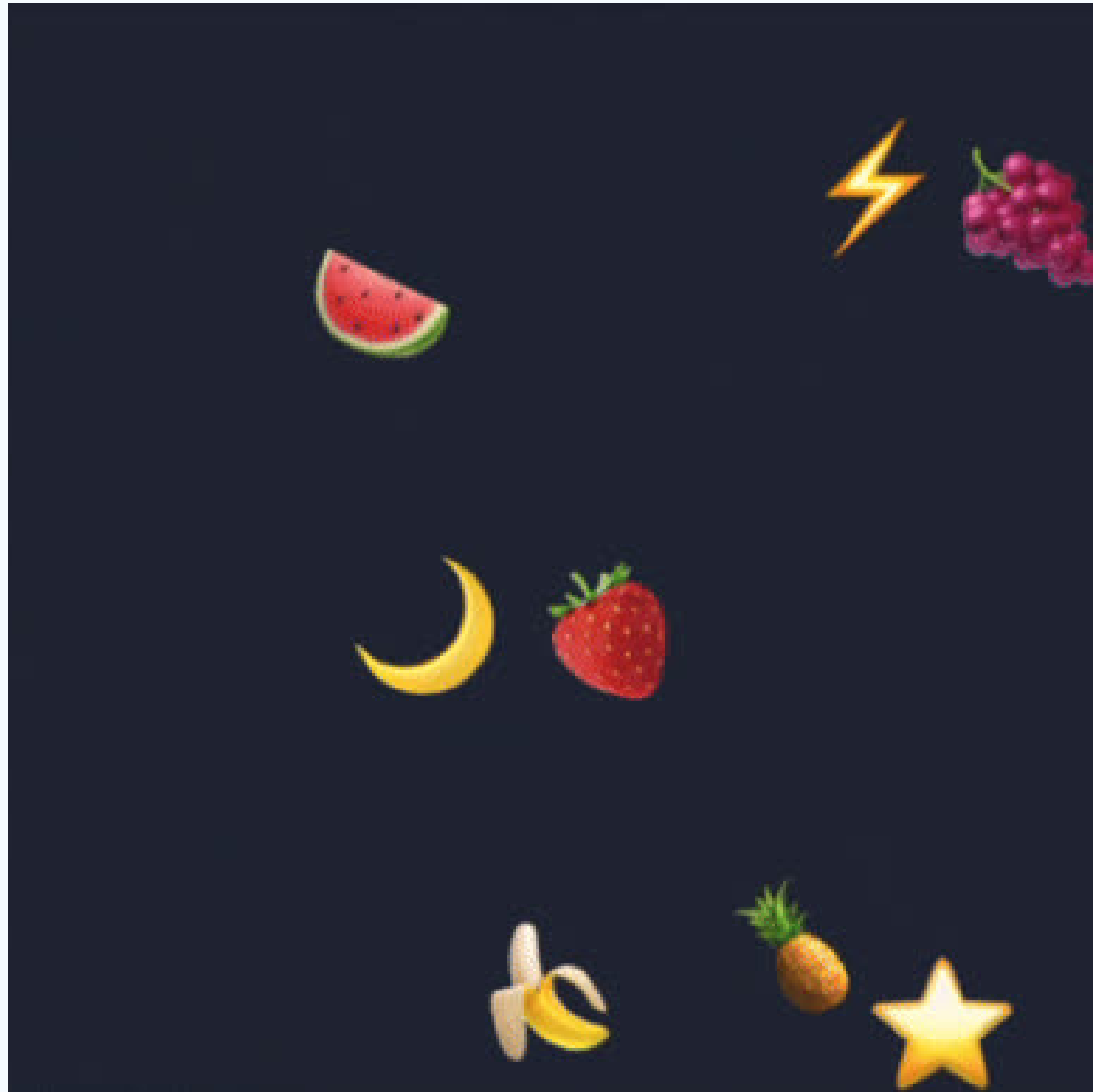
내 카드 : 10

겹치는 심볼을 입력하세요

시작하기

LEVEL1 ▾

카드



실습시간에 배운 예제를 응용해서 움직이는 심볼 구현

총 15가지의 심볼중에 한 카드에 랜덤으로 8개의 카드가 나타나도록 구현

정답 or 오답 처리시 다시 랜덤으로 카드가 다시 그려지도록 구현

```
// 8개의 랜덤 심볼 뽑기
function pickSymbols() {
  const picks = [];
  while (picks.length < per) {
    const pick = Math.floor(Math.random() * symbols.length);
    if (!picks.includes(pick)) picks.push(pick);
  }
  const card = picks.map((i) => symbols[i]);
  return card;
}

// 8개의 심볼이 랜덤으로 들어있는 카드 만들기
function makeRandomCard() {
  my = pickSymbols();
  your = pickSymbols();
}
```



```

// 겹치는 심볼의 이름 찾기
function findSymbolsNames() {
  const matches = [];
  for (let a of my) {
    for (let b of your) {
      if (a.name === b.name) {
        matches.push(a.name);
      }
    }
  }
  return matches;
}

```

```

//화면에 카드 출력
function drawCards() {
  myCard.innerHTML = "";
  yourCard.innerHTML = "";

  my.forEach((sym) => {
    mySymbols.push(new MovingSymbols(sym, myCard));
  });

  your.forEach((sym) => {
    yourSymbols.push(new MovingSymbols(sym, yourCard));
  });

  clearInterval(moveTimer);
  moveTimer = setInterval(() => {
    mySymbols.forEach((s) => s.move());
    yourSymbols.forEach((s) => s.move());
  }, 1000 / 60);
}

```

```

function submitAnswer() {
  const answer = document.getElementById("answer").value;
  const correct = findSymbolsNames();

  if (correct.includes(answer)) {
    myCardsLeft--;
    feedback.textContent = "정답";
  } else {
    myCardsLeft++;
    feedback.textContent = "오답";
  }

  myCountEl.textContent = myCardsLeft;
  answerInput.value = "";

  if (myCardsLeft === 0) {
    alert("성공하셨습니다.");
    clearInterval(timerId);
    clearInterval(moveTimer);
    isRunning = false;
    return;
  }

  makeRandomCard();
  drawCards();
}

```

타이머

127.0.0.1:5500 내용:

 시간 초과! 다시하세요!

확인

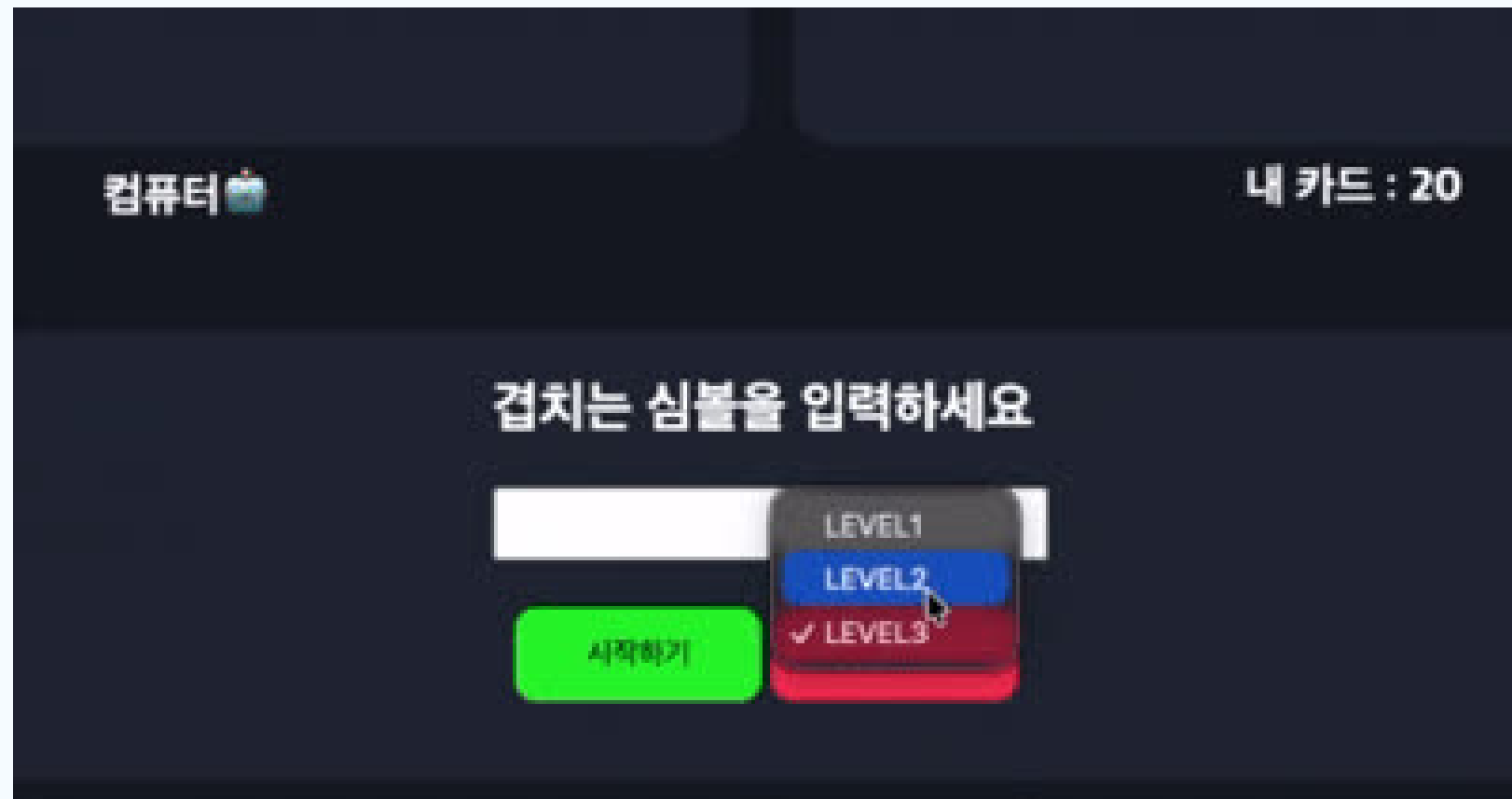
```
#timeBar {  
  width: 100%;  
  height: 15px;  
  margin-top: 10px;  
  transition: width 1s linear;  
  border-radius: 8px;  
}
```

변수에 제한 시간을 저장하고, setInterval()을 사용해 매초 1씩 감소시키고 이를 style과 css로 처리해서 바 형태로 보이도록 구현

타이머가 끝나면 시간 초과를 알리는 창이 뜨도록 구현

```
function startTimer() {  
  clearInterval(timerId);  
  timeBar.style.width = "100%";  
  timeBar.style.background = "rgb(40, 249, 40)";  
  let remaining = duration;  
  
  timerId = setInterval(() => {  
    remaining--;  
    const percent = (remaining / duration) * 100;  
    timeBar.style.width = percent + "%";  
  
    if (percent < 30) timeBar.style.background = "red";  
    else if (percent < 50) timeBar.style.background = "orange";  
  
    if (remaining <= -1) {  
      clearInterval(timerId);  
      clearInterval(moveTimer);  
      alert("🕒 시간 초과! 다시하세요!");  
      isRunning = false;  
    }  
  }, 1000);  
}
```

레벨 설정

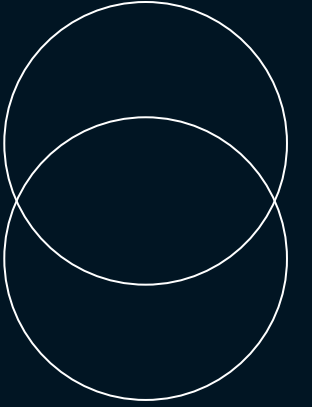


레벨에 따라 카드의 수가 달라지도록 구현

```
function level() {  
  const lv = Number(document.getElementById("level").value);  
  
  if (isRunning) {  
    alert("레벨이 변경되었습니다. 시작하기를 눌러주세요");  
  
    clearInterval(timerId);  
    clearInterval(moveTimer);  
    isRunning = false;  
  }  
  currentLevel = lv;  
  total = LEVEL_CARDS[currentLevel];  
  
  myCardsLeft = total;  
  myCountEl.textContent = myCardsLeft;  
  answerInput.value = "";  
}
```


마지막으로..

처음에 어떤 기능들이 필요한지 정리하고 시작했음에도 중간에 추가적으로 생각나는 기능들도 구현하다 보니 코드가 많이 지저분해지기도 하고, 기능 구현에 필요한 로직을 찾아보기도하면서 내 코드에 어떻게 적용해야할지 고민하는 부분들이 어려웠지만 이 부분들을 해결해가는 과정에서 많은 공부가 되었다.



Thank You